Management of spatial data for visualization on mobile devices

by

Fangli Ying

NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

A thesis presented in fulfilment of the requirements for the Degree of

Doctor of Philosophy

Supervisors: Dr Adam C.Winstanley & Dr Peter Mooney

Department of Computer Science

National University of Ireland Maynooth

Co. Kildare, Ireland

November, 2012

This thesis has not been submitted in whole, or in part, to this, or any other University for any other degree and is, except where otherwise stated, the original work of the author.

Signed: Fangli Ying
Date: 2011.11.29

Abstract

Vector-based mapping is emerging as a preferred format in Location-based Services(LBS), because it can deliver an up-to-date and interactive map visualization. The Progressive Transmission(PT) technique has been developed to enable the efficient transmission of vector data over the internet by delivering various incremental levels of detail(LoD). However, it is still challenging to apply this technique in a mobile context due to many inherent limitations of mobile devices, such as small screen size, slow processors and limited memory. Taking account of these limitations, PT has been extended by developing a framework of efficient data management for the visualization of spatial data on mobile devices.

A data generalization framework is proposed and implemented in a software application. This application can significantly reduce the volume of data for transmission and enable quick access to a simplified version of data while preserving appropriate visualization quality. Using volunteered geographic information as a case-study, the framework shows flexibility in delivering up-to-date spatial information from dynamic data sources.

Three models of PT are designed and implemented to transmit the additional LoD refinements: a full scale PT as an inverse of generalisation, a view-dependent PT, and a heuristic optimised view-dependent PT. These models are evaluated with user trials and application examples. The heuristic optimised view-dependent PT has shown a significant enhancement over the traditional PT in terms of bandwidth-saving and smoothness of transitions.

A parallel data management strategy associated with three corresponding algorithms has been developed to handle LoD spatial data on mobile clients. This strategy enables the map rendering to be performed in parallel with a process which retrieves the data for the next map location the user will require. A view-dependent approach has been integrated to monitor the volume of each LoD for visible area. The demonstration of a flexible rendering style shows its potential use in visualizing dynamic geoprocessed data. Future work may extend this to integrate topological constraints and semantic constraints for enhancing the vector map visualization.

Acknowledgements

It would not have been possible to write this doctoral thesis without the help and support from my supervisors, colleagues, friends and family. I would like to express my sincere thanks to all of you.

First and foremost, I owe my deepest gratitude to my supervisors Dr Adam C.Winstanley and Dr. Peter Mooney. Dr. Adam C.Winstanley, who is Head of Department of Computer Science, provided continued encouragement, kind support and useful suggestions throughout my research. Dr. Peter Mooney, who is an EPA Research Fellow, provided guidance, unsurpassed knowledge, and great help. Without their support and assistance this study would not have been successful.

I also would like to acknowledge the supervisory role of Dr. Padraig Corcoran, who is currently a researcher at the School of Computer Science and Informatics in University College Dublin. While Padraig was a researcher and lecturer at the Department of Computer Science at NUI Maynooth, he provided input to early development of the concepts of using progressive transmission in mobile visualization of spatial data outlined in this thesis. I enjoyed working with him and learned a lot from his advice, work ethic, and research experience. I look forward to potentially working with Padraig again in the future.

I gratefully acknowledge my sources of funding from the China Scholarship Council(CSC) and Irish Universities Association (IUA) that made my Ph.D. work possible. Additional travel funding was provided by ESRI for attending the conference at AGILE 2011 in The Netherlands.

I am deeply indebted to all the colleagues in our research group: Ricky Jacob, Bashir Shalaik, Blazej Ciepluch, Pouria Amirian, Anahid Basiri.

I also would like to thank all the people in Department of Computer Science for helping me with technical problems and making helpful comments through the study. Thanks for their enlightening discussions and friendly cooperation.

I would like to thanks my friends: Hao Wu, Xiaomin Chen, Binbin Lu, Wenbai Yang, SiLiang Tang, Yanpeng Cao, Jian Wang, and Jianghua Zheng. Thanks for your understanding and support for my work.

Last but most definitely not least I wish to give my sincerest thanks to my parents who have shown great understanding and patience towards this work. Thank you!

Publications from this research

During the course of this PhD, the research work and ongoing results have been published and presented at a number of peer-reviewed international conferences.

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **Selective progressive transmission of vector data**. International Conference of Geocomputation, UCL, London, 2011.

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **A model for progressive transmission of spatial data based on shape complexity**. ACM SIGSPATIAL GIS 2010. The SIGSPATIAL Special Volume 2, Number 3, November 2010.Note: Awarded Best runner up poster presentation at the ACM SIGSPATIAL GIS 2010 conference.

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **Using Java XML Tool to process OSM XML data**. At OpenStreetMap State of the Map 2010, Girona, Spain, July 2010.

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **A shape complexity approach to simplification of geospatial data for use in Location-based Services**. Proceedings of the 7th International Symposium on LBS and TeleCartography ,Sept. 2010, Guangzhou, China.

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **Polygon processing on openstreetmap xml data**. Proceedings of the GIS Research UK 18th Annual Conference (London, England, 2010), M. Haklay, J. Morely, and H. Rahemtulla, Eds., University College London, pp. 149-154.

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **How little is enough? Evaluation of user satisfaction with maps generated by a progressive transmission scheme for geospatial data**: In proceedings of the 14th AGILE International Conference on Geographic Information Science, Utrecht, The Netherlands, April 18-21, 2011. Fangli Ying was a recepient of an ESRI Scholarship for AGILE 2011

- Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. **Dynamic visualization of geospatial data on small screen mobile devices.** ADVANCES IN LOCATION-BASED SERVICES Lecture Notes in Geoinformation and Cartography, 2012, Springer-Verlag Berlin Heidelberg pages 77 -87.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Using maps on mobile devices

Significant development and changes are taking place in mobile technologies. The laptops and personal computers of the last two decades have evolved into all-in-one smartphones. Since wireless internet and mobile devices are now ubiquitous, citizens are no longer constrained to use computing devices and services in a single specific location. They are flexible to conduct their daily business and activities with mobile applications from anywhere at anytime (i.e. social network, tourist guide and road navigation). It has become increasingly popular to seek information related to spatial characteristics. Many mobile applications have been developed to deliver these types of location information using digital maps (e.g. Google Maps [Google, 2012] and Mapquest [Mapquest, 2012]). The combination of these map-based services with mobile applications are called *Location- based Services* (LBS) [Steiniger et al., 2006]. In recent years LBS have obtained great success in the global marketplace mainly because they provide citizens with a convenient way to access location-based information using their mobile devices.

Raster-based maps are currently the most commonly-used types of maps in current LBS. These maps are pre-rendered on the server side and are stored explicitly as digital image files in multiple scales with different resolutions. This divides a map into small map images called "tiles". When a user requests a map in an LBS these tiles are delivered to the user (as shown in the right hand side

of Figure 1.1a). When mobile users retrieve spatial information, the mobile application will download a sequence of maps tiles corresponding to the scale and resolution specified by the user. While this transmission method is efficient for LBS mapping services, these pre-rendered maps do not always meet the requirements of all users of these services. Pre-rendered raster-based maps are provided for a specific set of spatial scales and with a fixed cartographical style regardless of the purposes for which the LBS is being used for. This is shown in the left hand side of Figure 1.1a. As the popularity of LBS and mobile applications continues to rise, new challenges and opportunities have arisen related to the techniques and methodologies used to provide digital mapping.

1.2 Current problems with mapping in LBS

The recent development of mobile technologies has also sparked a dramatic increase in the volume of geographic information being collected and being made publicly available. This has seen the new phenomenon of volunteers forming as "crowds" using the Internet and social media to "mash up" geographic information from various forms (e.g. GPS tracks, geocoded photographs, tracing features from aerial imagery, social media activities) [Batty et al., 2010]. Up to a few years ago, the only sources of spatial data and digital mapping products were government agencies or commercial companies. Today this has completely changed. These crowds are collecting spatial data and information themselves in unprecedented volume and using social media and web technologies to manage, edit, and distribute this data [Batty et al., 2010; De Longueville et al., 2010]. This "crowdsourcing" approach has rapidly gained popularity and some notable examples exist. The built and natural environment of a rural or urban landscape may change dramatically, especially during natural hazards (e.g. Haiti and Sendai Earthquakes). *Volunteer geographic information* (VGI) communities have reacted spontaneously to these situations and generated vast amounts of crowdsourced geographic datasets in a very short time [Goodchild, 2007; Goodchild and Glennon, 2010; Pultar et al., 2009]. For example, *OpenStreetMap* (OSM) [OpenStreetMap] indicated that their volunteered mapping project in Haiti provided the geospatial data for emergency planning for island inhabitants during the dev-

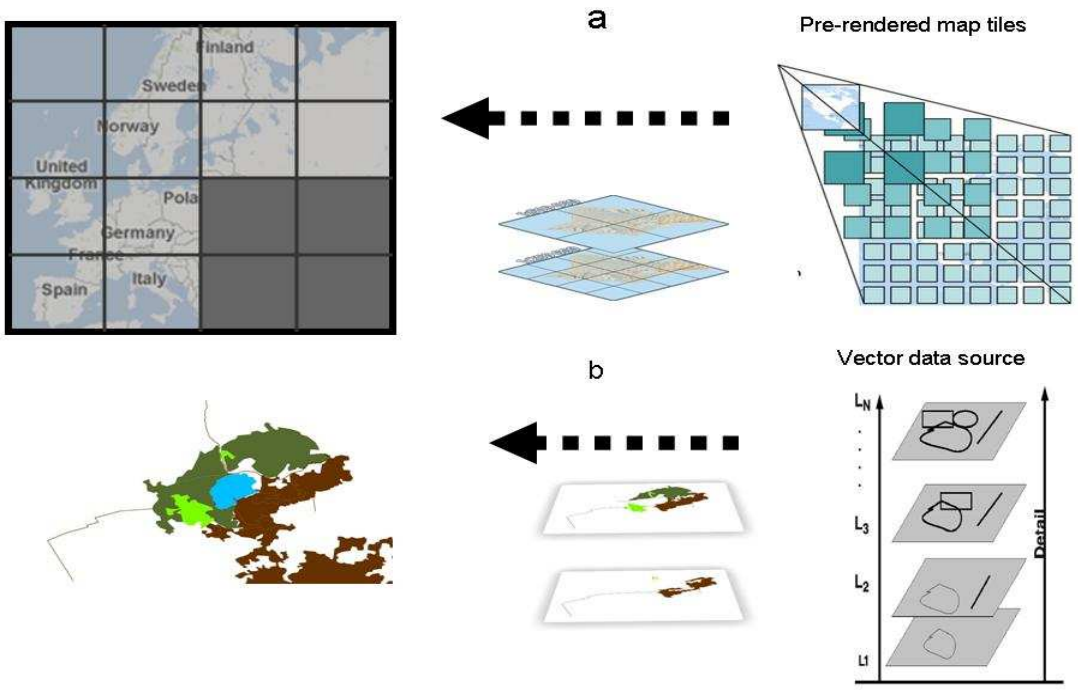

Figure 1.1: In this flowchart, we see in Figure (a) the traditional raster-based map retrieval technique. In Figure (b) we see a dynamic vector data map visualisation where pre-rendered map tile images are not used

astating earthquake of January 2010 [Heipke, 2010]. A detailed city map of the capital city was created by the volunteers for OSM within a few days. Crucially this map contained all of the most up-to-date spatial data. Traditional methods of spatial data collection and management and subsequent map production would not have been able to react or respond so rapidly. Whilst examples such as OSM in Haiti are very impressive, the explosion of user generated spatial content in recent years has introduced many problems related to managing and processing these large dynamic datasets for map services. Most traditional tile-based map services are challenged to provide up-to-date spatial information efficiently, since they must frequently update their spatial databases and generate timely digital map tiles. Depending on the size of these databases, these updating processes could take hours or even days. In summary, mapping applications using pre-rendered maps tiles may fail to capture and deliver timely spatial information for time crucial LBS tasks such as emergency or humanitarian assistance during environmental disasters. Currently, there is a lack of effective techniques to handle the management and visualisation of dynamic spatial data (such as VGI data) for use in LBS applications.

Using maps in a mobile context also presents some different challenges. Visualizing maps on mobile devices is fundamentally different to the standard office environment where one visualizes maps in desktop machines. For example, a tourist is walking in a city centre and trying to find a Chinese restaurant. He/She may use a mobile application whilst on the move and query to find spatial information based on his/her current location. In this situation, it may be difficult for mobile users to focus their attention on the small screen devices due to multi-tasking (e.g. walking, avoiding other pedestrians and reading the map). Nevertheless, they still expect to access the appropriate spatial data quickly [Burigat and Chittaro, 2005] and potentially perform some related queries on this data. For example, selecting all the restaurants on a street displayed on the map and retrieving corresponding customer reviews. However, it is difficult to provide such interactions if the mapping data is only presented as flat raster images [Reichenbacher, 2004].

On the other hand, vector-based mapping approaches have many distinct advantages in visualizing maps in LBS. These include flexible rendering style, more functionality for interaction and faster delivery of timely spatial data [Reichen-

bacher, 2004]. Vector data provides enormous potential for visualizing spatial data. Examples of formats include: Scalable Vector Graphics (SVG) formats [Neumann et al., 2003], XML formats [Lehto and Sarjakoski, 2005] and recently HTML5 format s[Boulos et al., 2010]). When a user of an LBS requests a map of a region, instead of sending the pre-rendered map tiles, the vector data itself is transmitted to his/her mobile device where it is then rendered in a mobile client application (1.1b).

However, efficiently transmitting large amounts of vector data over the Internet is still a challenging problem [Antoniou et al., 2009; Buttenfield, 2002]. There have been many notable efforts made to overcome the limitations of vector data transmission for visualization purposes. Map generalization approaches can extract map representations with significant data reduction from the original datasets [Agrawala and Stolte, 2001; Buttenfield and McMaster, 1991; Lehto and Sarjakoski, 2005; McMaster and Shea, 1992; Mustafa et al., 2006; Neun and Burghardt, 2009; Oosterom, 1993]. Progressive transmission approaches have been applied to this problem to exchange spatial data over the internet [Augusto et al., 2009; Bertolotto and Egenhofer, 2001; Buttenfield, 2002; Follin et al., 2005a; Haunert et al., 2009; Yang, 2005]. These approaches send data in incremental *levels of detail*(LoD) so that users can efficiently access the data. Small packages of data are transmitted with possibly the most relevant details transmitted first [Bertolotto, 2007]. Despite of the success of progressive transmission approaches, their main limitation is that they have mainly focused on the desktop computer as the visualisation medium. Very few of these approaches can be applied directly to visualisation on mobile devices because of their constraints imposed by the mobile device itself. The intrinsic hardware limitations of mobile devices, as opposed to desktop computers, makes effective delivery of vector-based spatial data a challenging problem. Nevertheless the approaches mentioned above can be extremely beneficial for developing adapted approaches for use specifically in the mobile context [Bertolotto, 2007; Bertolotto and McArdle, 2011]. These adapted approaches will require modifications which must take account of the following constraints:

- *Small screen size:* The small screen display size on the mobile device is highly restrictive for visualization of large-scale spatial data. Users only

can view the map at limited resolution levels.

- *Computing resources:* The limited computation power on mobile devices restricts the type of computation that can be performed for visualization. The transfer of data across the telecommunications network will also consume computational resources on the mobile device.

- *Networking issues:* The wireless communication network can suffer from problems of low bandwidth or even interruption. This is often beyond the control of the users. With low bandwidth connections mobile users must often wait a long time to receive a full dataset of spatial data before visualization can commence.

While advances in technology may help to overcome them, the development of mobile applications for LBS must still take these constraints into account in order to provide effective visualizations for the mobile user. With these constraints in mind we now summarize the key research questions in this thesis as follows:

1. If the data source is dynamic (such as a VGI data source), can one effectively use this source of vector data in LBS applications?

2. What are the key constraints of mobile devices which must be considered in the design of spatial data management strategy for mobile client applications?

3. How can progressive transmission strategies be improved to provide smooth and efficient visualization of spatial data in LBS mobile applications?

1.3 A data management strategy for the dynamic visualisation of maps on mobile devices

With the emergence of LBS applications, many technological issues have arisen when designing software solutions for the visualization of maps on mobile devices

as we discussed in Section 1.2. Efficient management of vector-based spatial data is still a challenging problem on mobile devices. In this research, in order to address the key challenges of visualization of dynamic spatial data on mobile devices for LBS, we have proposed a framework to facilitate the management of vector-based spatial data both in the server and mobile client environment. This framework is illustrated in the flowchart in Figure 1.2. OpenStreetMap (OSM) data, as a typical source of dynamic spatial data, is used as the case study in this work. On the server side (corresponding to the left-hand side of Figure 1.2), we researched and developed the following approaches to enhance data management for dynamic data sources and deliver efficient transmission of that data:

- A data processing software application for processing spatial data in vector format. The software can handle vector data extraction directly from a database or spatial data, which is being streamed "live" from a dynamic data source (e.g. OSM databases) [Ying et al., 2010b]. Data simplification techniques are implemented in the software tool to reduce the amount of data in the processed datasets in real time.

- A low redundancy data structure for storing and processing the spatial data. This data structure is used by the proposed progressive transmission scheme to organise the spatial data into increasing LoD (LoD_0, LoD_1, ..., LoD_n). This data structure enables instant access to the coarsest version of the spatial datasets (LoD_0) from mobile applications. The subsequent LoDs are then provided progressively with higher resolution until the final full resolution LoD LoD_n.

- A data selection strategy to enhance progressive transmission based on a proposed heuristic model. The criteria of this model is based on the results of shape analysis experiments carried out as user trials [Ying et al., 2010a, 2011]. A view-dependent progressive transmission technique [Corcoran et al., 2011a] is applied to deliver the suitable LoDs based on the results of the criteria.

On the client side (corresponding to the right-hand side of Figure 1.2), a series of data management strategies have been developed by carefully considering

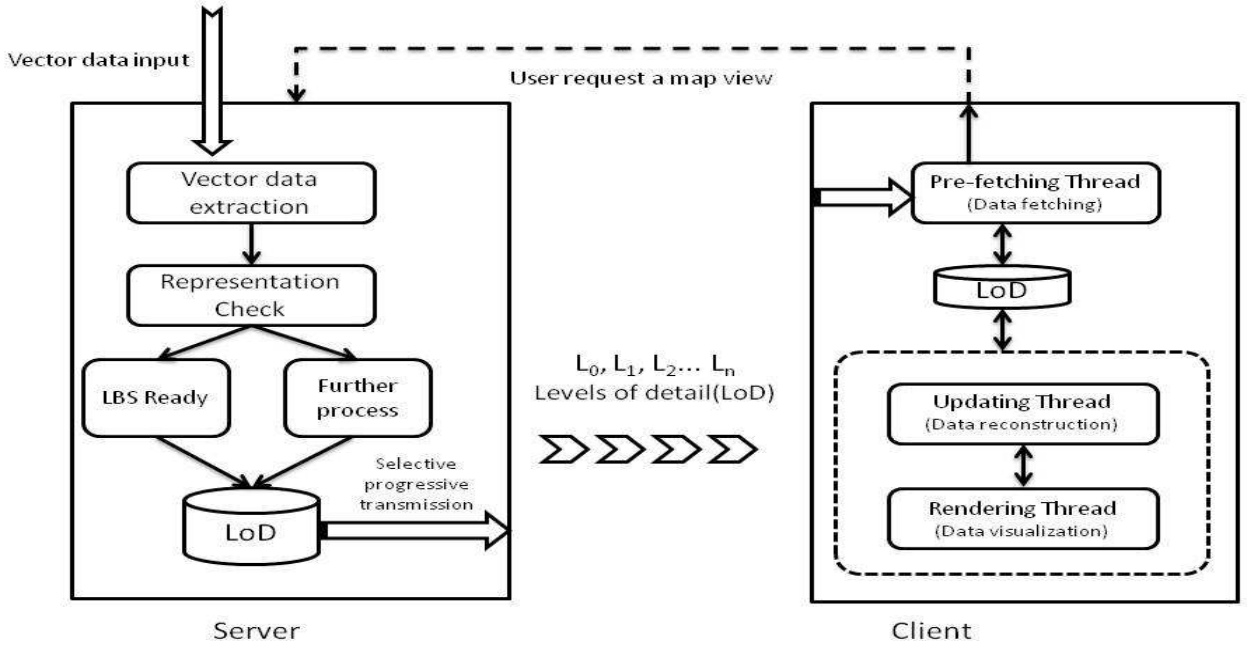

Figure 1.2: This workflow diagram shows the linkages between all of the individual components in the proposed framework for spatial data management and visualisation on mobile devices.

the limitations inherent in mobile devices for the task of dynamic spatial data visualisation. We summarised these data management strategies as follows:

- A multi-threaded data management scheme has been designed in the mobile client. It can schedule data downloading and data rendering in parallel in order to speed up the data visualization.
- A set of algorithms have been developed corresponding to the threads in the multi-threaded approach. These algorithms correspond to: data visualization, data updating, and fetching of the next view. This algorithmic approach is used in conjunction with a view-dependent progressive transmission scheme in the client in order to decrease the amount of data transmitted for irrelevant areas of the map visualisation while also avoiding the overloading of the phone memory and disk space with spatial data.
- A user-motion capture method has been implemented to predict the screen position where the user is most likely to move to on his/her next move action on the map visualisation on screen. The predicted position is then used as input to the algorithm for caching the spatial data that the user will require next. This assists in overcoming the performance issues related to telecommunication network latency.
- A flexible rendering approach is implemented to visualize the map dynamically. The user can specify their preference in terms of map styling (for example colour schemes).

This framework provides a comprehensive treatment to the issues of spatial data management for the mobile client devices, including data processing, data transmission and data visualisation. We outline the results of a number of experiments which practically demonstrate its advantages (overall processing time, data reduction and transmission, and smoothness of transmission and visualisation). This work demonstrates the practical potential of the proposed framework that can be realised through implementation in a real-world LBS application.

1.4 Research contributions

The overall intent and objective of this thesis was the proposal of a framework for efficient management of the vector-based spatial data for visualization on mobile devices. A broader objective was to develop this framework whilst working with the constrained computing environment of mobile devices. This framework is flexible to be applied to both static data sources (e.g. pre-setup data on a local server) and dynamic data sources (e.g. streamed OSM data). Many existing approaches in the literature can only manage static data, which has been pre-computed [Bertolotto and Egenhofer, 2001; Cecconi, 2003]. Such approaches cannot easily adapt to a dynamically updating spatial data source.

In this framework, the server application can process the requested spatial data on the fly. For high resolution (potentially over-represented) vector data, this framework performs data simplification processing to reduce the data volume whilst retaining the shape characteristics. Simplified initial datasets with a suitable data representation quality are generated for immediate visualisation on the client device. The incremental *LoD* data is then handled in a low redundancy data structure for progressive transmission upon the user's request [Follin et al., 2005a]. The screen metric transformation is applied to transform the spatial data for visualization on the small screen of mobile devices in required scale. Topological consistency issues are examined in relation to the OSM datasets used in the case-study.

The principle of progressive transmission is used in this framework for transmission of vector spatial data. Based on a series of user trials and shape analysis theory, a cost-benefit constrained model has been developed to improve the efficiency of progressive transmission of large numbers of LoD refinements. A view-dependent approach is integrated to select the LoD data within the user required view for delivery [Corcoran et al., 2011a]. This cost-benefit constrained model made the trade off of level-of-detail against visual importance of the shapes' features in the map visualisation. The novelty of this optimization process is that this framework can maximise the use of potentially limited bandwidth to deliver the relevant LoD for map visualisations in a smooth manner.

An automated data management strategy has been developed to enable low-

capacity mobile clients (i.e. poor network connectivity, limited computational resources, and low data storage budget) to visualize the spatial data effectively. A number of algorithms have been developed to support a multi-threaded data management strategy. This strategy can speed up visualization and prevent overloading data on the memory by discarding the irrelevant LoDs. The novelty of our approach is that it uses the available computational resources more efficiently whilst significantly reducing the overhead of memory usage for spatial data storage on the device. Following on from this we successfully implemented a data pre-fetching approach accompanied with real-time styling visualization of the spatial information based on the user's preference of visualisation scheme.

Overall we have identified the key challenges of managing spatial data in mobile visualization applications. Despite the advances in mobile technologies in the past number of years, this is still a non-trivial task. The experimental results and the user trials demonstrated that the proposed framework can improve the spatial data management on mobile devices whilst carefully understanding the constraints of mobile devices. Several peer-reviewed conference papers and book chapter articles have been published outlining the progress and results of this research. They will be discussed and referenced, as appropriate, throughout the proceeding chapters. This thesis work provides a comprehensive treatment of the management of spatial data for visualization on mobile devices with useful contributions to the current knowledge in this area. The work also provides several opportunities for the further research work in the immediate and long-term future.

1.5 Thesis organization

The remainder of the thesis is organised as follows:

Chapter 2 introduces the technical background and the related research work. We discuss the issues of using spatial data in LBS and identify the key challenges of handling spatial data for visualization on mobile devices. Then, we review a number of current techniques related to these issues from the research fields of: computer graphics, map generalization and progressive transmission.

Chapter 3 presents a model for processing the large quantities of vector-based

spatial data on the application server. We describe an identification process for the representation quality of spatial objects in a dataset and perform generalization processing to reduce the amount of spatial data needed to represent these objects. The experiments demonstrates that this approach preserves the overall representation quality of the map representations in multi-resolution contexts. The problem of small screen visualisation is dealt with by applying a screen transformation metric.

Chapter 4 describes a software tool implementing the model from Chapter 3. In this research chapter, a software tool has been implemented to process OSM data based on opensource tool. We demonstrate the advantages of this approach with a comparison with the traditional mapping services. It has shown an improved data access to dynamic data sources.

Chapter 5 designs and develops a framework for progressive transmission for mobile clients. We use the principles of progressive transmission to organise data into increasing LoD for transmission and subsequent visualisation. A series of user trials are carried out to assess the usability of the generated map visualisations. The results of these trials demonstrate the strength of progressive transmission. For certain datasets, users do not perceive dissimilarities between the generalized maps and the final delivered full resolution maps. We introduce a heuristic-based approach with view-dependent techniques to arrange the delivery of the sequence of LoD. We also describe some results which demonstrate that this approach provides smoother transitions between sequential LoD than static approaches.

Chapter 6 presents and discusses the strategy of monitoring the spatial data for visualization in the mobile clients. The design of this strategy considered the limitations of network connectivity and data storage budget in the mobile devices. A view-dependent progressive transmission approach is implemented to manage the data delivery corresponding to the user's current map view on-screen. A multi-threaded data management approach is developed to deliver and visualise the LoD efficiently in the mobile client. A predictive strategy is developed to pre-fetch the data for the next user view within the rendering time of the current view. The experimental results demonstrate the advances of this strategy in managing the spatial data in the mobile client application particularly during user interaction.

Chapter 7 provides a summary of the work of this PhD research. We retrospectively evaluate the achievements in this thesis against the aims and objectives in Chapter 1. In the evaluation, we discuss some improvements to this work in immediate future work. The thesis closes with a discussion of the long-term vision and future direction for spatial data management and visualisation on mobile devices.

Chapter 2

Literature Overview and Discussion

Location-based Services (LBS) are now emerging in a wide range of application domains. It is crucial that these services access spatial data with appropriate quality [Raper et al., 2007]. In practice, a data resource for an LBS could be updating, at irregular intervals, extremely quickly (e.g. crowd sourcing datasets). On the other hand, the data resource could be a relatively static dataset subject to infrequent changes. The volume of spatial data has increased dramatically in recent years. Traditional raster-based mapping applications are not flexible for use in the provision of up-to-date spatial data. This is due to the potentially long times required by tile generation processes. Vector-based mapping solutions are emerging mainly because of their potential for higher quality visualisation and increased user interaction possibilities. Nevertheless, vector-based mapping applications still suffer from drawbacks that have prevented their wider implementation and adoption. Considerable research has been carried out in related areas in recent years to address these issues related to transmission and visualization of vector-based spatial data over the Web [Antoniou et al., 2009; Buttenfield, 2002; Corcoran et al., 2011a; Yang, 2005; Yunjin and Ershun, 2011]. However, the ubiquitous nature of mobile devices means that challenges remain in the management of spatial data for visualization.

In this chapter, we begin with a discussion of the characteristics of spatial

data in LBS. This discussion has an emphasis on VGI data in Section 2.1. We then identify the various issues of handling spatial data in LBS applications in Section 2.2. Finally, we close this chapter with Section 2.3, which includes a review of current solutions developed in related areas with emphasis on work from computer graphics, map generalization and progressive transmission.

2.1 The use of spatial data in Location-based Services

There has been an explosion of technologies allowing communication with global computing and information infrastructures that connect billions of wireless mobile devices. These technologies have also enabled mobile users to query the environment around them via LBS applications whilst the user is on the move. These applications have a strong spatial component (i.e. location, proximity, distance) and are usually associated with digital maps. LBS provide spatial information based on the user's current location or the locations which they intend to move to at some later time [Krumm and Shafer, 2005]. LBS applications aim to make the data accessible for mobile users *from anywhere at any time* [Ilarri et al., 2010]. This ubiquitous computing paradigm brings great convenience in terms of information and data access. There are many popular LBS examples prevalent in human daily activities, including: finding directions to a business or attraction, obtaining news or information about the current location, making announcements to contacts in one's social networks, taking geo-located photographs or video and among others.

The work of Raper et al. [2007] illustrates the enormous diversity of LBS appearing and the wide range of application sectors that are represented. Along with increasing usage of LBS in multi-disciplinary fields, these applications involve the management of vast quantities of data. There are many different sources of geospatial data, many of which have fundamentally different characteristics and consequently impose a wide spectrum of requirements on the underlying LBS applications. Steiniger et al. [2006] created an inverted pyramid for illustrating

geospatial data usage in LBS. This is shown in Figure 2.1. General geospatial data (e.g. weather or traffic reports) are the most commonly used geospatial data and services in LBS. Meanwhile, there are some geospatial datasets which have been collected specifically for LBS purposes. In the bottom of the pyramid, there are these datasets, which could be collected for a very specific purpose such as building data for indoor navigation (i.e. finding a conference room).

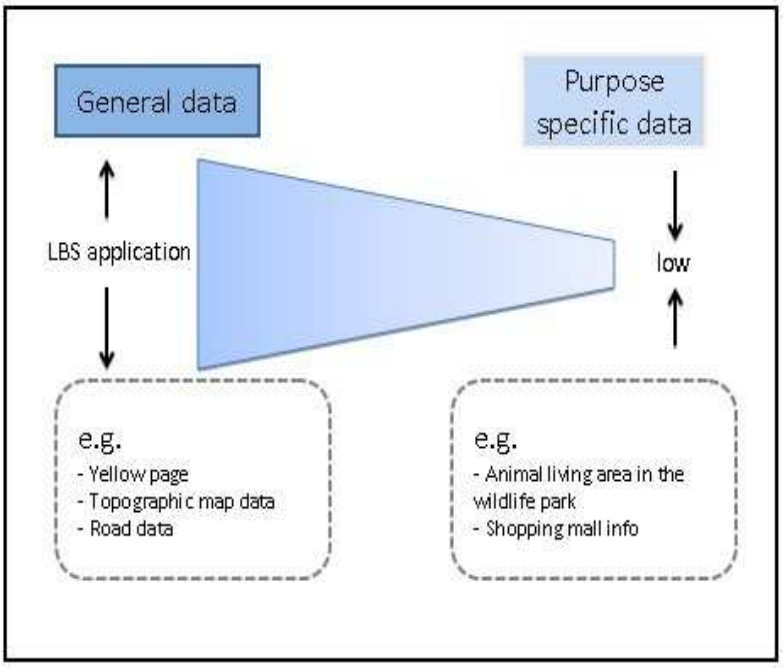

Figure 2.1: The Inverted Pyramid for geospatial data usage in LBS by Steiniger et al. [2006]

Tryfona and Pfoser [2005] distinguished data in LBS according to its semantics and intended use. The data consists of 3 principal components. These are outlined as follows:

1. **Domain Data**: including spatial and temporal attributes (i.e. location coordinates, spatial objects).

2. **Content Data**: describing LBS related contents (i.e. tourism attractions).

3. **Application Data**: consisting of the user profile and the services provided by the LBS (i.e. user preference data).

Mooney et al. [2010] catalogues geospatial data according to broader requirements for LBS. They state that data for Points of Interest Queries (POI), routing and navigational assistance, and other related services are the most popular. There are many more classifications from different points of view, but they overlap in many ways and often involve many of the same spatial query operations.

Recent advances in geoposition-enabled mobile devices and the emergence of the geoweb have now empowered citizens to collect geographic information or, at a minimum, annotate their content with geographic location information. This information is stored in online databases. Some of these online spatial databases not only provide derived maps for use in LBS, but also enable users to contribute spatial data using LBS applications. The new forms of crowd-sourced spatial data are being created through an increasing number of LBS activities. These activities include check-ins with location geotags using Foursquare [FourSquare, 2012] or using GPS-enabled mobile phones which share the locations and movements with friends on Facebook [Facebook, 2012]. Motivating individuals to act voluntarily is far cheaper than any alternative, and its products are almost invariably available to all [Goodchild, 2007]. Moreover, these contributions are characterized by locality (users contribute local knowledge) and by a very wide scope of content (pictures, restaurant reviews, jogging tracks, citizen gathered environmental measurements, and so on). As Elwood [2008]) mentioned, the local knowledge of these crowds is a very important factor in retrieving geographic information and in documenting and mapping both spatial and non-spatial elements of a place. The abundant spatial data from local contributors has the potential to be a very useful data source for location based services [Goodchild, 2009].

This growing phenomena of citizens collecting geographic information has now reached a critical mass and is deserving of a research focus as a fully fledged field of inquiry within GIScience [Elwood, 2008; Weiner et al., 2002]. The research term for this phenomena was "Volunteered Geographic Information"(VGI) and was coined by Goodchild [2007] and Sui [2008]. VGI technologies and practices are dramatically altering the contexts of geospatial data creation and sharing. The proliferation of VGI has enabled many Internet users to contribute to the

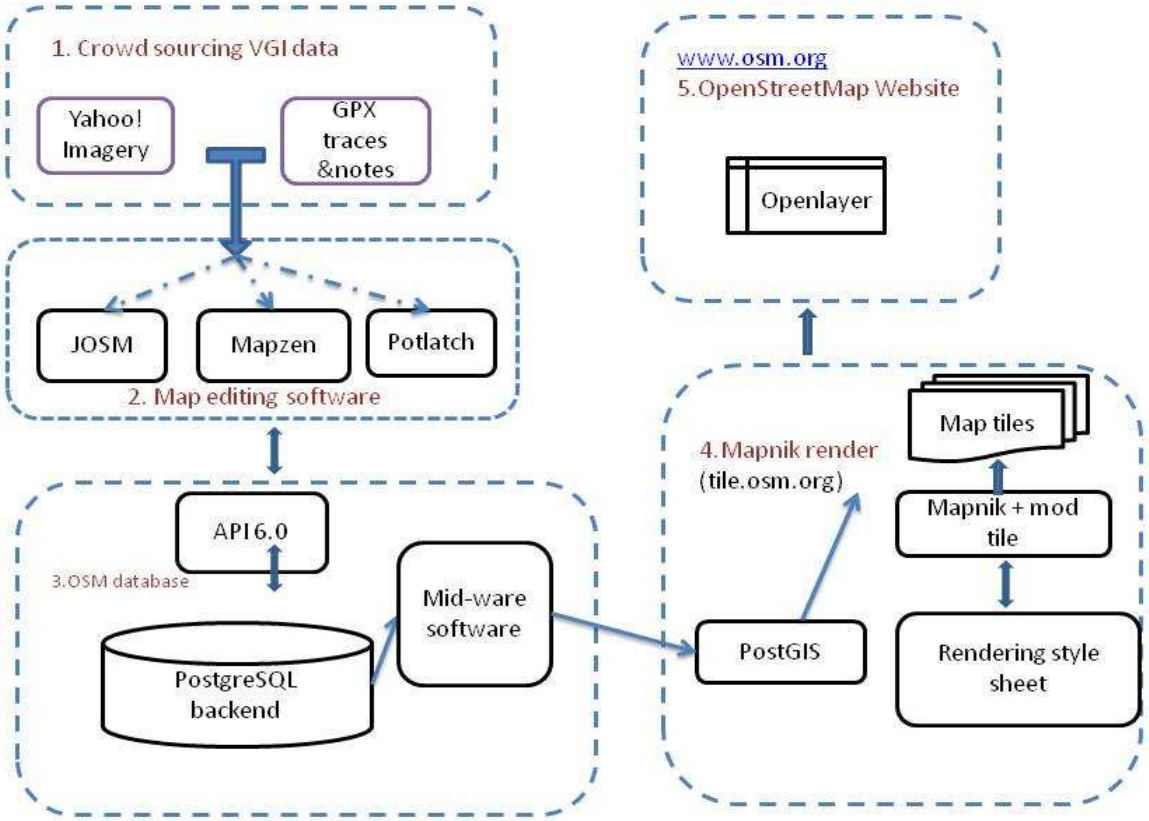

Figure 2.2: This figure shows the almost complete range of components and processes used for the collection, storage, manipulation, query, display, and visualisation of OpenStreetMap data. This flowchart is taken from the OpenStreetMap Wiki

construction of rich and increasingly complex spatial datasets. For example, as one of the most successful VGI projects, OpenStreetMap(OSM) is a collaborative project striving to create a free editable spatial database of the whole world [Mooney and Corcoran, 2012b; Neis et al., 2011]. OSM contributors from all around the world are encouraged to collect and collaboratively contribute geographic information to the OSM online spatial databases. More recently, OSM has begun to be used in commercial products (such as Nestoria, Mapquest, Foursquare). OSM data and maps are now widely used in many domains of LBS. OSM provides a new dimension to allow users to participate in accessing, editing, and sharing up-to-date spatial data under a free and open data license. We use OSM as it is currently the best example illustrating the process of how crowds of citizens can contribute VGI data to an online database and how a successful VGI project can share this data for use by LBS applications. A general logical framework is depicted in Figure 2.2. In the left hand side of the figure, the contributors are allowed to upload their VGI data in different formats to the online database. The formats for crowd sourced VGI data are varied. Component 1, as seen in the Figure 2.2, includes GPX lines from GPS devices and image data [Turner, 2006]. There are also numerous editor software packages (component 2) available such as JOSM (Java for OSM) [JOSM, 2012]. These packages allow the preparation of new data for OSM and the editing of current OSM data from desktop computers. The user of these packages can then transmit this data to the OSM database (component 3) through an API (Application programming interface).

As mentioned above, many commercial map services are now using OSM data as a source to provide spatial information for LBS applications, as shown in the right hand side of this figure. In a traditional tile-based map application environment, VGI data would be rendered in advance (offline) using a fixed style before use in LBS (such as the Mapnik style [Mapnik, 2012])(component 4 in Figure 2.2). When users perform a spatial query via these LBS applications, the map service will automatically deliver a sequence of corresponding map tiles associated with spatial information. This pre-rendering process may be time-consuming. Moreover, it is not flexible for delivery of up-to-date spatial information from frequently updating VGI databases. As VGI data is an emerging trend in recent years, the discussion will be heavily influenced by VGI data, there are still practical prob-

lems restricting the use of vector-based spatial data in LBS applications. Many of these issues arise in handling vector-based spatial data in the map services. We will now discuss these issues and challenges in the next section.

2.2 Spatial data handling issues for LBS

The increasing use of LBS results in a range of issues of data management. As stated by Jiang and Yao [2006], "Geospatial data are one of the key components of LBS, as in essence LBS are a kind of data or information services". LBS provides and delivers information or data to its users in a highly selective manner, by taking the user's past, present, or future location and other contextual information into account. However, Tryfona and Pfoser [2005] argue that "data involved in LBS have not been really examined in depth". Mooney et al. [2010] also argue that if geospatial data is not available which are fit for the purpose of the intended LBS then it is unlikely that the LBS will be a success among users. The constraints of mobile environments, the spatial property of location-dependent data, and the movement patterns of mobile users also pose great challenges for the data provision of location-based services to mobile users.

From the point view of mobile users, LBS applications must provide up-to-date spatial data since location-based information often change frequently (e.g. newly established bus stops, traffic diversions, shop relocation, etc). The maps in LBS must be presented efficiently and accurately. This becomes crucially important when these maps are being used for personal navigation. When users are using LBS applications, they usually need to interact with spatial objects on the map in order to query the related information (e.g. click a shop building on the map, which represented as a polygonal object on the map, and pop up a customer review about the shop).

Considering the dynamic characteristics of VGI data, there may be a requirement to adapt some methods of data management to process VGI data for LBS. Moreover, there are some real-world examples already partially implemented using VGI like MapQuest. Mooney et al. [2011] argues that the VGI data and information used by LBS for applications in urban environments must exhibit the following characteristics:

- be up-to-date and timely,
- be spatially accurate to such an extent that it is useful for most non-mission critical type applications,
- be of high quality in terms of other characteristics such as semantics and associated metadata.

Thus, several issues arising when using VGI in LBS mobile applications will be discussed in the next subsections. We will take OSM as a case study to illustrate these issues as follows:

- The large volumes of VGI data (section 2.2.1).
- Problems associated with the quality and precision of VGI data (section 2.2.2).
- The dynamic, real-time characteristics of VGI data. (section 2.2.3).
- The formats that VGI data can be transmitted in (section 2.2.4).
- Considerations of the mobile context within which the VGI data is being used (section 2.2.5).

2.2.1 Large volume of spatial data

The availability of large amounts of spatial data has significantly increased. This is essentially due to both the increasing number of different devices collecting such data (i.e. remote sensing systems, environmental monitoring devices and, in general, all devices linked to location-aware technologies) and the development of distributed computing infrastructures (e.g. Web 2.0 [Anderson et al., 2007]) as platforms to share and access any location-based information. This is significantly different with centralized systems [Hudson-Smith et al., 2009]. In ubiquitous computing environments, the online database acts as a central service or can be built from the bottom up in a decentralised fashion. The data is produced mostly by the user for the user. Some photo-sharing applications, such as Flickr [Flickr, 2012], allow users to publish and share photos with spatial information embedded, namely "geo-tags". These Photo-sharing Web applications emerged around

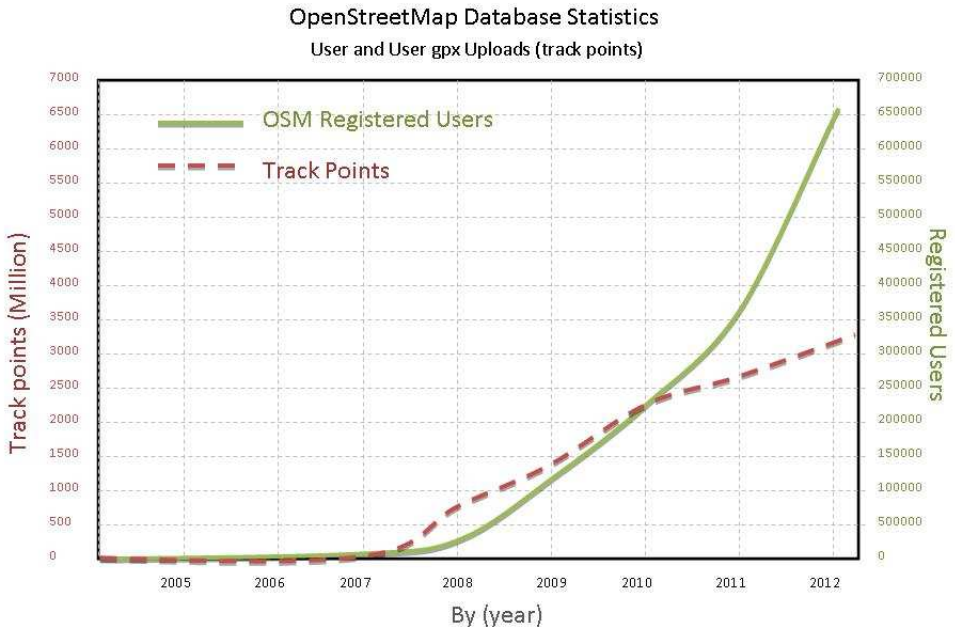

Figure 2.3: The increase in OSM data from 2005 until 2012

2004. Their social impact along with the phenomenon of user generated contents and the increased presence of VGI data have motivated researchers to consider them as a source of geographic information. Currently, the Flickr Website hosts more than 4 billion photos [Yahoo!2009, 2009] where more than 300 million of these are geo-tagged. The rise in contributions to OpenStreetMap is shown in Figure 2.3 [Stats, 2012]. There are millions of crowd-sourced VGI data items collected and contributed to OpenStreetMap's online database. The volume of data has increased by many orders of magnitude over the last few years. However, internet delivery of large scale data is often problematic due to the large size of such files, combined with a general lack of Internet bandwidth of mobile devices. Moreover, as these services grow in breadth and depth, the complex issues of accessing a sometimes messy landscape of location data will present researchers with more attractive challenges [Hudson-Smith et al., 2009]. However, there are still no particular infrastructures for the mobile devices to access these large spatial databases [Carboni et al., 2002].

2.2.2 Data quality issues

Geographical data usually corresponds to datasets collected and integrated from different sources (private or public institutions), produced by different processes (e.g. social, ecological, economical) over a geographical area, at different times (e.g. every 10 years), possibly using different devices. Limitations are imposed on the overall geographic data quality at all stages of the data lifecycle: from capture, through to input, manipulation, spatial analysis, to the final presentation of results [Worboys, 1998]. Given that the majority of VGI is collected by individuals, who may not have experience in professional collection and management of spatial data, it is inevitable that data quality issues will arise. The quality of data obtained using approaches such as the crowd-sourcing model in VGI can vary spatially and temporally. This is mainly due to differences in skills, equipment, and information technology tools used by those who contribute data [Flanagin and Metzger, 2007]. Additionally, the data capture equipment often has inherent limitations and this can lead to erroneous readings. Goodchild [1993] states that data quality is crucial and that each data entity should carry informa-

tion describing its quality with each operation on the data having an associated error tracking procedure. Goodchild stresses that systems should contain quality control mechanisms as standard. Worboys [1998] also states "without a careful treatment of spatial resolution and consequent imprecise data representations, it is difficult to approach the correct scale and generalisation".

It is necessary to know the representation quality of the spatial data in order to be able to use it effectively. Haklay and Weber [2008] and Kounadi [2009] performed comparisons of VGI data with authoritative datasets and stated that VGI data is "fairly accurate". Similarly, Coleman et al. [2009] explain that as there is a considerably wide range of motivations for users participating in the creation and sharing of spatial content on the Web. As a result, the quality of these data contributions can vary significantly. According to these authors, the understanding of the participants' motivation and experience can give valuable insight into the resulting content quality. However, almost all of the information that we possess about the real world is neither complete, nor precise. One of the most common representation problems of spatial data is regarding the number of nodes used to represent a polygon or polyline feature in a dataset. For example, in OSM database, some lines and polygons are over-represented (too many nodes) while others are under-represented (too few nodes given the complexity of the real-world feature the polygon represents). For example, the two polygons on OSM maps have been shown in Figure 2.4a and Figure 2.5a. The polygon in Figure 2.4a is of relatively high resolution and the shape in question in Figure 2.4b is over-represented with a large number of nodes in its data structure. On the other hand, the polygon in Figure 2.5b contains a lower level of detail and potentially the shape in question is poorly represented by the polygon in Figure 2.5a. This under-represented polygon could be selected as a candidate for immediate transmission. This polygon could not undergo any further reduction in the number of nodes used to represent it. However, delivery of all of the data in Figure 2.4a may not be necessary as this polygon could potentially undergo some form of generalisation and then be transmitted to the user device represented by a reduced number of nodes. For these over-represented spatial datasets, we feel that simplification can take place before this data is sent to a mobile device for visualization.

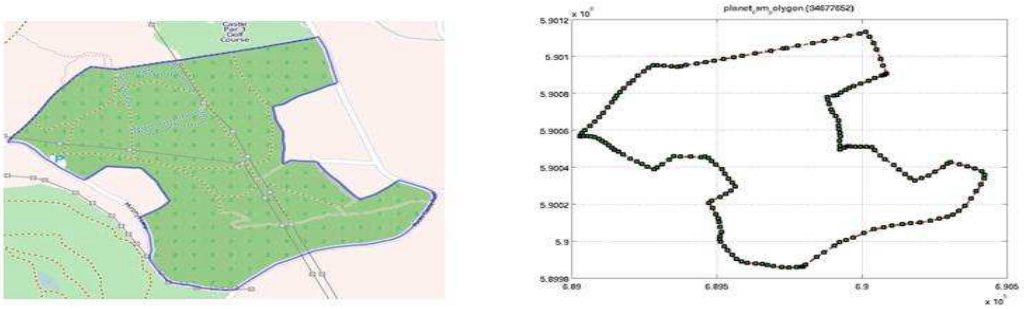

Figure 2.4: a. This is the original raster representation of the polygon on an OpenStreetMap tile layer. b. This is an example of a polygon which is over represented in terms of spatial detail with 647 nodes. The $\bar{K}$ is small being 0.01. More details of $\bar{K}$ are given in section 3.2

Figure 2.5: a. This is the original raster representation of the polygon on an OpenStreetMap tile layer. b. This is an example of a polygon which is under-represented. This means that there are too few nodes in the polygon to describe its structure and spatial detail adequately. The polygon has 51 points of data. The $\bar{K}$ is also relatively high at 0.486. More details of $\bar{K}$ are given in section 3.2

2.2.3 The dynamic nature of updates to spatial databases

A particularly valuable characteristic of crowdsourced data is the limited time needed in the general data collection process, and even more importantly, the time needed to publish this data on the Web (and thus be accessible to everyone). The whole data production cycle is much more flexible and faster than traditional spatial data collection and publishing procedures. The field of VGI is constantly growing and has an immense potential for mapping, mainly due to two reasons: (a) the technology is mature and is there to stay, (b) other means of mapping may be too slow (e.g. for car navigation systems or disaster mapping) or may be too expensive (e.g. updating the United States Geological Survey (USGS) 1:24.000 map scale on a countrywide basis) [Heipke, 2010]. The importance of this characteristic is paramount when it comes to emergency situations where early warnings are needed. For example, Yates and Paquette [2011] gave a detailed overview of how collaborative technologies and social media were used during the Haiti earthquake of 2010. In such a natural disaster, as shown in Figure 2.6, the availability of immediate responses from the local people on the ground for data collection is of high importance in the delivery and coordination of help from response units (i.e. the case of Hurricane Katrina in U.S.A. where coordination was poor and sporadic). Other sources of spatial data (e.g. satellite images) might not be at hand for a substantial period especially in the immediate aftermath of such an event [Goodchild, 2007].

This up-to-date geographical data can be used by different users to support their proper decision making. For example, firefighters and first-responders can use accurate geographical information to manage the impact of disasters, take decisions to evacuate residents, change management tactics and inform other crews by updating the set of available data on the disaster. Ecologists can employ geographical data to determine the best location to perform observations of animal or plant species and collect data about individuals. Utilities maintenance personnel may accurately locate equipment in the field and update information about their status. More importantly, the characteristics of location-based information is that the richest and most accurate content is local (i.e. specific to a given region, city, or county); therefore, much of it is best maintained locally. The more

local content providers there are the better the information content [Schiller and Voisard, 2004]. All these activities require users to gain access to geographical data visualizations in the field as well as to manipulate them by modifying features and collecting new data. VGI allows all of these tasks to be performed on a continuous 24-7 basis. There are no official release cycles for VGI data - as soon as it is contributed or edited it is available for access by other users.

In most commercial mapping systems, these spatial data are stored as large amounts of tile based maps. It becomes a significant problem to provide timely spatial information in this format because of the time consuming compiling process for the generation of the tiles. This is especially true in cases where the updates might be small (changes of metadata) but spread over a large geographical area: When the data in this database changes, the set of tiles must be regenerated. For large areas, such as countries or cities with multiple levels of resolution this can result in millions of tiles. For small areas, there can be several hundred tiles. As shown in Figure 2.6, this static tile based approach is not suitable for dynamic datasets like VGI, which changes quickly over time during the natural disasters. It is crucial for this situation to have open standard interfaces by which users can instantly publish data and others can consistently access this content.

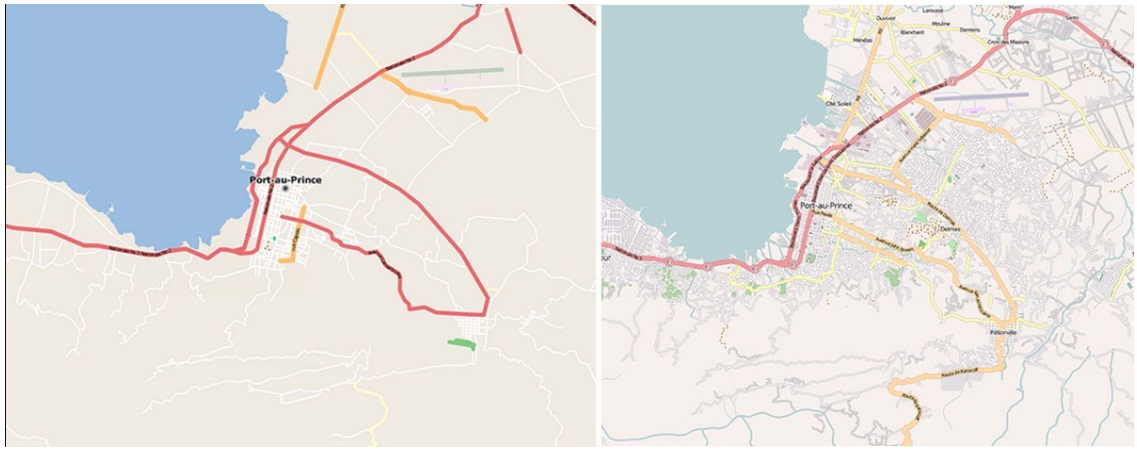

Figure 2.6: Figure (a) shows the original map of Haiti before the Earthquake. Figure (b) shows the OpenStreetMap only a few days after the Earthquake

2.2.4 Transmission format

There have always been discussions about data formats in relation to the development of LBS. In the early stage of LBS development, when mobile phones were client-end devices, raster based formats were widely used in map representation due to limitations of the client-end: poor data storage capability, low computing power and non-open operation systems.Furthermore, the raster based map is easy to implement in most spatial infrastructures [Plewe, 1997]. Most map services (i.e. Mapquest, Bing Maps and Google Maps) actually make use of powerful server farms generating raster map tiles in imagery formats (i.e. PNG (Portable Network Graphics), JPG (Joint Photographic Experts Group image format)). These pre-computed maps are then stored into multi-scale tile repositories and defused to the Web by using WMS (Web Mapping Services). Since the imagery format of raster data can be readable by most browsers and visualization softwares, raster-based maps are widely accepted as the data provision of map services.

However, recent research has shown some drawbacks of using this format for map representation [Boulos et al., 2010]. As he comments, the map area is cut up (to some degree) into arbitrary tiles, which means that cell boundaries usually have no bearing to the real world phenomena that are represented. When using raster maps, the semantic objects are reduced to an incoherent bunch of pixels if there is a rich set of attributes corresponding to the spatial data (i.e. location coordinates, review rates, local names and links). Effective user interaction requires that the individual elements of an object's representation are responsive [Neumann, 2004]. Therefore, an extra overhead of network latency and server processing is added to any request relating to the semantic content of the maps. Furthermore, the pre-computed method also suffers from the necessity of deciding on a single map style, as tiles cannot be generated in real time without a powerful server cluster or cloud system. Thus, map visualization is somewhat constrained by raster-based map tiles, because the style of such pre-rendered maps cannot be changed once it is sent to the mobile users, which makes most raster maps "static" in this sense.

Andrienko and Andrienko [1999] suggest that maps are not only a commu-

nication medium, but also act as tools "to support visual thinking and decision making". As discussed in the previous subsection, an increased level of interactivity has positive effects on user engagement in many aspects. Vector-based maps are emerging, mainly because of the higher quality visualisation and interaction possibilities available [Reichenbacher, 2004]. There were some vector-based maps available even though the early version of HTML Web browsers did not natively support the vector format. Some researchers like Steiner et al. attempted to use flash 5.0 as a plug-in for rendering dynamic spatial data. However, from the developer's perspective, the fact that Flash is a proprietary format makes investing or committing to such a format not always a wise option. Cecconi and Galanda [2002] introduced the SVG format for adaptive zooming in cartography. Dunfey et al. [2006] then created a framework for using SVG as an effective means of rendering GI data. In Lehto and Sarjakoski [2005], the authors use XML generlization for real-time encoding of the spatial data. These XML-based formats did not manage to gain widespread acceptance mainly because of Microsoft's refusal to natively support it in its browser. Upon the emergence of HTML5, some researchers (such as Boulos et al. [2010]) have begun using the canvas element in HTML5 (a new version of the WC3 standard) to natively support the rendering of vector data in web browsers. Figure 2.7 provides an example of a vector map rendered by the canvas of HTML 5. As shown in this figure, the graphic representation of HTML 5 provides a high quality and customised map visualization which could be used in LBS. As stated in [Reichenbacher, 2004] there are many advantages of using vector formats for representing the map. We have summarized these advantages as follows:

- **Context-linked spatial data:** This data format explicitly describes the geometry of geographic features. This allows linking between spatial data and its attributes. After users download the data with its attributes, they can perform local queries on the map features without relying on the connection to the server.

- **More interactivity:** The vector data can be made receptive to user interaction such as changes in focus, mouse clicks, scrolling or zooming the map and the other document events (i.e. viewing the reviews of a shop nearby).

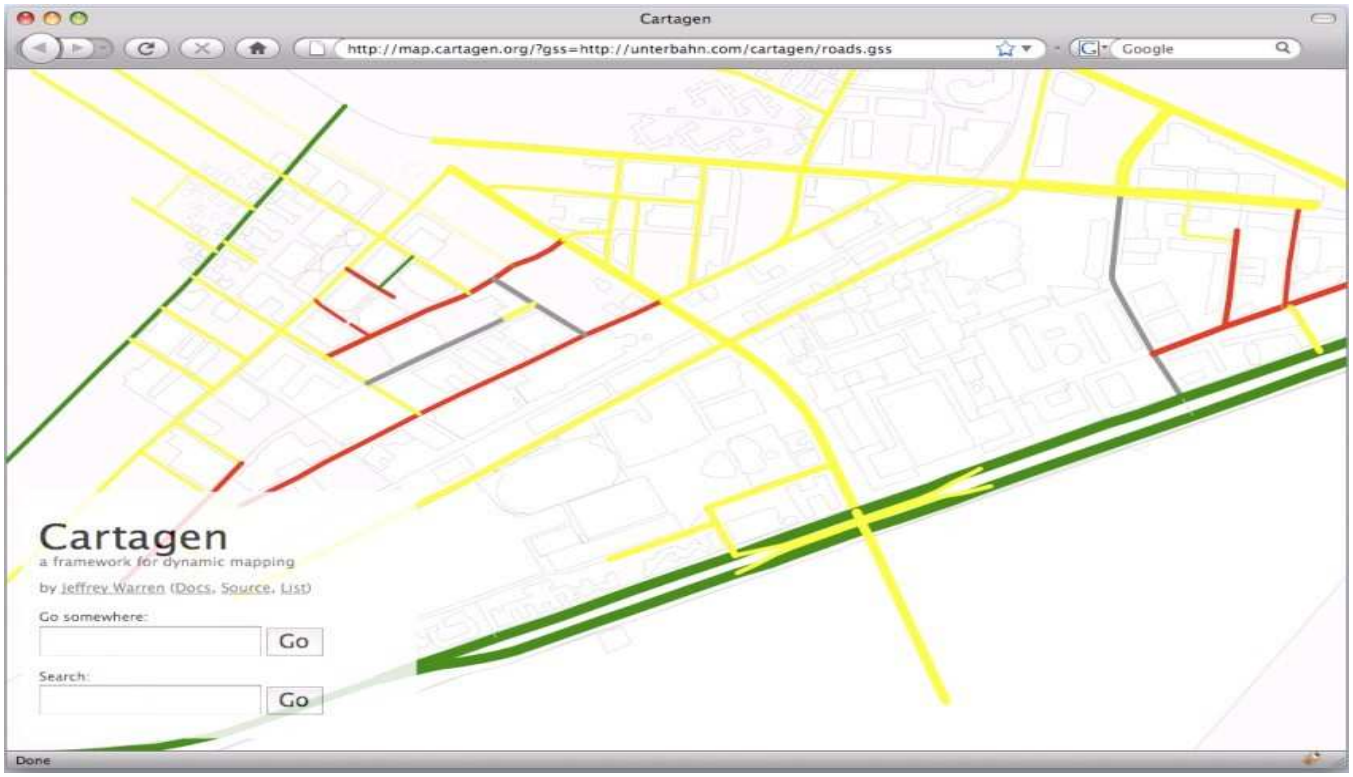

Figure 2.7: A Web based map generated by HTML5 canvas in the Cartegen project [Cartegan, 2010]

- **Resolution independence:** The visualization of vector based data is resolution independent. This is because the vector format consists of a fixed set of shapes while the raster map is composed of a fixed set of pixel dots. Scaling the raster map involves a higher resolution image, which will require more refreshing to update the current map. Scaling the vector map preserves shapes without the need for refreshing.

- **Customized rendering:** Vector data can be rendered in a style specified by users rather than pre-defined on the server side. The ability to store some data on the client device potentially enables the use of real-time generalization of maps. Thus, there is no need to keep the client always online with the server when moving.

- **Timely information:** Since vector datasets can be directly extracted from the raw datasets in the server, they can be consistent with the most up-to-date information collected (such as with VGI data).

Within the development of mobile technologies, the mobile devices has increased data storage capability and more powerful computing abilities. More and more devices are using vector data to represent spatial data in maps, including Android and iOS [Apple, 2012]. Given a vector based dataset stored on a device it is feasible to render a map locally on mobile devices by applying the rules of feature tagging in OSM ([OpenStreetMap, 2012]). Vector-based maps can potentially be used in an LBS mobile application to visualise spatial data. However, there is still no standard way to handle this data format for visualizing maps in mobile devices. Studies in this direction are still at an early stage [Bertolotto, 2007].

2.2.5 Development within a mobile context

When mobile users make use of the location information to conduct LBS tasks they usually require information and services on the move. This is different to fixed office environments. The mobile context is significantly different. This is due to a number of external events that should be considered and to which one often has to respond (e.g. consider being in an airport waiting for a flight) or

to activities that are carried out in parallel (e.g. walking, talking with your friend). Reichenbacher [2004] state that "the interaction styles with mobile devices are different from stationary PCs". It is difficult for users to focus their attention fully on the devices. Consequently, users have less cognitive resources available, and using the device becomes a secondary rather than a primary task. The presence of an interactive map is an intuitive way to enable users to interact with this content. Crampton [2002] catalogues the interactive functions of maps and argues that a highly interactive map provides many of theses functions whereas a non-interactive map (e.g. a scanned raster map) does not offer any interaction possibilities. Vector-based data can natively support interactivity, but also provides serious obstacles when it comes to mobile environments. There are many limitations of the mobile device that must be considered for providing vector-based interactive maps for LBS. These limitations will now be outlined as follows:

1. **Screen Size:** The small screen and resolution issues of the display on mobile devices strongly restrict the data representation and visualisation capabilities of the mobile device. Generalization is necessary to reduce information content in the spatial data. Suitable LoD are delivered to the mobile device for users to perform specific tasks under the knowledge that some data transformations have taken place.

2. **Network Connectivity Performance:** Communication over the Internet for mobile devices is often slow (e.g. 170 kbps in GPRS), expensive, and unreliable (e.g. frequent disconnections) [Ilarri et al., 2010]. Ilarri et al. comments that the latency of network communication heavily constrains the size of geospatial data that can be delivered to the end user. Ilarri et al. also argues that there is an urgent need for more efficient transmission of spatial information as the quantity of spatial data being used and generated by LBS and mobile devices continues to increase.

3. **Device Specifications:** The available memory, processing, and storage specifications of most mobile devices restrict the amount of spatial data that can be processed locally on mobile devices for real-time performance.

This also places restrictions on the types of algorithms which can be used to perform the processing. Consequently some computationally expensive algorithms are not acceptable on a mobile device.

4. **User Interactions:** The types of user interaction with user interfaces of applications on mobile devices can be complex. Unlike web-based mapping applications, there can be a significant number of zoom in/out and pan operations on mobile devices . As Reichenbacher [2004] states these are "tedious and cognitively complicated due to the global context loss". Many users expect "instant information access" using these interfaces. It is therefore necessary to deliver understandable and flexible geospatial information to mobile users for instant access.

2.2.6 Summary of issues illustrated with a case study of OSM

The VGI movement has shown significant potential to provide data for Location-based Services (LBS) [Goodchild, 2009; Mooney and Corcoran, 2012b; Mooney et al., 2010]. As a leading source of VGI data, OSM also has started to provide spatial data for LBS applications. Commercial products include MapQuest [Mapquest, 2012] and the US-based Cloudmade [Cloudmade, 2012]. OSM is freely available under an "Open Database License". The data is available in vector formats (SHP, XML) at the highest resolution possible including all spatial attributes for all geographical features [Ciepluch et al., 2009]. This allows developers to use geospatial attributes representing most features of OSM. The full list of features are available for viewing on the OSM Map Features page [Features, 2012]. As a typical VGI data source OSM data has all of the natural characteristics of VGI data and we use OSM data as a case study in this thesis. Using OSM as the specific example, we summarise the existing challenges of using dynamic crowd-sourced data in LBS as follows:

- **VGI data is stored in large online accessible databases:** The amount of data in these databases has grown dramatically in recent years. The OSM data is also published as complete dumps of the database in XML

and Binary format on a weekly basis. The size of the data in OSM is staggering. Event the compressed XML file of the whole world is over 16 Gigabytes in size. Similar to other authoritative sources of vector datasets such as National Mapping Agencies the size of vector datasets for regions, countries, and continents can be extremely large. It is simply not feasible to provide a map service directly from this data without an efficient data management strategy in place. There are still great challenges in providing a proper data infrastructure for mobile devices. Remote transmission of large amounts of OSM data is also very slow because of the effects induced from a potentially low bandwidth wireless connection. Only a few of these LBS applications attempt to handle raw spatial data at the mobile client side.

- **VGI is created in various representation qualities:** As in any vector dataset real world phenomena are also represented by points, lines, and polygons in the OSM database. While OSM is created and managed by a worldwide group of volunteers, there is no specific subgroup of these volunteers who provide quality assurance and quality control of the new data contributed data. The number of options (devices and software) available to OSM volunteers to add, edit, and contribute to the OSM database means that there are differences in the representation quality of lines and polygons. Some lines and polygons are over-represented (too many nodes) while others are under-represented (too few nodes) given the complexity of the real-world feature the polygon represents.

- **The VGI data in OSM is dynamically changing and is updated frequently**. This characteristic of VGI and OSM is different to National Mapping Agency data, for example, which usually apply much more rigid and well-defined update and change cycles to their commercially available datasets. Large amounts of crowd-sourced data are generated during times of natural disasters and political unrest over a short period [Pultar et al., 2009]. For example, in the hazard of the Tsunami in Northern Japan in 2011, Tokyo-based OpenStreetMap volunteer teams provided round-the-clock crisis mapping support. The landscape and infrastructure of the

disaster area was changing unusually quickly. Consequently, when such environmental or political situations occur, it can be difficult for traditional mapping services to provide up-to-date information and data. With rigid update and release cycles, infrastructure damage and landscape changes caused by natural disasters for example will not be visible in commercial datasets until the next update/release stage. In some cases, this could be only every six months.

- **Transmitting vector-based VGI data to mobile devices for visualizations:** The OSM data available for download or online access is of the highest resolution possible. This is often over-detailed for visualization on the small screens of mobile devices. Users in LBS access OSM data for a variety of reasons. Different users may be satisfied with the differing levels of variations and representations of the same OSM map. A proper data generalization process is necessary to select and deliver the dynamic data such as OSM which meets with the user's preferred representation quality.

2.3 Overview of current technologies for improved spatial data handling in LBS

With the rapid development of spatial data capture technologies and spatial data modelling methods, the size and accuracy of spatial databases have increased dramatically [Mok and Shea, 2004]. Relatively high quality data might be over-detailed in representation with regard to the small screen display of mobile devices. There are many limitations to accessing geographic information in wireless Internet environments and handling large amounts of spatial data efficiently. Moreover, because of the highly dynamic nature of the data, traditional information management techniques are not well suited for LBS. Zhou et al. [2002] indicates "data management in mobile computing environments is especially challenged by the need to process information on the move, to cope with resource limitations, and to deal with heterogeneity (different formats)". In this section, we discuss some related work in different disciplines and describe various solution approaches in regards to these issues. We first review issues of rendering vector

data in low-specification clients. This has been a long-term problem in the computer graphics research domain. We then provide some discussion of existing map generalization techniques, which are used to simplify and adapt map visualizations to a lower resolution screen display. Progressive transmission techniques are lastly discussed for the delivery of map data over limited bandwidth networks.

2.3.1 Contributions from the computer graphics domain

The issues of handling, transmitting and visualizing massive volumes of vector data in low-end clients have been studied for many years in the field of computer graphics. Some of these proposed approaches could provide valuable research results for us towards providing solutions for similar data handling problems with spatial data. Highly detailed geometric models were usually represented by meshes of triangles in computer graphics [Foley et al., 1996]. The massive graphical data for representing complex objects are expensive to store, transmit, and render. Hoppe [1996] pointed out the practical problems of handling data in multiple resolutions. He also explored the key issues of using progressive meshes to represent complex graphical objects with a sequence of continuous decreasing LoD. We have summarized and outlined the key procedures for the efficient display of meshes as follows:

- **Simplification process:** Automatically generalize the meshes and use approximations for efficient rendering. The simplified mesh (M_0) can be represented by a subset of the vertices in the original mesh (M_n) after a number of edge collapse (ecol) operations: $M_n \overset{ecol_n}{\rightarrow} \ldots \overset{ecol_1}{\rightarrow} M_1 \overset{ecol_0}{\rightarrow} M_0$;
- **Multiple representation process:** It is common to represent the meshes in multiple representations using a LoD approximation. Since the edge collapse is invertible, a progressive mesh (PM) can be transformed into successive resolutions with a number of vertex split operations (vsplit) $M_0 \overset{vsplit_0}{\rightarrow} \ldots \overset{vsplit_{n-1}}{\rightarrow} M_{n-1} \overset{vsplit_n}{\rightarrow} M_n$ and the vertex hierarchy structure; Hence, a detailed mesh can be used when the object is close to the viewer, and coarser approximations are substituted as the object recedes from the viewer.

- **Progressive transmission process:** When a mesh is transmitted over a network one would like to show progressively better approximations to the model as new data is incrementally received. Thus, M_0 will be sent first and follow with a sequence of refinements $vsplit_0$, $vsplit_{n-1}$, ...,$vsplit_n$

- **Compression process:** To minimize the storage space some approaches use digital encoding (compression) techniques to store the progressive meshes.

- **Selective refinement process:** When users change their view port they require high detail in a different region. Thus, it is only necessary to select the LoD within these visible areas for transmission.

A complex data model can be efficiently displayed by a much simpler approximation with the help of continuous refinements of LoD. These processes are significantly beneficial to the $2D$ vector-based spatial data visualization when visualizing a large amount of complex spatial objects.

However, for low-specification client side devices it is difficult to reconstruct the PM in LoD because of constraints on the resource budget. Luebke and Erikson [1997] used hierarchical dynamical simplification methods which only maintained a list of visible polygons for rendering. Schmalstieg and Gervautz [1996] proposed an *on-demand* geometry transmission approach that combined several techniques, including LoD, progressive refinement and graceful degradation to deliver the data "just in time" over the network to the rendering application. However, performing LoD tree traversal locally is also expensive (in terms of memory and processing resources) for low capacity clients. Southern et al. [2001] presents a view-dependent transmission scheme for the stateless client without reconstructing the multi-resolution hierarchy on the client by utilizing *frame-to-frame coherence* [Luebke and Erikson, 1997] and client caching. In this scheme, there is more time to render a higher resolution version of the entire dataset. To avoid the impact of the network latency they divided the view-dependent tree into blocks so that the mesh updates are transmitted on a per-block basis and the client can cache a fraction of the view-dependent tree in the form of blocks. Cheng and Ooi [2008] described the progressive transmission of LoD using receiver-driven approaches, which is a solution for interactive delivery by

estimating the visibility and visual contributions of the refinement. Zheng et al. [2008] proposed an interactive approach for view-dependent rendering by introducing a look-ahead strategy to predict the user's view movement and handle frame updating. These view-dependent LoD rendering principles can potentially be applied in a similar situation in LBS applications. This enables spatial data transmission to low-end mobile clients in an *on-demand* fashion. These types of pre-fetching data strategies are useful in situations where there are low-capacity mobile clients. For such devices, it is also necessary to manage the data in a selective manner for efficient visualization. In summary, the strategies outlined above, for efficient data management for meshes. could be useful as the starting point for solutions to the problem of handling large scale spatial data on mobile systems.

2.3.2 Map generalization for small screen display

While spatial data grows larger and more detailed, the screen size of mobile applications has not changed very much. While tablet PCs are becoming popular in the current market, these devices are not in a position yet to replace a smaller sized mobile phone devices. Mobile device users usually have limited screen space to display high quality digital maps which are intended for desktop computers. For example, when users access some tourist photos on a map based on their current location, the server can return many unrelated results. Users that submit popular types of queries such as "search a hotel nearby" are usually overloaded with additional 'pop-ups' which display all hotels in the city. The principal problem in displaying such a map on a small screen is that an unrefined visualization design will emphasise the unnecessary data thereby making it extremely difficult to read or understand because the screen is too small to show everything in detail [Mac Aoidh et al., 2012]. Therefore, map generalization process is necessary to reduce the complexity of the map to give easier access to the spatial information and provide a more user-friendly map interface.

Before displaying complex spatial information, it is necessary to perform a generalization and filtering of the information to obtain suitable maps to fit the small screen of mobile devices whilst exhibiting appropriate map usability. Many

Figure 2.8: An example of the display of a more effective route map for the small screen mobile devices from the work of Agrawala [2002]

generalization techniques attempt to generate various LOD to adapt to the users' requirements [Buttenfield and McMaster, 1991; McMaster, 1987]. McMaster and Shea [1992] provided an overview of different map generalization techniques (e.g. Simplification, Smoothing, Aggregation, Enhancement). However, some complex generalizations take a longer time to generate the required map representations whilst others simpler approaches may not provide high quality cartographical maps. Cecconi [2003] have summarized these approaches into three groups of approaches: On-the-fly mapping approaches, On-demand mapping approaches, and combined on-the-fly and On-demand mapping approaches. On-demand mapping is a prime driver in adapting spatial data to user requirements [Edwardes et al., 2003]. Crampton [1999] defines this as "the creation of a cartographic product upon user request to its scale and purpose". This point is reinforced by the work of Reichenbacher [2004]. Cecconi et al. [2002] and Cecconi and Galanda [2002] implemented on-demand mapping in a multi-scale database. The on-the-fly approach can only afford to perform basic generalization processing. Harrie et al. [2002] implemented a method in which real-time generalization operators were used for displaying building polygons on small mobile devices. As this happens

in real-time, it relies on *on-the-fly generalisation*. This approach uses fish eye techniques to display the map feature and always give high detail in the central place on the map. However, this approach has the limitation of causing distortion of the actual shape features and it assumes the users are always focusing on the central place of the map. This can have the knock-on effect of causing unrecognisable map features in the other areas of the map.

The EU-project *GiMoDig* developed methods for delivering geospatial data to the mobile user [Hampe and Sester, 2001]. One part of this project dealt with real-time data integration and generalisation. Apart from that project, the on-the-fly generalization approach proposed by Lehto and Sarjakoski [2005] uses the XSLT transform for XML-based vector data. The authors claim that this allows emphasis of user requirements and works well for multi-scale data in real time. Similarly, the Cartogen project, by Boulos et al. [2010], provides dynamic map styling with the canvas element in HTML5. Cartogen allows the user to customise the vector data styles for display. Corcoran et al. [2011b] states that visualization of vector data can be achieved using the new methods of inline-SVG and the HTML Canvas. Corcoran et al. also demonstrated the effectiveness of HTML5 for vector data delivery by implementing new HTML5 features (i.e. WebSocket and Canvas APIs) in vector data transmission and visualization. Cecconi and Galanda [2002] proposed adaptive zooming approaches to perform on-the-fly generalization on individual LoD. This approach provides a combination of the on-the-fly generalization and the on-demand generalization. Agrawala [2002] gives another good example of on-the-fly and on-demand mapping. This approach generalized route maps in real time whilst improving map usability to meet specific user requirements for personal navigation tasks(as shown in the Figure 2.8). He also points out that cartographic data is not only adapted to the usability requirements of the user-interface in the mobile context, but also adapted to the limited nature of mobile devices in terms of their technological environment [Agrawala and Stolte, 2001]. As a consequence, and because of the nature of mobile device displays particularly on the smaller mobile devices, one cannot simply rely on the techniques designed for traditional web or desktop applications. Arrie et al. [2002] states that "ideally, the user should have a large-scale map of his immediate vicinity for choosing the right direction at an intersection, for example. At the same time,

the user requires a small-scale overview map where he can see his destination". Sester and Brenner [2004] proposed a model to visualize only the information on the screen which adequately fits the current resolution.

Thus, understanding user behaviour and information preferences is important for map generalization in LBS Li [2006]. Oulasvirta et al. [2011] conducted a study regarding the perceptual-interactive search people perform when using $2D$ and $3D$ maps on mobile devices. They state that mobile map applications should be friendly and with good usability. The visualization of multi-scale maps on mobile devices should also exhibit high usability. The problems of displaying a map on mobile devices are exacerbated by the nature of spatial data where a large information space needs to be presented and manipulated on a small screen [Tonder and Wesson, 2009]. In Lavie et al. [2011] the authors carried out trials of maps for in-vehicle navigation systems. The found that maps with minimal detail produced best performances and highest user evaluations. Cartographic aesthetics were also rated highly by their study participants. In map usability tests carried out by Kratz et al. [2010], the authors found that users preferred map zooming to map panning and scrolling. They comment that the map navigation tasks require users to view large features of the map (in a zoomed-out state) in order to locate the point of interest they are interested in. In [Harrie and Stigmar, 2010] the authors provide some measures of map information that can be used as constraints for the selection of data layers and in real-time generalisation. Harrie and Stigmar [2010] found that using some measures had better correspondence with human judgement than object areas alone, including the number of objects, the number of points, and the object line length. However, their results are only restricted to the building objects.

Furthermore, Arrie et al. [2002] argue that the user should have a fairly simple, non-detailed, map to start with. However, when they continue to work further with the task with the maps, they will need progressively more detail where they will zoom in to their area of interest and potentially interact with the spatial data behind the features on the map. This provides the opportunity to progressively transmit data to the mobile device and build up the detailed map the user requires in carefully managed increments. The next section discusses this strategy of building a map visualisation using this incremental approach called *progressive*

transmission.

2.3.3 Progressive transmission

Transmitting large volumes of spatial data over the internet is difficult. As the scale of spatial data becomes larger, it becomes increasingly difficult to pre-distribute these datasets to mobile applications. Transmission overheads increase significantly when delivering the data from the server to mobile clients. Instead of downloading a complete copy of the spatial data for offline usage these datasets could be managed in a distributed fashion for on-demand access. This would provide significant savings in network bandwidth as much less data would be transferred between the clients and the servers. As we have discussed, this concept was inspired from techniques well known in the computer graphics domain which were described in section 2.3.1. Progressive transmission of spatial data essentially combines several techniques including multiple representation(LoD), progressive refinement, and on-demand delivery to send large amounts of spatial data "just in time" over the network to the rendering process. These techniques allow the applications to "operate on a tight resource budget" which is useful if an LBS application being used in mobile applications constrained by: the small screen size, the limited bandwidth and device storage. As shown in the Figure 2.9, the user can download the initial datasets with coarser resolution for instant access [Arrie et al., 2002] where they can gain an overview of the geographic information. This coarser map is then refined by a series of LoD. This progressive improvement can be stopped at any intermediate level of detail when the user is satisfied with current refinements. This results in a resource saving to the user in terms of both download waiting time and bandwidth usage.

Progressive transmission approaches are widely implemented in imagery transmission over the Internet [Sherwood and Zeger, 1997]. The lowest resolution of the raster data is transmitted as the initial dataset. This initial transmission is then refined with additional pixels. Other approaches use high-rate data compression techniques to gain considerable data size reduction in image using approaches such as JPEG formats [Committee, 1999]. More complex approaches to data compression are provided by Jacquin [1992] who uses fractal theory or the use of

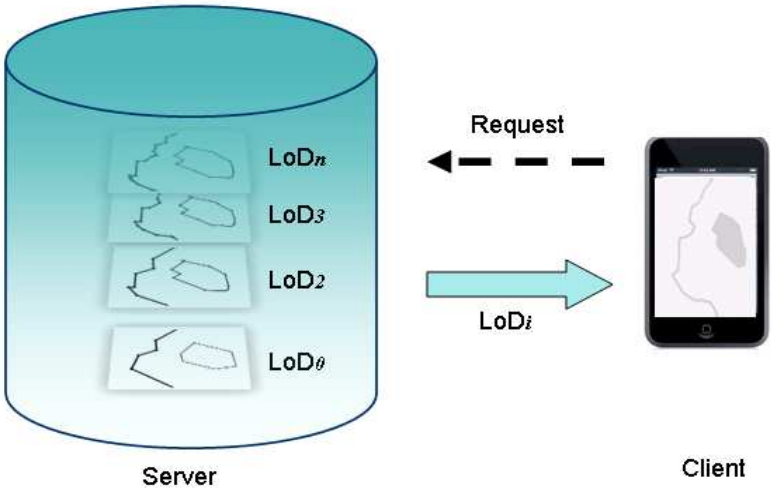

Figure 2.9: An example of the progressive transmission delivery of a spatial dataset to a mobile device user

wavlet decomposition by Grossmann and Morlet [1984]. Tile servers are now very common, it is reasonably easy to deploy imagery approaches to most applications. However, the nature of pixel-based datasets means there is no direct correspondence with issues related to spatial data elements. One of the biggest problems of imagery-based approaches is that there is little scope for interaction with the underlying spatial data. Special overlays (ironically mostly vector-based formats) would need to be provided to provide the user with an interactive experience.

In comparison to purely imagery-based raster approaches, vector-based spatial datasets are more suitable for mobile users to access spatial data instantly and dynamically in LBS applications. Vector-based spatial data progressive transmission has many advantages, which include: the transmission of smaller data sizes, faster response times, and possibly the transmission of only the most relevant details from a spatial dataset [Bertolotto, 2007]. Yang and Weibel [2009] states that the progressive transmission of vector data is beginning to receive more attention as it provides a very promising solution for improved efficiency of data delivery in the web-based environment. The last few years have seen a revolution in how GIS and spatial-data mapping is performed with a serious move away from fixed

desktop approaches to web-based applications and spatial data services [Burigat and Chittaro, 2005; Dunfey et al., 2006]. It becomes a common place where extensive research on vector data progressive transmission has been documented in the literatures over the last decade or so.

Bertolotto and Egenhofer [2001] provided a formal model for progressive transmission based on a distributed computing architecture. Bertolotto and Egenhofer's work is one of the key pieces of literature in this area. Their approach based on a simple client-server architecture and initially provides coarser LoD for instant access. This map is then iteratively refined through the transmission and integration of further spatial data related to the user request. The user can immediately perform analysis using the coarser map and terminate further downloads when the map approaches their desired LoD. An example of Bertolotto and Egenhofer's progressive transmission approach is shown in Figure 2.10. The LoD_0 is the simplified version and can be transmitted to the user immediately. Following LoD_0 the increasing LoD (LoD_1, LoD_2, LoD_3, ..., LoD_n) are transmitted.

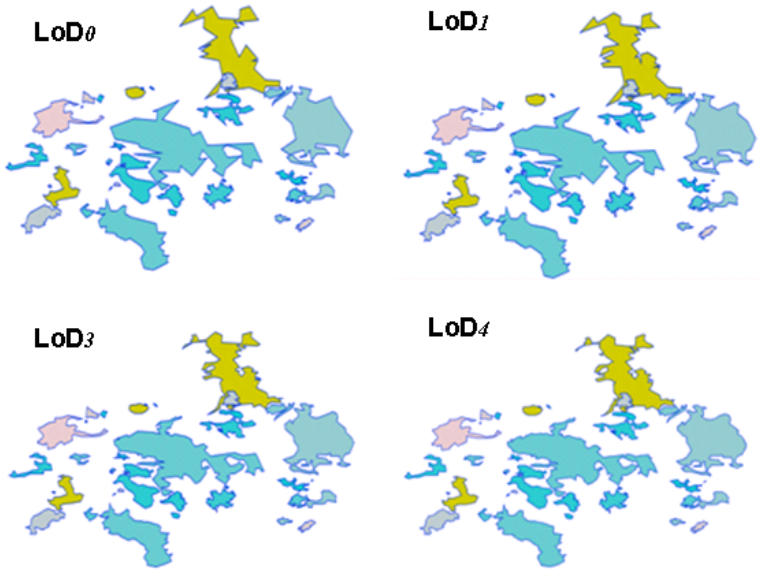

Figure 2.10: A sample of four distinct Levels-of-Detail (LOD) for a set of polygons transmitted to a user device as proposed in the work of Bertolotto and Egenhofer [2001]

Buttenfield [2002] implemented line generalization techniques to pre-compute

the series of LoD for faster transmission. This approach subdivided the polygonal objects by using a modified Douglas Peuker (DP) algorithm [Douglas and Peucker, 1973] and stores the data into a strip tree data structure in the server for delivery to the user upon request. Follin et al. [2005b] proposed a model for managing multi-resolution spatial data in mobile systems. However, there are no performance evaluations shown in these works. Haunert et al. [2009] used a tGAP (topological Generalized Area Partitioning) structure to enable efficient retrieval and query of required LoD, but this approach only restrict the polygonal objects. Zhang et al. [2011] presented a solution for progressive transmission, where the topological relationships are maintained during the simplification by removing a Voronoi diagram's self-overlapped regions. These approaches have taken account of topological consistency in the process of progressive transmission. However, due to the complex computation involved in topology consistency checking, these approaches can only perform the generation of LoD by pre-computation offline. This approach is suitable to spatial datasets of any size which are not changing very frequently. However, these approaches are not suitable for using dynamic data such as VGI as it is updating frequently and the process of topological consistency checking may need to be rerun after each update.

As the volume and resolution of VGI spatial data increases, it is not possible to transmit complete vector datasets over the Internet. Because of the limitations of mobile devices, it is important to implement efficient transmission techniques for quick responses and reduced storage. To overcome the limitations of mobile devices (as described in section 2.2.5) efficient data management techniques are required. Firstly, transmitting these large files over the internet to the mobile device is difficult in practice. Alternatively the mobile device owner would have to download the file at some other location and move it to their mobile device memory card. This would probably require some advanced computing skills and would be beyond many users' capabilities. Investigation of how the data in the dataset is represented could lead to dramatic reductions in the size of the dataset. Douglas and Peucker [1973] and Ramer [1972] described the "Douglas-Pueker" algorithm commonly used for line simplification and well known amongst most GIS practitioners. Line features in GIS datasets are usually of high resolution. These lines may represent linear real-world features such as highways or pedes-

trian paths. The Douglas-Pueker algorithm is often applied to such objects to remove (simplify) unnecessary nodes. This will be discussed in greater detail in Section 3.2. Zhou and Bertolotto [2004] introduce a new data structure used to encode representations obtained by applying Saalfeld's modified DP algorithm [Saalfeld, 1999]. Zhou and Bertolotto's approach not only provides an efficient generalization process for progressive transmission, but also prevents the self-intersections occurring during generalization. This could happen where polygon shapes have nodes removed with causing their shape structure to induce an illegal self-intersection. Kolesnikov [2007] provided a solution for compressing vector maps using polygon approximation and quantization to encode the data into a much smaller volume. Oosterom [1993] proposed a topological data structure for a variable scale representation of an area partitioning without redundancy of geometry.

Since the screens of mobile devices are small, from the viewpoint of visual cognition, the delivery of smooth progressive map visualizations for LBS users is crucially important. Fatto et al. [2008] conducted a survey concerning spatial factors affecting a user's perception in map simplification. These types of popping effects did cause users to negatively percieve the delivered visualisations. One must manage the visual impact of multi-resolution maps. Progressive maps should serve their intended purpose while providing user control for map customisation. It is not acceptable that users of LBS services using a progressive transmission approach are presented with 'jumps' or 'pops' between sequences of LoDs. An ideal progressive transmission would be undetectable to the user. For maps containing many polygons and lines, a methodology for determining a globally suitable generalization is required as the display capabilities of mobile devices are more restrictive than the traditional desktop machine. Sester and Brenner [2004] proposed an approach for generalization of vector data tailored specifically for mobile devices with limited screen resolution. Kjeldskov and Graham [2003] stated that more research was required to understand the usability of LBS and desktop applications using progressive transmission approaches. This research should be focused from a user-perspective and not a purely theoretical viewpoint. In recent work Ying et al. [2011] concluded that some heavily simplified or under-represented spatial objects might display significant "popping

effects" between levels if the transition effects are not adequately smooth. Ai [2010] mentions the "granularity requirement" of progressive maps which means that only when the change between two transmitted datasets is small enough can the human eye experience the effects of continuous or gradual animation of the data. Otherwise, features might "pop" into existence as an effect of the transition from one LoD to the next. To guard against this effect authors such as Yunjin and Ershun [2011] successfully applied advanced techniques such as Integer Wavelet Transformations in vector data compression. Other approaches implement multiple resolutions for generating proper LoDs.

Finally, because there is limited main memory on mobile devices it is necessary to manage the data in a selective manner to avoid overloading data in main phone memory. Augusto et al. [2009] developed a Web-based prototype to manage multiple resolutions using quadtrees. This approach focused on the data management for client interactions with maps and it reduces the irrelevant LoDs when the user is frequently changing their map views. As a user zooms out on a map the scale of the map decreases allowing the user to make sense of the global context. To display a map on a very small scale most of the geometric objects in a map are retrieved and displayed. However, attempting to transmit too many objects will impair the user's viewing of the relevant information. In some cases, large objects, such as national roads, highways and national forests with several hundred segments may occupy only a single pixel on the display. Previous approaches transmit all of these segments from the server side and then draw them onto the client screen. Indeed, most users only require a section of the object, thus, transmitting the complete object is a waste of CPU and memory resources. This problem becomes more serious for applications that view a large scale map in real-time. Corcoran et al. [2011a] classified View-and-Scale based progressive transmission. Antoniou et al. [2009] proposed tile-based vector maps for transmission of spatial data. These view dependent approaches can significantly reduce the amount of data one must store on mobile devices and also simplify the computation complexity of large amounts of spatial objects. Moreover, Mustafa et al. [2006] has proposed an approach to perform the look ahead policy and prefetch the data of the user's next view whilst rendering the user current view. This strategy can overcome the latency of the network, however, there

is no evidence reported from this work. From this discussion of the literature, we can summarise that the application of progressive transmission for visualisation on mobile devices still has many challenges due to these limitations.

2.4 Chapter summary

In Chapter 2, we began with a discussion of the use of spatial data in LBS and also described how VGI had emerged as a new form of spatial data. VGI has grown rapidly and now must be considered as a source of data for LBS. We then identified the challenges of handling spatial data in LBS with emphasis on visualization. OSM data, as a good example of VGI data, is used as a case study in this work. We conducted a detailed analysis of some of the issues related to using VGI in section 2.2. These issues include: dynamic data updates, large volumes of data, and data quality issues. In order to provide high quality and interactive maps, it is promising to use vector-based maps as an alternative solution to more traditional approaches such as tile-based mapping. It is also necessary to take account of the limitations of mobile devices while handling vector-based data. To overcome these issues and limitations we provided a review of current techniques in related fields. In research fields such as computer graphics, several useful and classical techniques of progressive meshes are studied and a number of insights are obtained from the rendering techniques employed for use in low-end clients. Various map generalization techniques are investigated. The implications for map production are also discussed. Progressive transmission of vector-based spatial data provides a promising solution for efficiently transmitting data from the server side to client side. The principle of progressive transmission is suitable for application to a dynamic source of spatial data such as VGI. However, very few of these existing approaches can satisfactorily address the challenges of VGI data. This provides us with an opportunity to provide a comprehensive overview of approaches to managing spatial data for visualization on mobile devices. In the following chapters, we attempt to address these shortcomings and provide a framework to manage spatial data for visualization on mobile devices.

Chapter 3

A framework for vector based spatial data processing

Traditionally, mobile mapping processes was carried out on the server side. However, pre-computed map tiles are not very flexible in responding to the various user requirements of modern map services. Recent developments in mobile graphics hardware technologies enable vector data to be rendered locally by the client-side. Examples include the Android and iOS mobile operating systems. Visualizing vector data on mobile devices is now feasible. It is a promising way for the mobile users to interact with spatial datasets in a more efficient and context-responsive manner.

However, delivering large volumes of spatial datasets over low bandwidth networks is often difficult and leads to a frustrating user experience. We have all experienced the frustration of being unable to use Internet applications on the mobile devices due to poor Internet connectivity. High resolution spatial datasets are stored in spatial databases. Very often these data are too detailed to be displayed appropriately at the scale available due to screen size[Ying et al., 2010a] on most client mobile devices. Additionally these datasets can contain much more data than is required for the task and not necessary for transmission particularly in cases of low bandwidth networks. From the point of view of the user's visual perception it is better to simplify the map to a certain level of representation. This simplification should meet with user specifications, result in a reduction of

the amount of data required for transmission, and not compromise the usability and context of the map. From the user's point of view, this approach will also provide a faster response to the user's spatial queries and interactions.

In this chapter, we will begin by describing some of the characteristics of vector data. We shall emphasize some of the characteristics of the emerging VGI data. We have chosen a well-known example of VGI OpenStreetMap data as a case study. The use of OpenStreetMap allows us to explain the main practical problems with handling large quantities of vector-based spatial data. We then propose a framework for processing large quantities of vector data. This is realised by streaming the data to our application server and processing the spatial data to identify the features of the datasets, which need to undergo simplification. Within this framework, we apply several shape representation metrics to best understand which features should undergo simplification. Finally, we generate the LoD data, which then can be transmitted to the user's mobile device application using a progressive transmission scheme.

3.1 Vector data generalization

In this section, we attempt to establish several shape metrics for exploring the data representation of OSM spatial objects. These shape metrics are applicable to any vector data source. When we apply these shape metrics to our OSM data we compute the values of these metrics on the objects in the dataset and identify the candidate objects for "data generalization" and then marking other objects "ready for delivery" to LBS applications. As shown in Figure 3.1, this process starts to take the OSM objects (extracted from a OSM XML) as input. We will provide a more detailed description of OSM XML in Chapter 4. We use open source software approach, which can extend to consume other vector data formats (e.g. ESRI Shapefiles and GeoJSON).

A pre-defined error threshold is set to control the granularity at which we select objects from the data. The process automatically checks the representation of an object by applying shape metrics. Using the results of these metrics we determine if the object must undergo a set of further processing procedures. The "under-represented" objects are selected as "LBS ready" candidates for im-

mediate delivery to the client device while the "over-represented" ones must be subjected to a series of simplification processes. The workflow of the data generalization processing framework is outlined in Figure 3.1. When all of the "over-represented" data objects have been processed they must be organised in an efficient data structure. For this purpose, a tree based data structure is used to handle generalized LoD datasets for the progressive transmission. The generalized data is adapted to the screen visualization within the proposed projection constraints. We also discuss the issues of topology in regard to the OSM relations.

The theoretical foundations for this framework in the remaining sections are outlined as follows:

- the selection of objects for simplification and generalization
- multi-scale simplification
- the computation of error metrics and adaption of the generalized data objects for the small screen mobile client display and visualisation; the data structures for storage of the LoD
- topological preservation during the simplification and generalization processes.

We begin with the identification of data objects for simplification and generalisation.

3.2 Identifying data objects for simplification and generalisation

Many OSM datasets are generated by the volunteers using GPS-enabled mobile devices. Those datasets are usually recorded at a high resolution. There are often more data points than is necessary for visualization in LBS map services. Redundant data points will decrease system performance in terms of transmission and subsequent visualization. For example, as stated by Meratnia and de By [2004], if data is collected at 10 second intervals, the total amount of data will

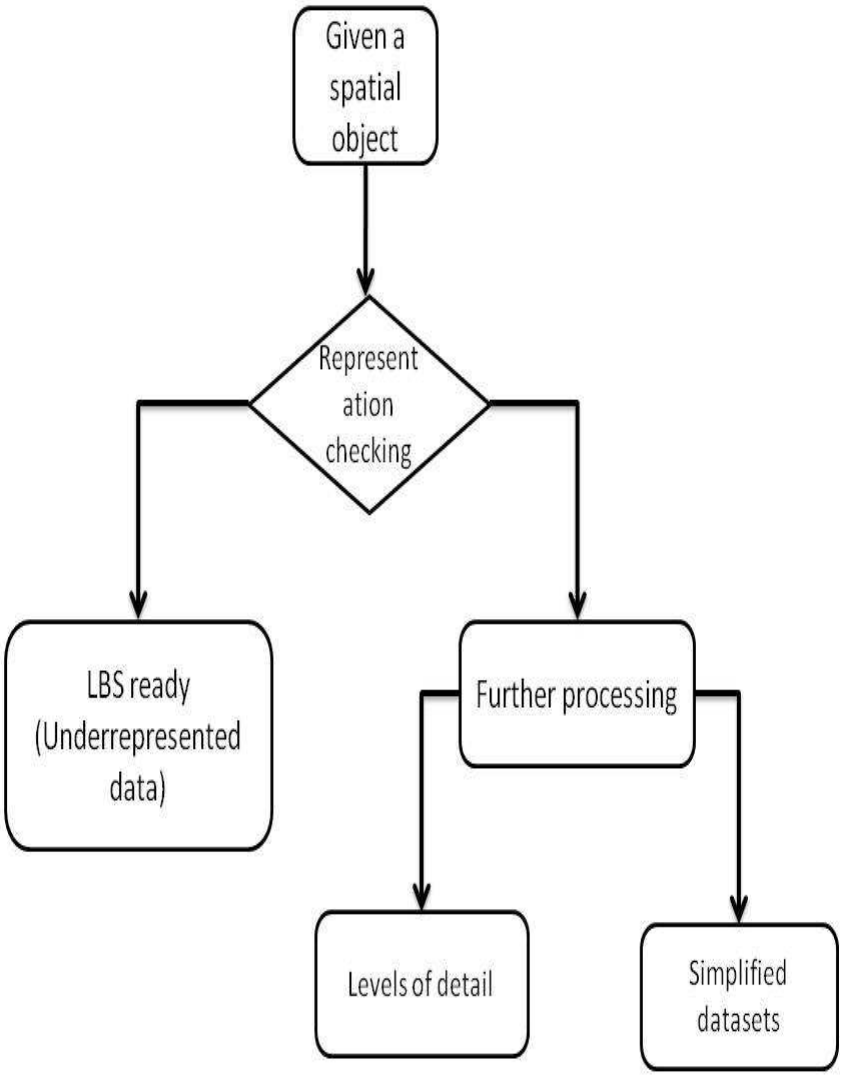

Figure 3.1: The workflow of the data generalization processing component in the data processor component as illustrated in Figure 4.3 in Chapter 4 on page 83

reach 100Mb to request only 400 objects in a typical day of data collection. Very often, there are high resolution straight line features (such as roads and highways) or regular shaped features such as parks, artificial waterbodies. It is difficult to predict in advance without processing, if a given region of OSM will exhibit large numbers of over or under-represented features [Mooney and Corcoran, 2012b]. However, it is not always the case that features are over-represented. For a variety of reasons (VGI data collection using tracing of aerial imagery, lower resolution GPS-enabled data collection and editing by other users) features can also be under-represented. This was shown and discussed in Figure 2.5. For processing OSM data with variable levels of representation quality, we first need to identify spatial objects that are over-represented as simplification candidates; otherwise, they are selected as "ready for LBS" datasets for delivery immediately. This is described in the flowchart of Figure 3.1. The over-represented spatial objects must undergo simplification and be represented as an LoD data structure and simplified down to the given threshold.

A polygon should undergo simplification, if the removal of a subset of the polygon's vertices can be performed without affecting the overall shape, to such an extent that it is not recognized as its original form. It has been shown in the literature that significant visual parts of shapes are somehow related to convexity [Basri et al., 1998]. Only insignificant vertices can be considered for removal during simplification. Latecki and Lakamper [1999] proposed convexity evolution methods and the following metric K determines the significance of each vertex to the overall shape of the polygon in question. As outlined by Barkowsky et al. [2000] it "is in accordance with our visual perception of these shapes". Suppose for some vertex v in the polygon p with incident edges on v called s_1 and s_2 then the K metric for significance is given by:

$$K(s_1, s_2) = \frac{\beta(s_1, s_2) l(s_1) l(s_2)}{l(s_1) + l(s_2)} \tag{3.1}$$

In equation 3.1, the variables are explained as follows: l is the length function normalized with respect to the total contour length of the polygon, while $\beta\ (s_1, s_2)$ is the turning angle at the vertex in question as shown in Figure 3.2. Informally this metric in Equation 3.1 will determine vertices with a greater turning angle

and adjacent edges of a greater length as being more significant to the contour of the polygon. We assume that the vertex with the larger both relative lengths (i.e. $\overline{x_2x_3}$ and $\overline{x_3x_4}$)and the total turn of the angle (i.e. turning angle v_2), the greater is its contribution to the overall shape of a curve (i.e. visual part 2 is greater than visual part 1). Thus, the cost function K is monotonically increasing with respect to the relative lengths and the total curvature. Significant vertices are assigned a high K value with insignificant vertices assigned a low K value in equation 3.1. The main accomplishment of this discrete curve evolution process, described in this section, is that it allows for automated simplification of polygonal curves. Importantly for map display and visualization this allows us to neglect minor distortions while preserving the perceptual appearance [Barkowsky et al., 2000]. Examples of this metric are shown in Figure 3.3. To assess the overall representation quality of polygonal objects p, a threshold λ is given to identify if the object requires further simplification. To determine the significance K of each vertex from a given polygon p the following steps are performed:

1. For each vertex with adjacent edges i and j, we determine its corresponding significance by evaluating $K(i, j)$.

2. Calculate $\sum K$ which represents the sum of K over all polygon vertices

3. For each polygon vertex calculate $KS(i, j)$ which represents the significance of that vertex to the overall polygon shape:

$$KS(i, j) = \frac{K(i, j)}{\sum K} \tag{3.2}$$

4. The mean of the KS values is $\bar{KS}$, it then calculated over all $KS(i, j)$. $\bar{KS}$ has a range between 0 and 1.0.

The establishment of the λ parameter (through a process of experimentation with different threshold values and outlined in previous work Ying et al. [2010a]) allows the simplification of all polygons P within this threshold. The polygons with $\bar{KS}$ higher than λ will be assigned to "LBS ready" candidates for immediate delivery. In Figure 2.5 a polygon identified as "LBS ready" data from the test set

as it has $\bar{KS} = 0.468$ which is a high value. With $\bar{KS} = 0.468$ this corresponds to this polygon shape having many large turning angles with long incident edges. As a result, most of the vertices in the polygon are highly significant and this polygon should not undergo simplification. Alternatively in Figure 2.4 $\bar{KS}$ is small value(less than 0.01), which represents a very low value. This indicates that overall there are small turning angles with short incident edges at these vertices. Given that there is a large number of vertices representing the polygon, some of these vertices could be removed by simplification but the overall structure and visual significance of the shape will not be compromised or lost. Equation 3.2 is easily calculated for all of the vertices in the polygon object.

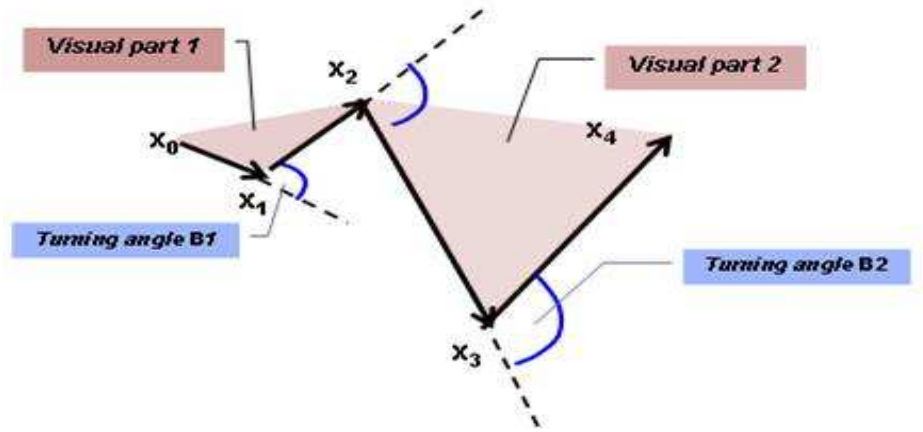

Figure 3.2: Each vertex x_i with the adjacent edge $\overline{x_{i-1}x_i}$ and $\overline{x_i x_{i+1}}$ is the turning angle B. The visual part which is the area representing the overall contribution of this vertex.

3.3 Multi-scale data simplification

It is necessary to consider multiscale data simplification in the context of LBS applications on mobile devices. Crowd-sourced data is a real-time data resource

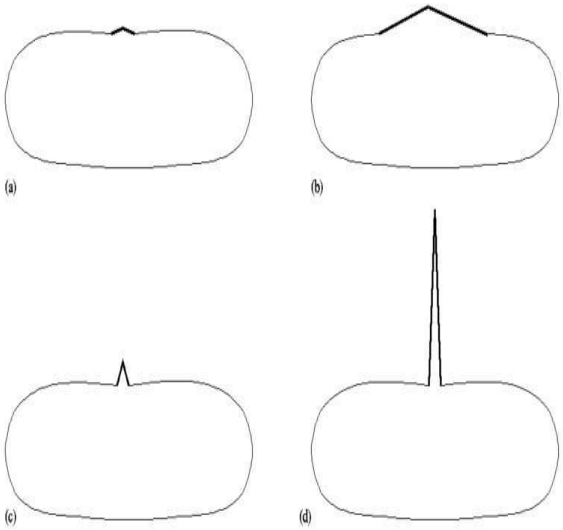

Figure 3.3: The relevance measure K from equation 3.1 of Latecki and Lakamper [1999] is highlighted in bold arcs. The example shows figure d has higher K value than figure a as the vertex has bigger turning angle and longer edges. As outlined by Barkowsky et al. [2000] it "is in accordance with our visual perception of these shapes"

for LBS and can be used in numerous application domains such as: traffic management, navigation, environmental monitoring, and so on. However, this and other data are collected and stored in the OSM database with only one resolution. Usually this is of the highest detail possible as it is extremely rare and unlikely that the contributors to OSM simplify or generalise the data in any way before upload to the OSM database. The narrow bandwidth often available to mobile devices makes the transmission of large datasets very difficult. Transmission of large chunks of datasets can usually mean that more data than needed is delivered to the small screen devices. To avoid this scenario, spatial data simplification becomes particularly important in order to speed up the transmission and spatial operations for mobile users of the LBS. To respond to the users' real time requests, a fast data simplification strategy is proposed which must achieve the following goals:

1. It should perform real time data simplification to generate multi-resolution LOD

2. It should obtain high quality representations of data features, which are appropriate to the resolution and scale. This ensures that the usability of the rendered maps is not affected.

3. The data structure, which organizes the LOD should allow for easy retrieval of the required LOD and with minimum redundancy.

There are many types of data simplification and compression methods. Compression of data is a well studied problem in Computer Science and compression of spatial data is a popular research problem in GIS and Geocomputation. Typically data can be converted and compressed to another file requiring less storage space used such as ZIP [PKWARE, 2012], RAR [RARLAB, 2012] and so on. These universal methods can achieve good rates of compression without losing any information. Very often GIS datasets are transmitted in bulk in ZIP formats as there can be significant reductions in overall file size. This is especially useful in the case of static and non time dependent transfer. In the case of dynamic LBS applications compressing the data using ZIP or RAR is not practical. In particular, in the case of dynamic VGI datasets, the extracted data would need to be

compressed and then uncompressed on the client device. The VGI dataset could be changing very quickly and the compression may need to consider changes to the dataset before transmission. Additionally compression of the dataset would not solve the issue of the mixture of under and over-represented objects within the dataset. More commonly for scenarios involving more dynamic and real-time contexts simplification is applied to the actual vector data itself.

Algorithmic approaches to simplification

Probably the most famous approach to the simplification of vector data is the DouglasPeucker (DP) algorithm [Douglas and Peucker, 1973]. DP is an algorithm for reducing the number of points in a curve or polygon that is approximated by a series of points. The simplified curve consists of a subset of the points that defined the original curve. There are a number of other "fast but sub-optimal methods that can achieve the data simplification with time complexities ranging from $O(N)$ to $O(N^2)$" [Rosin, 1997]. These are mostly heuristic approaches for the approximation problem within the simplification and are summarised by: *split methods* (as outlined in the original DP algorithm in Douglas and Peucker [1973] and subsequently in Shu et al. [2002]), *merge methods* such as those described in M. and J.D. [1993]; Pikaz and Dinstein [1995] and Latecki and Lakamper [1999], *split and merge methods* such as those described in Pavlidis and Horowitz [1974]; Xiao et al. [2001], and also *dominant points detection* as described by Masood [2008]. Most of these methods can achieve up to 80% accuracy compared with optimal solutions. Optimal algorithms include approaches involving dynamic programming algorithms such as that of Kolesnikov and Franti [2005] and the genetic algorithms of Yin [1999]. These optimal approaches can achieve high quality approximation. However, their disadvantage lies mainly in the increased cost in terms of time and space complexity from ($O(N^2)$ to $O(N^3)$). The time and space complexity is only suitable for a small number of vertices and is not acceptable for the proposed issues. One must also consider the unpredictable nature of the representation of our OSM datasets. One cannot easily predict the representation of the objects in a given OSM datasets. Consequently the worst case complexity of $O(N^3)$ could be realised in areas of high resolution

representation such as urban centers, highway centerlines and university campus maps.

In the OSM test datasets, the over-represented polygons will undergo more steps of simplification than other more suitably represented polygons. Given a set of polygons P the following algorithms are applied to each polygon p in the set.

- The DP algorithm has a hierarchical structure starting with a crude initial guess by selecting the vertex at greatest distance from the line segment defined by the first and last point. Then the remaining vertices are tested for proximity to that edge. If the distance *dist* is greater than a specified tolerance ϵ, then the farthest vertex is added to the previously simplified polyline. Using recursion, this "split based" process continues for each edge until all vertices of the original polyline arc lie within an error allowance ϵ .

- The proposed modified algorithm (Algorithm 1) is derived from the work of [Latecki and Lakamper, 1999] and uses a merge-based approach to iteratively estimate the insignificant vertices with low KS values. The KS value for each vertex computation requires a fixed number of operations per vertex (line 2). The elimination of a vertex requires the update of the KS values of only the current vertex's immediate neighbours. The algorithm has a stopping rule based either on the number of dropped vertices or on the current error-value (line 3). At the beginning of the algorithm, the errors are expected to fluctuate around small values when we estimate the vertex with smallest KS. During the iterative process, several levels of error are expected with the potential for noticeable jumps. In the experiments, a practical way to choose a threshold was using some initial runs of the algorithm on the test datasets in order to choose an adequate level of threshold value.

- The dynamic programming algorithm was given in Salotti [2001] and represents an optimal solution to approximate the vector data with overall minimum distortion of the shape but with $O(n^3)$ time complexity. Therefore it is not acceptable for this approach, but the optimal solution can be

used to check the performance and the quality of the proposed algorithm. These approaches use global error metrics to determine the simplification a stopping condition. Given a polygon P =$\{p_1$,p_2, ... ,$p_N\}$ = $\{(x_1$,$y_1)$, ... $(x_N$,$y_N)\}$. The error of the approximation of the curve segments from p_i to p_j of polygon P with the corresponding line segment (q_m, q_{m+1}) of Q is defined here as the sum of squared Euclidean distances from each vertex of p_i; ... ; p_j to the corresponding line segment (q_m, q_{m+1}) as illustrated in Figure 3.4.

$$e^2(q_m, q_{m+1}) = \sum_{k=i+1}^{j-1} l_k^2 \tag{3.3}$$

The total approximation error $e^2(q_m, q_{m+1})$ of the input polygonal curve P by the output polygonal curve Q is the sum of the errors of approximating each segment p_i to p_j of P by the corresponding line segment (q_m, $q_(m+1)$) of Q. The optimal approximation of P is then the set of vertices (q_2, ..., q_M) that minimizes the cost function E:

$$E = min \sum_{m=1}^{M} e^2(q_m, q_{m+1}) \tag{3.4}$$

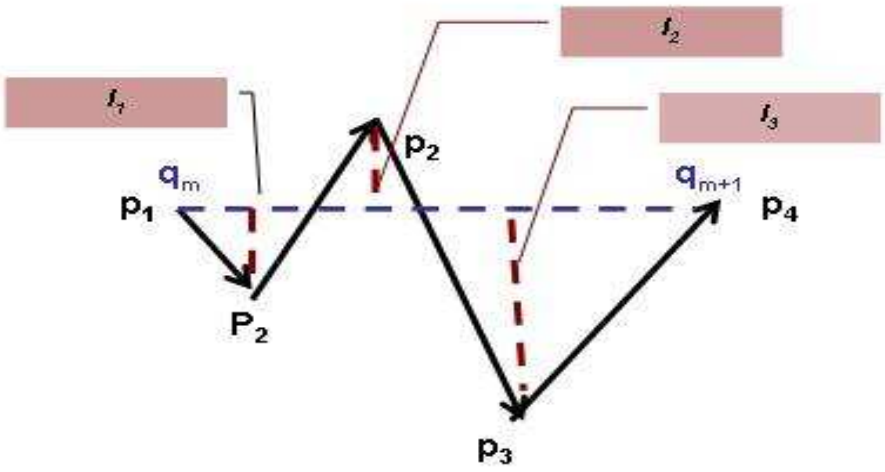

Figure 3.4: The global error for the curve q_m and q_{m+1} is the sum of the l which is the vertex p_k to the line segment p_i to $pi + 1$

Algorithm 1: Data simplification process for merge-based approach

Data: Polygon p with vertex array $p[]$

Result: A simplified Polygon p' with a new vertex array $p'[]$

Data initialization: Let k be the number of vertices to be removed. or: Let ϵ be the threshold. $N = p.length()$;

Calculate $K(i, j)$ for each vertex $p[i]$ and put all vertices K values in the PriorityQueue ($O(n)$ operations);

for $i := 1$ *to* k *(or: repeat until the* KS *of vertex* p *is greater than a given* ϵ*)* **do**

    1.Estimate the $p[i]$ with minimal SK_i value in the PriorityQueue($O(\log n)$ operations);

    2.Compute the error-values of p's two neighbors, say q and r. $O(1)$ operations ;

    3.Update the K for adjacent vertices $p[i-1]$,$p[i+1]$ ($O(\log n)$ operations);

    4.$N = N - 1$ ($O(1)$ operations);

end

This algorithm is implemented in the software tool which we shall describe in Chapter 4. We will now provide a short discussion of the application of this algorithm to some real-world data. We applied the simplification process to some OSM data. Three datasets were chosen from OSM data of different areas with various numbers of nodes as shown in the Figure 3.5. The performance of the simplification process is shown in the Figure 3.8. The results show that the proposed algorithm and DP algorithm can significantly reduce the data within a short time. The DP algorithm is slightly quicker in general. Since the proposed algorithm and the DP algorithm are both suboptimal solutions, when we are applying this simplification automatically, we need to know that the simplification maintains the visual appearance compared to the optimal solution. In order to evaluate the quality of these sub-optimal algorithms, Rosin [1997] introduced a measure known as fidelity (F). It measures how good a given sub-optimal solution is with respect to the optimal approximation in terms of the approximation error. The Fidelity measure F is given in Equation 3.5:

$$fidelity = \frac{E_{opt}}{E_{sub}} \times 100\% \tag{3.5}$$

Figure 3.5: test datasets:1.Erne River in Ireland(OSM id:31025269);2.Finn River in Ireland(OSM id:177922060); 3 A boundary of Howth in Ireland (OSM id:42137400)

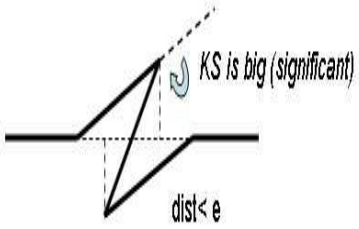

Figure 3.6: The Douglas Peuker algorithm will estimate this vertex within the threshold. However, the proposed approach recognizes that it is significant as it has a large turning angle and then long adjacent edges

Figure 3.7: Fidelity (F) of solutions for the test shapes by proposed Merge based approach and Douglas- Peucker (D-P) algorithms. The results are for 5 consecqutive LoDs.

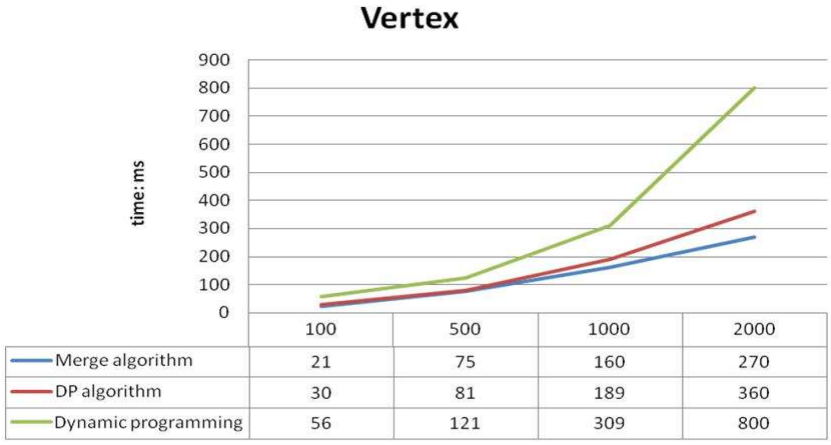

Figure 3.8: The time complexity of processing simplification of given datasets by the proposed Merge based approach and Douglas-Peucker (D-P) algorithms

Figure 3.9: This is the interface of the demonstrating application. It compares the difference of three algorithms: the proposed algorithm (in red line), DP algorithm(in green line) and optimal solution(in yellow line) with respect to the original map (in black line). The preserved nodes have been shown with corresponding processing time. The red drag bar on the top allows the user to adjust the error thresholds.

The results in Figure 3.7 show that the proposed algorithm is better at preserving the visual features of the map compared to the DP algorithm according to this fidelity measure. This is because in some cases, like Figure 3.6, The DP algorithm will estimate the vertex with the threshold without observing the large turning angle. As shown in Figure 3.9, the simplified polygons generated by these three algorithms demonstrate the different performance of preserving visual features, the line produced by this algorithm lies between the other two algorithms. Given a full resolution polygon P with level L_N, at each level from L_N down to L_0, the lowest level of simplification, the nodes with the minimum $K(i, j)$ are removed by this approach with time performance $O(N \log N)$. This is better than the DP algorithm which has a worse case complexity of $O(N^2)$. Moreover, for use in progressive transmission, this algorithm can automatically generate the sequence of simplification by putting the most insignificant vertex in another PriorityQueue ($Q2$) when it is estimated from the list iteratively. This PriorityQueue can be used to prepare LoD for the progressive transmission process.

We have dealt with preserving the overall visual appearance and contour of polygons and polylines which must undergo simplification, we must also consider how to display of these simplified shapes on a small screen. The next section deals with the development of error metrics for small screen display of the simplified shapes.

3.4 Error metric transformations for small screen displays

When users zoom a map or when there are changing resolutions of the display this can usually lead to the granularity of presented spatial objects changing. For smooth presentation, we must make projections and mappings of the coordinates systems between screen scale and the real data scale. This provides for the adaptation of the error threshold of simplification to the corresponding scale. This is shown in Figure 3.10.

Given each vertex v_i of a polygon $p = v_1 \ldots v_n$ with its positional coordinates (X_i,X_i) in the OSM data, the AOI (Area of Interest) is selected by user with

$scale_m$ (X_l,X_r,Y_t,Y_b) in OSM where X_l, X_r, Y_t, Y_b is the boundary of left, right, top, bottom of the scale respectively. The user specified error threshold in the mobile screen is defined as ϵ. The following function is defined to transform the coordinates from screen scale to the scale of AOI for the simplification of OSM data.

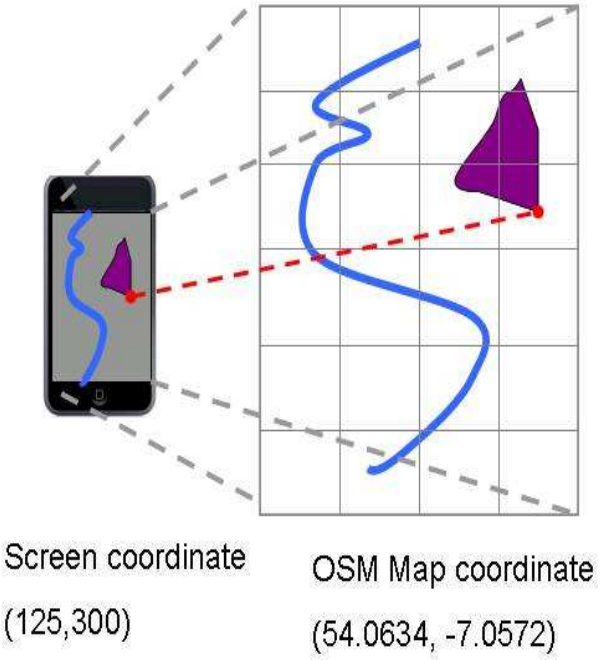

Figure 3.10: The coordinates in mobile screen are (125,300) in 640 * 400 resolution, the actual coordinate in OSM is (54.0624, -7.0562) within the bounds (-7.0572(l),-6.9712(r),54.0634(t),54.0284(b))

$$P(X, Y) = f(X, Y, scale_a); \tag{3.6}$$

The equation 3.6 shows this concept in mathematical notation where $P(X, Y)$ is the positional projection of the vertex v (X, Y) on mobile devices. To derive $P(X, Y)$, we use the following constraints for the projection:

1. Suppose the scales of AOI on mobile devices are ($scale_1$, $scale_2$, ..., $scale_n$)

as illustrated in Figure 3.11. In this image $scale_1 > scale_2, > \ldots, > scale_n$.

2. The projection f_i of v (X, Y) on the mobile display will be $\tilde{v}$ $(\tilde{x},\tilde{y})$ in $scale_i$

$$f_i = (\frac{scale_i}{scale_a}); \tag{3.7}$$

The projection coordinates of v on mobile device can be calculated by the following equations

$$\tilde{x} = f \times (x - x_l); \tag{3.8}$$

$$\tilde{y} = f \times (y - y_b); \tag{3.9}$$

The error allowance of AOI in screen area can be translated to the error metric of the actual OSM map data using the following equation:

3.

$$E = \frac{\epsilon}{f_i}; \tag{3.10}$$

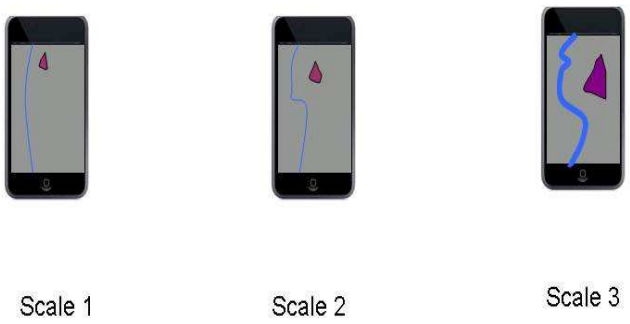

Figure 3.11: In this figure three different scales are presented. The same map is displayed at different scales on the same mobile device screen

E is the error threshold for the OSM data simplification under f_i. When the user is continuously zooming to a given level of resolution, E is derived by the constraints above, so features are represented at the appropriate quality, the

screen projection coordinates can be calculated. For example, given an AOI scale $scale_a$ of $1km^2$ and a screen scale $scale_i$ of 400×400 pixels resolution where 1 dot pixel is the error allowance ϵ for the mobile user with this screen resolution, we calculate that E is within $2.5m^2$ ($\epsilon \times 1000/400$) for the threshold of simplification of this vector data.

3.5 Data model for handling incremental LoD

After the data has been generalized with the simplification process, it is necessary to generate the incremental LoD. Several other authors have described efficient transmission strategies for large amounts of data using multi-resolution approaches. The best known of these include Real-Time Generalization (RTG) by Oosterom [1993] and Lehto and Sarjakoski [2005], and the LoD approach by Cecconi and Galanda [2002] and Sester and Brenner [2004]. As the data we are dealing with is usually over-detailed with respect to the user's mobile computing needs there have been a few different works which have focused on methods to transmit the resolution of data which is specifically adapted to the user's query in a mobile or web-based context [Follin et al., 2005b]. Cecconi [2003] used a mixed approach which can compute the intermediate states between LoDs. Bertolotto and Egenhofer [2001] and Zhou and Bertolotto [2004] proposed a model for structuring multi-resolution vector data in a hierarchical tree structure. Sester and Brenner [2004] proposed a continuous generalization strategy based on the incremental transfer of generalization operators g_i.

A formal expression of the progressive transmission of the incremental LoD representation P_n of a polyline into its coarser one P_m is outlined as follows:

$$P \equiv P_n \equiv P_{i0} \xrightarrow{g_0} P_m \tag{3.11}$$

Given a polygon P with a set of vertices v_1, v_2 $\ldots v_n$ we then perform the simplification approach as outlined above and generate LoD. Given an ϵ, it can generate a corresponding LoD. If it performs the continuous generalization on the data and generate the multi-resolution within different ϵ such as $\epsilon_0, \epsilon_1 \ldots, \epsilon_m (\epsilon_0 > \epsilon_1 \ldots, > \epsilon_m$ and $\epsilon_m = 0)$, then it results in a set of LoD which can be represented

as l_0, l_1 $\ldots l_m$ $(m < n)$. Level 0 is the most simplified version while level m is the full version of the map data. So the vertices of the lower version of LoD l_i $(0 < i < m)$ are the subset of the full version l_m. If performed with continuous generalization and iteratively remove a number of vertices within the ϵ in each l_i, the difference between the levels is defined as a set of vertices V_i. As the OSM is a single scale dataset, the incremental LoD for progressive transmission is formalised as follows:

$$l_0 \overset{V_0}{\rightarrow} l_1 \overset{V_1}{\rightarrow} l_2 \ldots \overset{V_{m-1}}{\rightarrow} l_m; \tag{3.12}$$

For split-based approaches such as the DP algorithm mentioned above the simplified dataset can be organized by using a Binary Line Generalization Tree (BLG tree) as described by Oosterom [1990]. The BLG is an efficient scaleless data structure for a single polyline object. The BLG stores the results of the simplification as a binary tree and consequently there is no need for further simplification processes to display the multi-resolution data. Figure 3.12 illustrates the procedure for generating a BLG tree from the DP algorithm and the resulting hierarchical tree structure. It is easy to understand the hierarchy of the polyline nodes in terms of their position in the binary tree. This approach by Oosterom [1990] has several practical advantages. These are summarised as follows:

1. This approach requires a diminishing number of calls to the DP algorithm
2. This approach is flexible to changing the number of points on the line and generalising the map data according to different scale requirements;
3. The whole approach can be speeded up by retrieving a line at a specific scale;
4. This approach maintains the topology of spatial data.

For the proposed merge-based approach, a Priority Queue is a more efficient data structure for handling the vertices with various importance values when they are characterised by priority level [Becker et al., 1991]. Higher importance can be characterised by a high KS value (3.2) being assigned as high priority. This implies that a geometric object appears on a map only if its priority is high

enough. The object is represented only by those of its defining vertices that have sufficiently high priority. Figure 3.13 shows an example of this implementation. Each priority level will be given a priority index number. Lower index numbers correspond to the smaller scale maps while higher index numbers belong to larger scale maps. For each scale object, those that have higher priority values than the priority corresponding to the map's scale will not be retrieved. In general, the Priority Queue tree is a height-balanced tree. This is where the PR-Tree has a great advantage over the BLG-Tree, because the BLG-Tree can become grossly unbalanced.

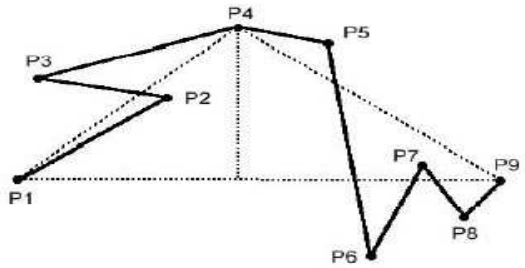

Figure 1a. The original polyline

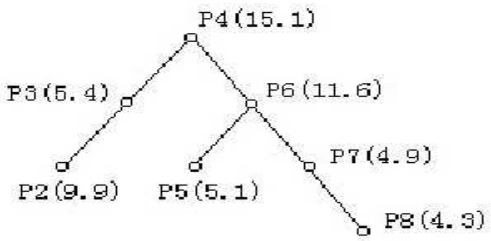

Figure 1b. The BLG-Tree of the polyline

Figure 3.12: This example illustrates the procedure for generating a BLG tree using the Douglas Peuker algorithm. The resulting hierarchical tree structure is also shown

Both of these techniques allows us to perform efficient retrieval of the data in a multi-resolution context. Generalization of the OSM vector data is performed on the fly when a user requests a specific geographic area from the OSM dataset.

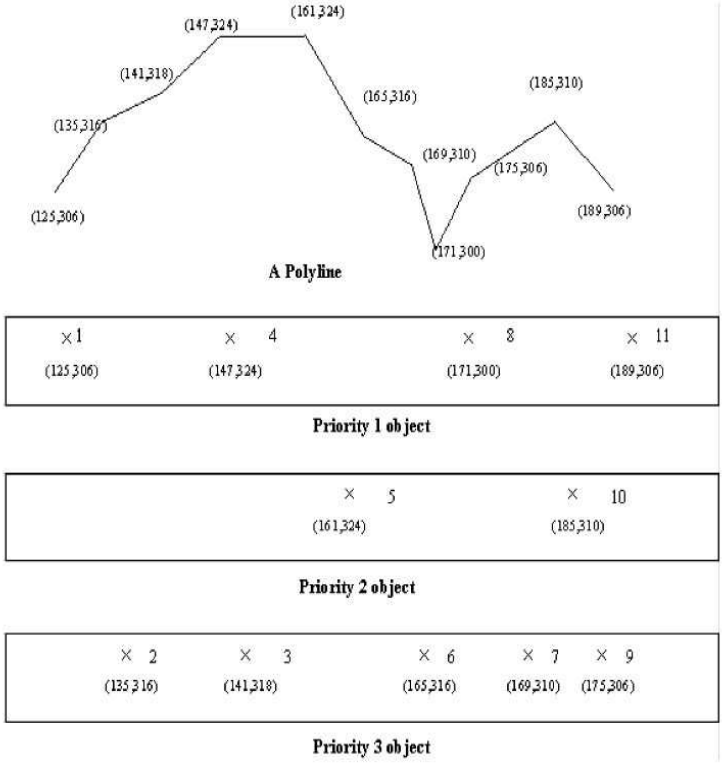

Figure 3.13: All of the nodes in this polyline are arranged into priority levels based on their KS values. One can see from the illustration that the first and last node and two of the intermediate nodes have the highest priority

As discussed above the simplification of polygons can adversely affect the overall shape of polygons contours and geometrical structure. Only insignificant vertices can be considered for removal during simplification. As shown in Figure 3.14 the most significant nodes will be promoted to a higher level while the unnecessary nodes will be demoted to a lower level for transmission later. These unnecessary nodes will not enhance the visual structure of the object for immediate viewing and visualisation on the client device. This process continues until all of the nodes are stored in a special data structure. This data structure maintains a priority sorted order on the nodes by ranking them by which level of detail they belong to. The highest LoD level 0 which provide the lowest representation will be delivered immediately for quick rendering and visualization. The lower LoD, which corresponds to higher resolution will be sent incrementally then to refine the previous LoD. However, when the user moves around the screen area, the method will accelerate this refinement process in the original user-specified region to enhance the visualization quality rather than in regions outside of this.

The processing related to the framework above is performed on the server machine(s). The server machine could have a local mirrored copy of the OSM database which is updated using frequent "diff" applications to the database to ensure that it always has the most up-to-date OSM data. Alternatively the server machines could download OSM data over the Internet. However, this approach is a little error prone due to high traffic on the OSM servers. The data structure for organising the vertices and the polygons in the OSM dataset are shown in Table 3.1 and Table 3.2.

3.6 Topology preservation during data processing

Topological consistency is a practical problem for vector data simplification applications [Bertolotto, 1998; Corcoran et al., 2012; Poorten et al., 2002; Zhou and Bertolotto, 2004]. The solution to the problem is not trivial and it is a time consuming task [Kolesnikov, 2007]. A common topological problem is the one of processing that causes self intersections to occur in a polygon. Some approaches

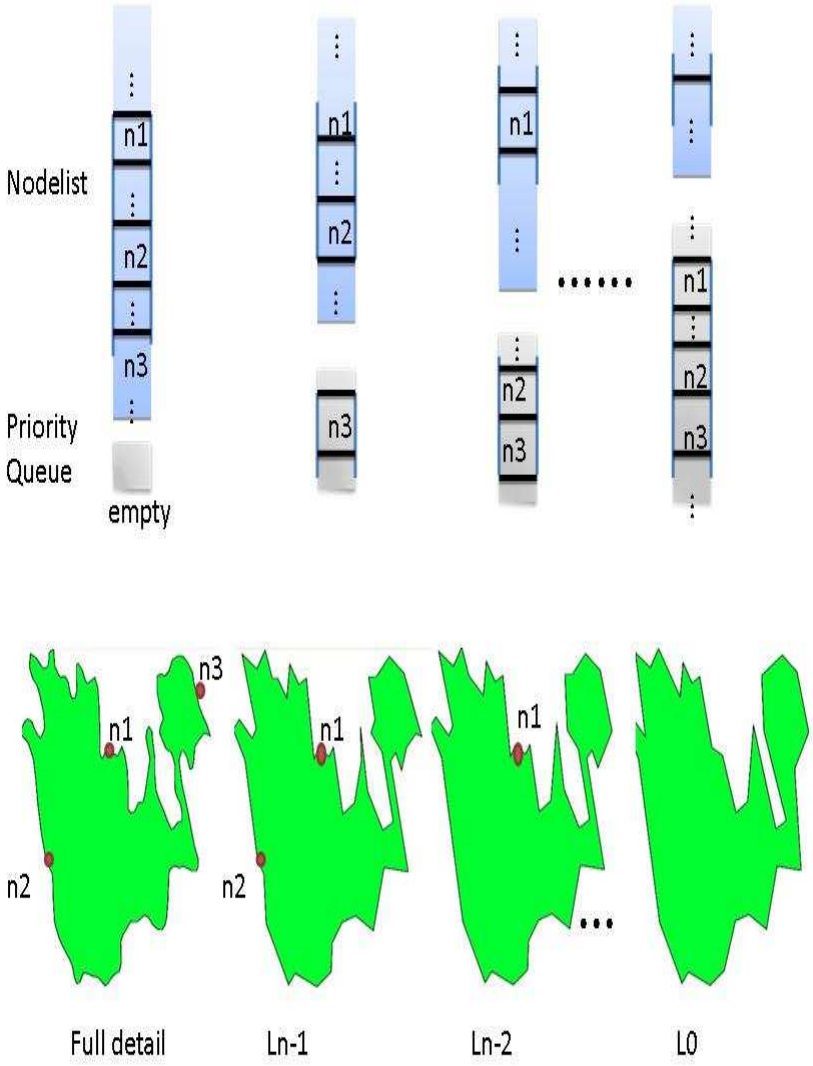

Figure 3.14: This is an example of the concept of LoD. The nodes are estimated according to the significance to the shape. In the figure, the most insignificant vertices are removed from the objects first, which has been shown in the nodelist of Full detail level and the corresponding shape. The removal nodes will be put into the Priority Queue in the lower level($L_n - 1$, $L_n - 2$,. . ., LoD_0). The earlier the node is put into the Queue, the later the node is being used since it is more insignificant as shown in L_0.

Data Name	Data Type	Description
(Lat,Lon)	Geographic Coordinates	Location Information
Next,Prev	Vertex ID pair	Previous and Next Vertices
Error	Double	Approximation error for this vertex
NodeId	int	Unique id for this vertex
Removed	Boolean	Vertex accepted as a candidate for removal
K_Level	Integer List	The level of details (LoD)s this vertex appears in

Table 3.1: This table shows the key variables in the data structure containing all of the vertices in a dataset

Data Name	Data Type	Description
Polygon ID	Integer ID	The unique ID of the polygon
Vertex List	List of Vertices (Vertex IDs)	List of Integers representing vertex IDs
CompSim	Double	The similarity score compated with the original shape of this polygon
Inside	Boolean	If there is a self intersection or another object inside this polygon

Table 3.2: This table shows the key variables in the data structure containing all of the polygons in a dataset. The data structure for the vertices in the dataset is given in Table 3.1

can avoid self-intersections occurring during each step of simplification by using Voronoi diagrams [Mustafa et al., 2006]. Other approaches can preserve "unremovable points" in an object [Barkowsky et al., 2000; Saalfeld, 1999] whilst one can also remove vertices guided by some simple constraints [Yang, 2005] to achieve topological consistency during the simplification process. However, efficient topology checking is still an open question in this area.

OSM data contains various topological relationships, which are expressed in the relation element. The relation element of OSM data will be described in more detail in the next chapter. Relations, in OSM, are logical groupings or arrangements of OSM objects (points, lines, polygons) and some rules specifying their topological relationships. The sample OSM code in Code Snippet 4.4 shows one building with two inner rooms which is then visualised with the illustration in Figure 4.5. This is a very typical relation description in OSM data (expressed in this case in OSM-XML). The polygon contains "inner rings". During the simplification process, we must check that the simplification process is correctly preserving these types of topological relationships in the data. This approach to preserving topological relationships must be consistent. An inconsistent approach can lead to unpredictable or undesirable results. In OSM data, many contributors attempt to represent the geographical reality as accurately as possible in the OSM database. This can include large polygons acting as 'containers' for other polygons. For example, a building polygon includes many small inner polygons which represent rooms. Implementing procedures to check for topological consistency during data processing is reasonably straight-forward. However, these procedures are time consuming and must be managed correctly as they could cause degradation in performance of delivery of the map data to the user device. Most geometry processing libraries contain highly efficient and optimised routines for checking topological relationships such as self-intersections and overlaps.

3.7 Chapter summary

In this chapter, we have provided a mathematical description of a framework for processing vector-based spatial data with OSM as a case-study. The chapter discussed the unique characteristics of OSM as a spatial data source. We detailed

the mathematics of shape simplification, which identifies the insignificant nodes removal will not have a detrimental effect on the overall shape of the polygon or polyline. We also discussed the transformation of the data for display on a small screen device. The data structures used to store the LoD are described so that the simplified data can be stored and retrieved efficiently. The chapter closed with a short discussion of the issue of topological preservation during data processing.

The next chapter will outline a software tool we developed to process the OSM data and implement the shape processing techniques outlined in this chapter. As stated in this chapter OSM data is primarily exchanged between the OSM servers and clients in OSM-XML format. Many GIS practitioners are a little hesitant in handling XML data. The development of software should carefully encapsulate the details of the XML data processing in the spatial data processing component of the software. We will also provide some details on the way spatial data is represented within the OSM-XML data format but it can be extended to any well known vector based spatial data formats.

Chapter 4

A software tool for processing vector based spatial data

In this chapter, we implement a software tool based on the framework we have described in Chapter 3. This software tool resides on the server machine. The tool performs the spatial data processing required to respond to client requests. It delivers the spatial data using progressive transmission schemes to the mobile device. This chapter examines various features in the OSM XML data format and proposes an XML processing algorithm based on opensource. We discuss how we process the data with this software tool. Finally, we provide a summary of the key components in this software tool.

4.1 Understanding OSM-XML data

We decided to use Java as the implementation language. In addition to this Java has a wide range of libraries for XML processing, file input-output, networking, which are efficient and easily used. Primarily this tool has been developed using open-source components. Using the Geotools tool-kit (http://www.geotools.org/) this software can easily adapt to the most popular vector-formats (such as Shapefiles, geoJSON, and KML) with minimal reconfiguration of this software. However, the emphasis from this point will be on the processing of OSM data. As we attempt to access the most up-to-date data

from OSM, this software tool is configured to process the OSM-XML data format. OSM data can be downloaded and accessed in two ways. Sources such as Geofabrik (http://www.geofabrik.de/) offer OSM data in OSM-XML and ESRI Shapefile formats. Geofabrik updates these files every three hours and conveniently cut the world OSM file into country and regional extracts. These files are easier to work with than the entire world OSM file particularly when one is only interested in a small area or region. For networking applications OSM raw data is freely available for accessing, querying and editing by using a RESTful API [API, 2012] which returns OSM data in OSM-XML format. Users can send HTTP requests to download up-to-date data from the OSM database. This approach downloads OSM data using the REST API to ensure that we have access to the most up-to-date data possible from the OpenStreetMap project. Developers of applications using OSM also have the option of hosting their own local network mirror of an OSM database. The OSM-XML file can be downloaded from Geofabrik and imported into a spatial database management system such as PostgreSQL PostGIS. This approaches offers the benefit of local network access to OSM data. However, this data can very quickly become 'stale' and out-of-date. Software options are available to apply "diff" patches to the spatial database to update the OSM data in the database by the minute, hour or day.

The OSM XML file usually has a clear human readable structure. The XML represents a list of OSM data primitives (nodes, ways, and relations) that are the architecture of the OSM model. We illustrate below a general OSM XML file example for representing a spatial object in a real world map. The Science building of NUI Maynooth is shown in Figure 4.5 and the associated OSM XML is explained as follows:

- *OSM* (An example is shown in Listing 4.1) . An OSM element has been included in the XML file. The OSM element contains the version of the API and the generator that created this file (e.g. an editor tool).

```
1 <osm version="0.6" generator="CGImap 0.0.2">
2 <bounds minlat="53.3826430" minlon="-6.6015530" maxlat="
      53.3834170" maxlon="-6.5996000"/>
3 // include ''way'',''node'' and '' relation'' here
4 ...
```

```
5 </osm>
```

Listing 4.1: OSM request is wrapped as an osm element

- *Node* (An example is shown in Listing 4.2). A list of nodes represent geographic points with a latitude and longitude. The element contains coordinates under the commonly used $WGS84$ reference system. Other editing information is included such as who created the dataset ("user") and when the changes happen ("timestamp").

```
1 <osm>
2  <node id="391294644" lat="53.3829170" lon="-6.6013797" user="
       Blazejos" uid="15535" visible="true" version="1"
3     changeset="1095229" timestamp="2009-05-06T10:50:54Z"/>
4     ...
5 </osm>
```

Listing 4.2: An example OSM file with the node elements highlighted

- *Way* (An example is shown in Listing 4.3). This is a spatial object entity with a block of vertices ("nd") which use "ref" to link to it's nodes ("node") and their nodes ids ("id") in the list 4.2. Depending on whether the first reference equals the last one a *way* is called *closed* or *open*. Closed ways represent polygons while open ways represent polylines.

```
1 <osm>
2 <way id="34128421" user="mikeg88" uid="587314" visible="true"
      version="11" changeset="10435974" timestamp="2012-01-19T11
      :02:49Z">
3   <nd ref="391294644"/>
4   <nd ref="391294645"/>
5   ...
6   <nd ref="391294664"/>
7   <nd ref="391294644"/>
8   <tag k="amenity" v="public_building"/>
9   <tag k="building" v="university"/>
10  <tag k="name" v="Science Building"/>
11  </way>
12 </osm>
```

Listing 4.3: An example OSM file with way elements highlighted

- *Relation* (example in Listing 4.4). A relation in OSM is a logical grouping of *node*, *way* and potentially some other *relation* as its *member* with its non-spatial feature notations *tag*. This is where the OSM database can form multi-polygons as complex objects with relations. Each entity participating in a relation must play a certain *role* in it. Relations can represent geographic features such as large cities. More commonly relations are used to create logical geographical groupings such as housing estates, administrative districts in cities, and groups of water features.

```
1 <osm>
2  <relation id="1078860" user="Blazejos" uid="15535" visible="
       true" version="1" changeset="5269515" timestamp="2010-07-20
       T11:01:21Z">
3   <member type="way" ref="34128421" role="outer"/>
4   <member type="way" ref="34765096" role="inner"/>
5   <member type="way" ref="34765097" role="inner"/>
6   <tag k="building" v="yes"/>
7   <tag k="name" v="Science Building"/>
8   <tag k="name:ga" v="Foirgneamh na nEolaiochta"/>
9   <tag k="type" v="multipolygon"/>
10  </relation>
11 </osm>
```

Listing 4.4: An example OSM with relation elements highlighted

As shown in the Figure 4.1, the abstract structure of OSM data can be expressed in the understandable form of OSM-XML. We take one OSM dataset as an example: the Listing 4.1 defines a bounding box of the area in an OSM map which is shown as an actual map in Figure 4.4. It represents a science building with two rooms in this building (denoted with yellow lines), thus the Relation for these spatial objects is: Science Building is an "outer" object while the other two rooms are "inner" objects within OSM. We do not focus heavily on Relations in the OSM XML in this work. Relations are the most complex form of

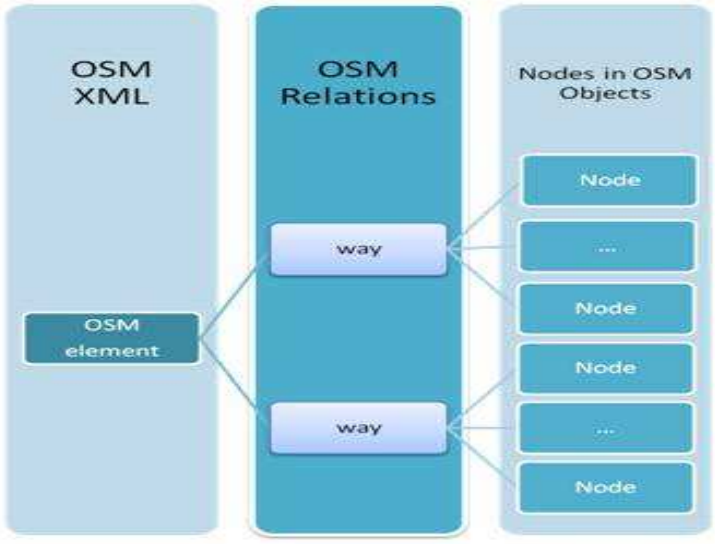

Figure 4.1: This is the structure of OSM XML (*OSM* 4.1), which has included multi-objects(*Way* 4.3) in the Relations (*Relation* 4.4) with their nodes (*Node* 4.2)

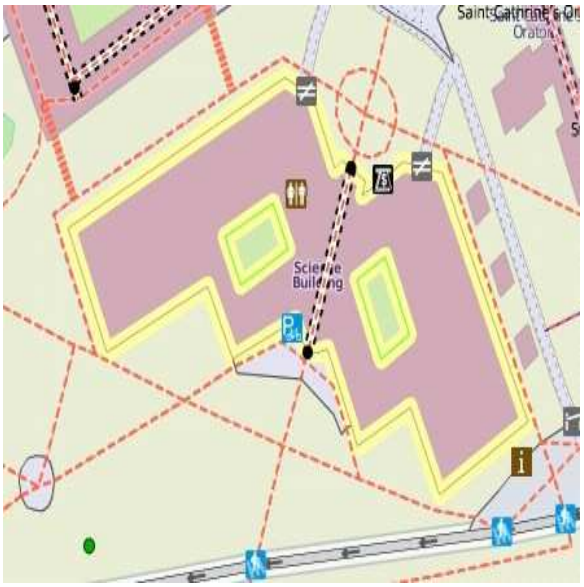

Figure 4.2: This graphic shows the actual rendered image of the OSM data displayed in the Listing 4.1 and subsequent listings to Listing 4.4

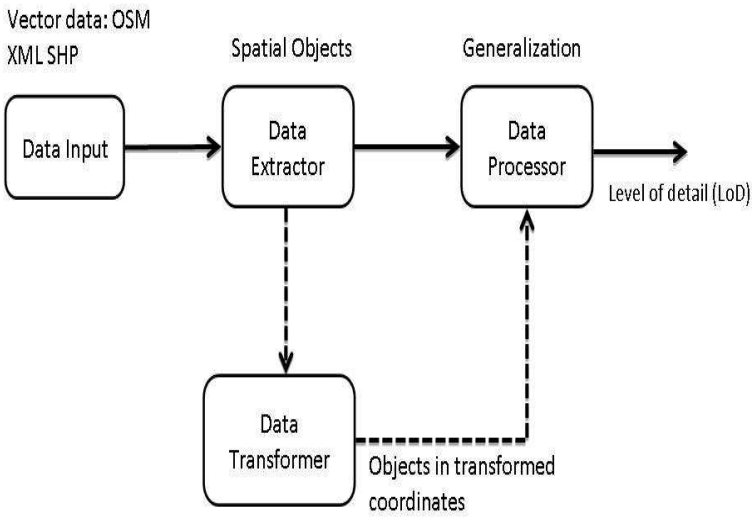

Figure 4.3: This flowchart shows the four principal components in this software tool for processing the vector datasets and implementing the mathematical foundations of the framework described in Chapter 3

the spatial representation in OpenStreetMap. Many contributors to the project avoid creating relations for a number of reasons including poor software support for relation editing and a general lack of understanding amongst contributors in what relations represent.

4.2 Description of components in the software tool

The principal motivation behind the development of a specialised software tool for preparing OSM-XML data (and indeed extending to other vector data formats) is that mobile devices cannot handle large spatial datasets efficiently for on-the-fly visualisation and other spatial functionality. For most mobile devices (which are Location-based Service (LBS) aware) it is still reasonably difficult to efficiently handle such vector based geospatial data expressed in XML while providing quick responses to user requests and efficient rendering. Most vector datasets only have

one scale and it contains all feature data information at the highest quality in the XML file. As outlined in Section 2.2.2, many features are represented with very high resolution. Subsequently, it is necessary to perform pre-processing to extract the user required features for efficient delivery of mobile visualization.

These types of considerations and the concepts from Chapter 3 are integrated into the workflow of the overall framework flowchart which was presented in Figure 1.2 in the Chapter 1.1. In the left side of this figure the workflow must: process the vector data, perform generalisation, organise the generalised data into increasing LoD and prepare these LOD for progressive transmission. Crucially, in terms of processing the XML in networked environments, this software tool can process the OSM XML vector data by file streaming techniques. The software can then apply generalization processing in real time to the data in the stream. As stated in Section 2.2.6, this approach can also be extended to process other vector data in other well known formats such as Shapefiles, Geo-JSON and KML. We describe the software tool in more detail. The four components of this workflow of the software are illustrated in Figure 4.3 on page 83. There are four key components: (1) the data input component, (2) the data extractor, (3) the data processor, and (4) finally the data transporter. These components shall now be described as follows:

1. ***The Data Input Component***: The data input component responds to the user request and takes vector data as input. For the purposes of the testing and case-study, we allowed the specification of a bounding rectangle of geographic coordinates. The user request can be composed as a HTTP GET request for the data from the OSM API. The request is formed as a URL such as `http://www.openstreetmap.org/api/0.6/map?bbox=L,B,R,T` where L and R are the western and eastern sides (longitudes) of the bounding rectangle while B and T are the southern and northern sides (latitudes) of the bounding rectangle. This request is then sent to the OSM API. Rather than downloading the entire file containing the data from the request, this software can process the data in real-time using a streaming of the XML. All polygon features inside this bounding box are streamed to the server and processed. However, this could be easily extended to other geospatial data formats such as ESRI Shapefiles or

imported from geospatial databases (i.e. PostgreSQL) by using GeoTools. The GeoTools package [GeoTools, 2012] is written in Java and enables this software to read a number of different vector data formats such as GPX and ESRI Shapefile. It is an extensive library and has reached de-facto industry standard status.

2. ***The Data Extractor***: We have implemented this generic algorithm to process the streaming vector data as shown in the example algorithm in Algorithm 2 for typical OSM XML data. It takes XML stream input from OSM (line 2) and use *polygonlist* to record the objects and *nodehashmap* to record the nodes. The"polylist" is a list which maintains a collection of polygonal objects, which includes all the spatial information. The "node-hashmap" is a hashmap with a list of ("key", "value") pairs for retrieval of the location information(as "value") of the nodes by using a reference id (as "key"). It iteratively checks the elements in XML and identifies the key elements such as "osm"(line 9 - 11), "node"(line 12 -14) and "way" (line 15 -17) and then extracts the spatial information and adds to the *polygonlist* and *nodehashmap* when it comes to the end of the element (line 19 - 33). This component is based on the Stax XML Java toolkit [Tutorial, 2012]. Stax is an opensource tool which has many advantages including the availability of cursor level access to the OSM XML data. Compared to other XML tools, such as XMLBean, Stax is more efficient in fetching the geospatial data from XML formats [Tutorial, 2012]. Using Stax we can move the cursor pointer forward, skipping to the specified geographic features, and then extracting the spatial objects from XML without requiring large memory consumption. This Java XML processing approach means that this software will run on most Java-enabled servers allowing real time data processing. The geographic objects (nodes, polygons, polylines) are then bound directly to Java objects for a further processing.

3. ***The Data Transporter***: All vector datasets are embedded into a geographic coordinate system. OSM data is presented in WGS84 (Latitude-Longitude) coordinates, so it is necessary for accurate area and distance calculations that the data be transformed to a meter-based coordinate sys-

tem. The UTM ($UniversalTransverseMercator$) is used. The key function of the "Data Transporter" is to translate OSM XML data to the screen resolution coordinate system while maintaining the functionality of spatial calculations (ie. calculate the distance between two locations). This enables the calculation of error metrics by transforming the coordinates between a mobile screen and the OSM data (see more details in Section: 3.4)

4. ***The Data Processor***: In this component we apply the shape analysis metrics discussed in the paper Ying et al. [2010a]. An overall score will be assigned to each spatial object in the vector data and the work of the "Data Processor" is to compute whether the geographic features are ready for direct delivery for LBS applications or whether there is a need to perform further generalization processing. If this score is smaller than an error allowance threshold, then the corresponding spatial object is marked as "LBS-Ready" for an immediate delivery. This threshold definition was explained in Section 3.2. Otherwise, this software checks if there are any topological conflict issues [Ying et al., 2010b] or over-representation issues. If there are issues, it suggests further processing.

Algorithm 2: The procedure of processing a typical OSM XML file

```
1  XMLeventReader initialization and Open a new XML stream;
2  PolygonList ⟵ ∅; nodehashmap ⟵ ∅;
3  while eventReader.hasNext ≠ ∅ do
4      e ← eventReader.nextevent;
5      if e.isStartElement ≠ ∅ then
6          element S ← the start element tag;
7          if S is "osm" then
8              Start to extract spatial objects from element < osm >;
9          end
10         if S is "node" then
11             Read nodes and extract key element: "id","lat" and "lon" from
               element < node >;
12         end
13         if S is "way" then
14             Start to extract spatial objects from element < way > and
               Extract element "nd" and get the location reference from
               nodehashmap;
15         end
16     end
17     if e.isEndElement ≠ ∅ then
18         element E ← the end element tag;
19         if E is "osm" then
20             End the extraction work at < osm>;
21         end
22         if E is "node" then
23             Add the node into the nodehashmap at < node>;
24         end
25         if E is "way" then
26             Add the spatial object into PolygonList at < way>;
27         end
28     end
29 end
```

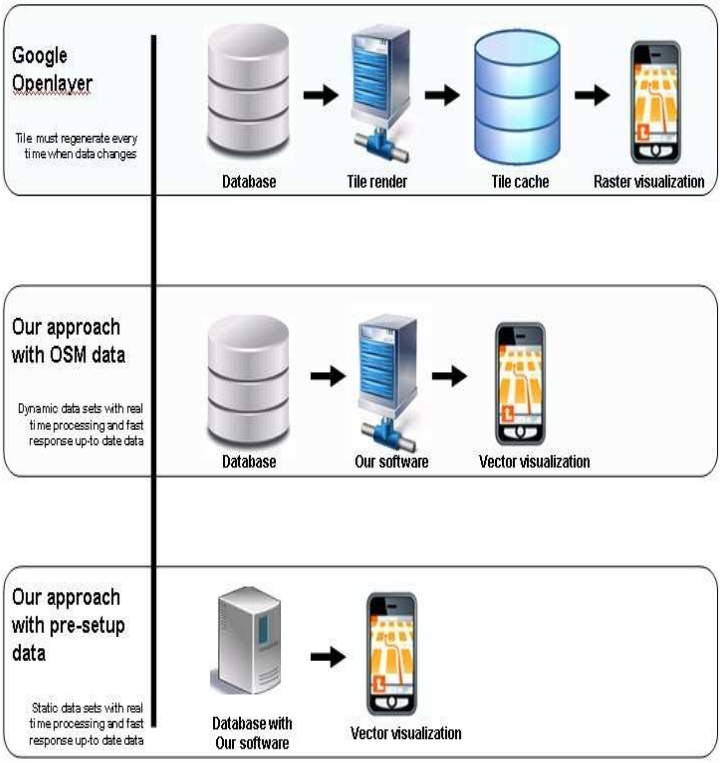

Figure 4.4: Three different approaches for map visualisation. The traditional method of map tiles and raster visualisation is shown at the top. This approach then can be implemented with dynamic or static datasets.

4.3 Key benefits offered by this software tool

Given the framework presented in Chapter 3 , in order to provide a service for the requirements of Progressive Transmission in Chapter 5, we have developed the software tool described in this chapter to the exact specifications of this framework. There is also the advantage that we can then more precisely arrange the parameters of this framework in order to prepare LoD data in a proper form. We have described the computational aspects of the data processing tool, we will outline some of its advantages. Compared to the traditional tile-based mapping systems, such as Google Maps, this approach provides a dynamic processing framework for vector based data using OSM data as the case-study. As shown in Figure 4.4 the software implementation can stream OSM-XML directly from OSM. This data is then processed on the fly. The simplified data is then packaged into prioritised LoD and then transmitted to the client device. The key difference with this approach, as illustrated in Figure 4.4, is that we do not rely on the server to generate raster map tiles. We are not tied to a single representation style or server-rendered zoom levels. Multiple map datasets can be processed and displayed simultaneously and the map can have dynamic styles by performing the rendering process on the mobile client. This will be discussed in greater detail in Chapter 6. This allows us to create more aesthetically pleasing map visualisations with customised appearances for spatial objects depending on their type. Users can dynamically perform queries on the data (if this functionality is made available to them). The visualisation builds smoothly rather than having 'jumps' or 'gaps' as each level of detail is delivered and integrated into the visualisation. The LoDs are computed such that the appearance of the objects does not change drastically between sequential LoDs. All of these advantages combined make this software more flexible for map display on mobile clients than using the traditional approach of raster map tiles. Moreover, because we do not rely on a pre-rendering process of map tiles, this software can handle the dynamic OSM data. This is crucial given the rate at which the OSM database is changing.

Implementation Example

Before the conclusion of this chapter, we provide an implementation example to illustrate the key benefits offered by this software tool. In this example, we are using vector data from OSM and raster map tiles from OSM to highlight the differences of three approaches: (a) the proposed approach implemented in the software tool outlined in this chapter, (b) alternative vector approaches using OSM data sources (such as Cartegan by Cartegan [2010]) and (c) tile-based approaches (which corresponds to the first implementation in Figure 4.4 on page 89). We have not generated the tiles in (c) ourselves. However, we could easily have done this using the Mapnik software toolkit (`http://wiki.openstreetmap.org/wiki/Mapnik`). Mapnik is used by OSM to generate the map tiles for the OSM website and associated map applications. The stylesheets and style configurations used on the OSM website are freely available.

Figure 4.5 shows the three corresponding maps generated by these three approaches using the same map area from OSM. Figure 4.5a is a simplified version of vector map by the proposed approach using the same area as the raster map in Figure 4.5c. As outlined in Table 4.1, the raster approach is provided with the highest resolution (non-simplified) vector OSM dataset. Table 4.1 summarises the differences in these three approaches. It assists us in emphasising the advantages of using our proposed approach for vector data transmission and visualization over the other two approaches. As shown in this table, we use the simplified JSON format for transmission which is smaller than OSM (in the 'Size' row of the table). However, this is more flexible than tiles (in the 'dynamic or static' row of the table). The output map from three approaches shows that the proposed approach can generate appropriate visualization quality. Since the input data to our visualisation is at lower detail in this approach, we benefit from faster transmission and efficient visualization of the maps compared to the high detail input maps in approaches 2 and 3. The proposed approach and approach 2 can stream the spatial data from a dynamic data source and process this data on the fly. This is much more efficient than the generation process of pre-computed tiles. In summary, the proposed approach can offer faster interactions with the maps whilst reducing the amount of data required for the map visualisation. When

users require more details, the additional refinements can be transmitted to them by progressive transmission, which is the technique proposed and explained in the next chapter.

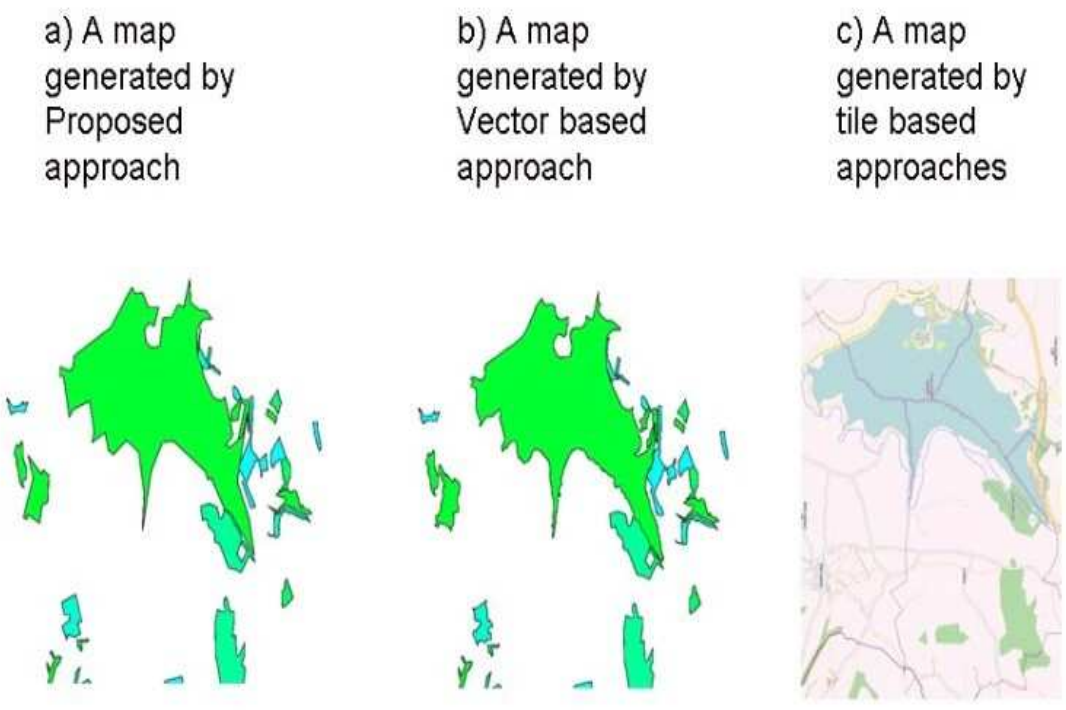

Figure 4.5: This figure provides screenshots of the output of the three types of mapping approaches for the same OSM input dataset. (a) shows the proposed approach; (b) shows an alternative vector approach; (c) shows the output of a raster tile-based approach. Details of these approaches are outlined in Table 4.1

4.4 Chapter summary

This chapter has described a software tool which can process vector-based data efficiently and provide generalized datasets for LBS mobile applications. The focus of our analysis is on vector data collected in the OSM project. As the OSM project manages VGI data, there are problems in the representation of geographic features, which must be addressed before generalization can be performed. This

Map Characteristic	Approach 1	Approach 2	Approach 3
Source of Data	Vector	Vector	Raster
Data Format	JSON (simplified data)	OSM XML	Image tiles
Dynamic or Static	Dynamic	Dynamic	Static
Overall Size	300kb	2mb	120kb (combined size of tile images)
Visualization quality	Good	Good	Good
Detail	Low	High	Standard
Generation Speed	Reasonably fast (2 seconds)	Fast (< 1 second)	Very slow: precomputed in the server, served over Internet quickly
Transmission speed	Fast (< 1 second)	Slow (5 seconds)	Fast (< 1 second)
Rendering speed	Fast (< 1second)	Slower (10seconds)	Fast (< 1second)
Interaction flexibility	Interaction possible	Interaction possible	No interaction possible without additional functionality
Scalable	Yes	Yes	Fixed scale

Table 4.1: This table summarises all of the aspects of computational performance of the three mapping approaches: 1) our proposed approach, 2) other vector approaches - in this case Cartagen, and 3) the standard tile-based approach

chapter has proposed the first step in the server-side framework for dynamic processing of vector-based data. It has also described the preparation of data for progressive transmission in the next chapter. We conclude this chapter by summarising that it is more flexible to process this VGI vector data on the fly. This results in a significant reduction of the data load for transmission while still maintaining a appropriate visual appearance for visualisation of the maps on small mobile display screens. The next chapter will describe the steps necessary to deliver the LoD to the client device for incremental refinement and visualisation in our proposed progressive transmission scheme.

Chapter 5

Progressive transmission of vector-based spatial data

In the previous chapter, we described a framework of processing vector-based spatial data and a subsequent implementation of these in software. In this chapter, we describe the actual progressive transmission of this vector-based spatial data. Increasing LoDsth of a selected dataset are progressively transmitted for incremental refinement and visualisation on the client device. We first describe a progressive transmission scheme to generate LoD for the given OSM data and gradually refine the simplified coarsest map with equivalent increasing LoD. A user study is conducted to survey the user satisfaction with these generated maps at the different resolutions. The study shows that users are generally satisfied with the highly detailed maps but not with those of lower resolution. We then discuss complexity metrics for shape similarity measurements and conduct additional user trials. We attempt to quantify some intuitive measures of dissimilarity in maps from the user's point of view. A clustering method is employed to assist in finding obvious intuitive measures such as area importance and shape complexity. These measures are used as heuristic metrics to identify the dissimilarity of spatial objects when progressively refining the maps for visualisation. To achieve better user perception of these maps, we then propose a selective progressive transmission strategy using these metrics to minimise the "popping effect"(i.e. sudden significant visual changes) during progressive transmission.

5.1 Description of the progressive transmission scheme

In many LBS applications, it is necessary to exchange spatial data over the network and visualize it as soon as possible and with the highest quality [Paolino et al., 2011]. For example, a geoscientist may need to conduct a mapping activity with a mobile device, while viewing and accessing the spatial data. Remote access to high quality spatial data via a mobile network is difficult in such situations due to slow or obstructed network communications and the potentially large data volumes involved. Users usually can view the data only after the entire dataset has been fully downloaded. This is commonly experienced on the Internet when one downloads video or audio files without streaming. The entire file must be successfully downloaded before playback can be performed. In order to mitigate the issue of downloading dataset, a data processing framework was proposed in chapter 3. As one can recall from the previous chapter, over-detailed OSM datasets are selected for further simplification. Users are allowed to access the simplified datasets immediately and perform interactive spatial operations directly on the spatial objects in the dataset. However, a coarser version of spatial datasets produced by this process may not be sufficient for detailed viewing, for instance, when the user zooms into a specific area and wants to obtains more detail for that area. The simplified datasets might not be of sufficient detail to provide the fine resolution required for many spatial reasoning tasks. Consequently, a progressive transmission scheme is described to gradually refine the map dataset by transmitting incremental LoD refinements until an acceptable detail is reached. The users can wait until the final LoD has been delivered (full resolution) or may choose to stop the progressive transmission process at any given intermediate LoD.

It is necessary to first transmit the most significant map details, due to constraints imposed by screen resolution and network bandwidth. Subsequent transmissions can transmit additional detail until the entire spatial dataset has been delivered successfully. Progressive transmission has many advantages including the transmission of smaller data sizes, quicker response times, and possibly the transmission of only most relevant details [Bertolotto, 2007]. Progressive transmission approaches generally provide users with a map at a coarser LoD initially.

This map is then iteratively refined through sequential transmission and the integration of further data [Augusto et al., 2009; Bertolotto and Egenhofer, 1999, 2001; Zhou and Bertolotto, 2004]. The user can immediately perform analysis using the coarser map and terminate further downloads when the map approaches a desired LoD. The maps of varying LoD, received by the user, are generally created using map generalization methods. Some results in Zhou et al. [2005] indicate that the progressive transmission strategy provides users with faster access to the initial datasets and obtain LoD within a reasonable time period.

In this work, the constraints of mobile applications must be specially considered and integrated into this progressive transmission scheme. The system architecture follows a classical client-server architecture. The server side manages the data processing and data access computation. This includes LoD data retrieval and management of dynamic data sources. The client manages the data visualization process. This includes communication with the server and performing map rendering. In this chapter, we focus only on the management of the progressive transmission of required LoD refinements for effective access to the spatial data. The main goal of this work is to design a framework which delivers the optimum LoD to a user within the limited bandwidth. The framework is shown in Figure 5.1 which illustrates the key steps in the progressive transmission process. An example of the progressively transmitted data, rendered as map visualisations, is then shown in Figure 5.2. Figure 5.1 operates with the following steps.

- When the user interacts with a map and selects a geographic area, this generates a request using screen parameters which is sent to the software running on the designated database/spatial data server. This is in turn transformed into a data request to the original vector data server. In the experiments, we performed this in the form of an OSM API request direct to the OpenStreetMap servers. However, this is easily changed to a request to a local database, a network data store, or indeed a locally mirrored copy of the OpenStreetMap database.

- When the request has completed, a vector dataset is downloaded. This software takes the vector data as input and performs several steps to identify

and simplify the over-detailed features. Then the LoDs from L_0 to L_N are generated and handled by a tree-based data structure as described in Chapter 3.

- We then initiate the data management strategy to select the most suitable LoDs within a group of polygonal objects which are visible in the user's viewable area. These objects are marked (or qualified) as delivery candidates. The potential for a limited bandwidth network is considered. A limited number of LoD datasets are selected for inclusion in each subsequent data package transmission.

- The application software on the server begins the transmission process from L_0 which contains the most generalised and lowest level of detail. When L_0 has been transmitted successfully, the nodes (and polygon structure information) for L_1 are prepared for transmission. This process is repeated until the final nodes of the original dataset are transmitted with L_N. GeoJSON objects are generated for each L_i.

- The nodes transmitted during L_0 can be considered to be the most significant nodes or vertices in the overall shape structure of the polygons within the original dataset. The nodes transmitted during L_N can be considered the most insignificant nodes. The user may decide that they are satisfied with the map representation at one of the map levels L_j and can choose to stop transmission; Otherwise, the full detail maps will be sent to users. If they choose to terminate the transmission, the visualisation remains on the mobile device screen as the data up to this L_j now resides on the mobile client.

While there is a great merit in the development and implementation of a progressive transmission scheme, the crucial yardstick to measure the success of this scheme is related to the usability of the maps and how they are evaluated by potential users. In the next section, we will discuss the issue of the usability of the maps generated by a progressive transmission approach.

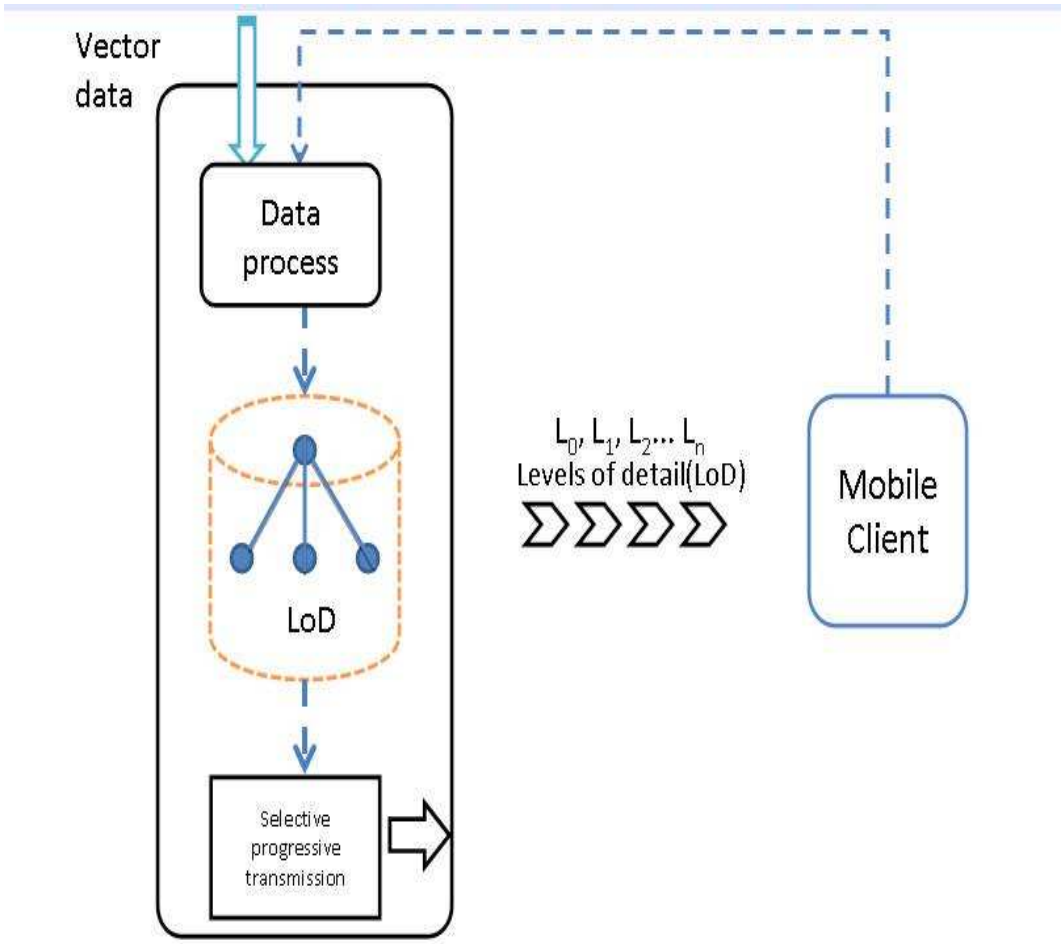

Figure 5.1: The client sever architecture is illustrated. The processing steps employed for the input OSM data is shown on the server (left). The increasing LoD are then prepared and transmitted to the client (right) where a thread-based approach to visualisation and rendering of the data is applied.

Figure 5.2: This is an example of the framework progressively generating LoD (most simplified coarse version at the left and highest resolution to the right). Users can stop receiving the incremental LoD when they are satisfied with the current map representation presented to them

5.2 Map usability issues in progressive maps

Regardless of what computation and data processing happens behind the scenes, one of the keys to the success of any client application is whether the user is able to use the application and that the application is fit-for-purpose. As Teo et al. [2003] remarks, it is now very important to deliver applications which meet the expectations of user satisfaction, as this is now critical "in the competitive market place". The software tool can generalize the map data and generate LoD by applying simplification at different reduction tolerances. However, this introduces the issue of selecting the proper tolerance values to apply. An incorrect tolerance value might result in a very poor generalization. Normally users must decide on a suitable tolerance on a case-by-case basis or from their own experience as very little automation of this decision-making is available. This may lead to unstable results. For instance, if users assign a tolerance to the current map and this produces an unrecognisable map it can mean that there will be no significant visual improvement in receiving the next set of points. The generated map is neither usable nor is it acceptable for its intended purpose. Jenks [1981] demonstrated that reductions of less than 40% of the total number of nodes in a set of objects does not produce a significant visual impact. Therefore, when implementing a progressive transmission strategy, it is important to perform generalization to obtain output maps, which are suitable and exhibit good "map usability". As outlined by Paolino et al. [2011], users usually match their perception of the real world with the displays on the map to affirm their current positions or their

understanding of the geography of the regions.

Since map services are increasingly used in LBS for information presentation, finding and using the best means of adapting maps for a small screen is an important research challenge. Methods from the domain of computer vision may be used to perform map simplification in an evolutionary manner, which preserves a contour's overall shape across LoD as mentioned in Chapter 3. However, such a local criteria for a single map feature cannot be used to determine a suitable evolution for a collection of map features. For maps containing multi-polygonal objects, a methodology for determining a globally suitable generalization is required. The display capabilities of mobile devices are more restrictive than the traditional desktop machine. This is a major difference and must be kept in mind during the development of mobile applications.

Sester and Brenner [2004] proposed an approach for the generalization of vector data tailored specifically for mobile devices with a limited screen resolution. Kjeldskov and Graham [2003] stated that more research is required to understand usability from a user perspective as many assumptions are made by researchers on what users actually want from map-based displays. The visualization of multi-scale maps with multiple objects on mobile devices should also exhibit better quality with regarding to user perception.

The problems of the map display on mobile devices are exacerbated by the nature of spatial data where a large information space needs to be presented and manipulated on a small screen [Tonder and Wesson, 2009]. Ying et al. [2011] carried out trials of maps generated by progressive transmission. They found that maps with moderate LoD produced the best map quality and highest user evaluations. In map usability tests carried out by Kratz et al. [2010] the authors found that users preferred map zooming to map panning and scrolling. The authors comment that tasks such as map navigation require users to view and classify larger features of the map (in a zoomed-out state) in order to locate the point of interest they are interested in. Harrie and Stigmar [2010] provided some measures of map information that eventually should be used as constraints for the selection of data layers and in real-time generalisation. They found that measures such as the number of objects, number of points, and object line length had better correspondence with human judgement than object area alone. Paolino et al.

[2011] also conducted an empirical user study to evaluate user object recognition on progressive maps with different LoDs. Yang et al. [2011] proposed a model to handle the multi-resolution visualization of buildings in urban environments when considering the thresholds sequence in LoDs.

Overall most scholars in this domain agree that maps must be adapted for mobile screen display while considering potentially limited network bandwidth. They cannot be a straightforward extension of map-based applications for desktop machines and screens. As an increasingly large number of map users are accessing cartographic products through mobile and position-enabled devices, researchers have shown that users appear to find maps with less, rather than more, information to be useful and usable. The user satisfaction with different aspects of the map is important to analyse the usability of the map. This leads us to the next section, where we describe some empirical studies that we carried out to assess the satisfaction of users with maps generated by the progressive transmission strategy.

5.3 User satisfaction with maps generated by the progressive transmission scheme

In the area of geographic information management, simplification of lines is a generalization process that aims to eliminate unnecessary map details [Fatto et al., 2008]. As they stated, the simplification process is currently addressed as a solution to several research issues in the field of spatial data management such as: faster map transmission, plotting time reduction, storage space reduction and faster data processing. When users are not satisfied with the initial simplified maps, further incremental LoD should be sent to refine the coarsest version of maps until the user is satisfied with the map representation. He argues that this task is more complex than simplifying map data. Any LoD will cause the changes which can affect the overall map visual perception as each subsequent LoD adds more data and information to the map. Selecting the most suitable LoD is a difficult task, but yet one which is important to the overall success of approaches. In this section, we will outline a number of user trials we performed

where several participants were asked to evaluate their user satisfaction with the LoD in maps generated by the progressive transmission scheme. The goal of this experimental analysis is to understand the usability issues related to the use of LoD. It attempts to find if there are differences in different users' perception of LoD. This analysis could help us to quantify a suitable solution for transmitting specific LoD in a user preferred manner.

5.3.1 Experimental design

We designed an empirical study aimed at determining if there was a relationship between the data reduction and the user satisfaction for given LoD. We used the OSM datasets as this case study. In order to test user satisfaction with these progressively generated maps, we recruited 10 users to take part in this user trial. All users had a GIS background. They were given an explanation of the progressive transmission process and an outline of how the experiment was designed. We ran the progressive transmission software for 10 input datasets. The users were provided with datasets with 5 pre-determined distinct levels:

- $L5$ the full version with no reduction in data size,
- $L4$ the original reduced by 20% (called $r20\%$),
- $L3$ a further 20% reduction (called $r40\%$),
- $L2$ a further 20% reduction (called $r60\%$),
- $L1$ a final 20%, which represented an 80% reduction (called $r80\%$) on the original map dataset.

These LoDs are generated by reversing the order of our simplification process. The map datasets used in the trials were chosen as follows. We chose 10 case-study areas in OSM for map datasets as shown in the Table 5.1. In maps with multi-objects, all objects will have an equivalent level of detail. The scale is set at $1 : 50000$. To carry out this evaluation of this model, we omitted line features. The 10 datasets selected include single polygon datasets and multi-polygon datasets. There are 4 datasets with one polygon: $c42$ (211 nodes), $d14$

(309 nodes), $r16$ (46 nodes), $r50$ (51 nodes). The remaining 6 datasets are multi-polygon and have between 277 nodes ($n6$) and 2172 nodes ($n7$). An example of this progressive transmission process applied to the $N3$ map dataset is shown in Figure 5.4.

A mobile phone running the Android OS was used. The delivery of the geospatial data to the rendering application on the client mobile device occurred in the order L_1 through to L_5. As recommended by Li and Nakano [2007] the screen display was kept simple and uncluttered. A screenshot of one of the map displays on the Android device is shown in Figure 5.3. Like the polygon in this figure, we selected test data with minimal overlap between polygonal objects and also removed the effect of color on map recognition. The test started with the user clicking the maps on the mobile screen. The size of the maximum bounding rectangle from which the map datasets were generated was limited in our study to 2 square kilometres. For urban areas in OSM, this can contain a large volume of data [Mooney and Corcoran, 2012a].

Results of the user trials were collected as follows. The users can click through a series of options to display the 10 map datasets and fill a questionnaire on paper to express their satisfaction. A ten point Likert scale [Matell, 1971] was used in the questionnaire given to each subject with answers ranging from (1), which corresponds to a "strong dissatisfaction with map", to (10) which corresponds to a "strong satisfaction with map". This broad scale provides users with a wide range of possible answers to correctly express their opinion. Table 5.2 (on page 106) shows the mean user scores for each progressively generated map for each dataset while Table 5.3 (on page 108) shows the corresponding standard deviation. Figure 5.5 summarises all of the results of the survey for all users. The legend indicates the rating that the users gave to each map. The downward trend (decreasing levels of satisfaction) with $R60$ and $R80$ is evident from this graphic.

5.3.2 Discussion of results

In the experiments, we attempted to quantify how satisfied users were with the progressively generated maps. We also attempted to quantify if they were also

Figure 5.3: This is a screenshot of the mobile device interface upon which user trials were performed. This is the Android operating system

Dataset Name	Object Description	Nodes
c42	Single Polygon	211
d14	Single Polygon	309
n1	Multi-polygon	825
n2	Multi-polygon	1631
n3	Multi-polygon	278
n4	Multi-polygon	1351
n6	Multi-polygon	277
n7	Multi-polygon	2172
r16	Single Polygon	46
r50	Single Polygon	51

Table 5.1: The table details of the ten study areas used in the trials. The first column represents the name of the study area. The second column specifies whether the study area contains a single or multiple polygons. The third column represents the total number of polygon and line vertices in the study area. For example, the Figure 5.4 corresponds to example code n3 in this table

Set	Final	R20	R40	R60	R80
C42	8.5	7.6	7.9	4.4	3
D14	8.6	7.2	5.3	3.9	3
N1	8.6	8.	5 6.5	4.1	4.2
N2	8.2	7.3	5.9	3.8	3.2
N3	8	7.7	5.9	3.7	3.2
N4	7.9	8	8.2	8.1	8.1
N6	6.3	6.2	5.1	3.9	3.1
N7	7.9	7.9	8	6.4	3.9
R16	6.1	6.2	4.9	3	2.2
R50	6	4.7	4.1	3.2	2.3

Table 5.2: Users were asked to rate their satisfaction with the presented map at a given Level of Detail. The ratings were on the scale of 1 (strong dissatisfaction with map) to 10 (strong satisfaction with map). This table shows the mean of user scores for each LoD and the final version of the map.

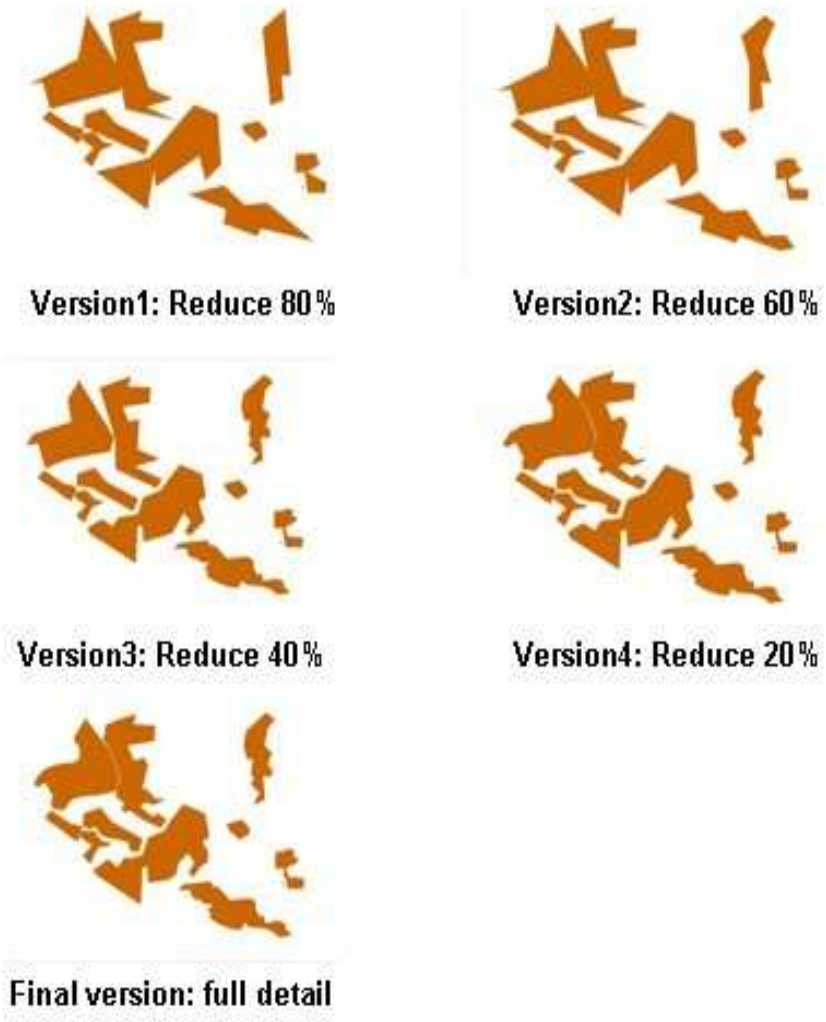

Figure 5.4: An example of the progressive transmission process where the user views five intermediate versions or LoD of the same map.

Set	Final	R20	R40	R60	R80
C42	1.27	1.9	1.37	1.58	0.94
D14	1.07	1.23	2.06	1.85	1.15
N1	0.97	1.18	1.35	1.45	2.04
N2	1.03	2.11	0.32	1.69	1.14
N3	1.25	0.48	0.99	1.7	1.99
N4	2.02	1.83	1.75	1.91	1.91
N6	2.26	2.04	1.2	1.37	2.08
N7	0.74	0.57	0.67	1.96	1.52
R16	1.52	1.75	1.37	1.83	1.14
R50	1.83	1.89	1.66	1.87	2.06

Table 5.3: Users were asked to rate their satisfaction with the presented map at a given Level of Detail. The ratings were on the scale of 1 (strong dissatisfaction with map) to 10 (strong satisfaction with map). This table shows the standard deviation of user scores for each LoD and the final version of the map.

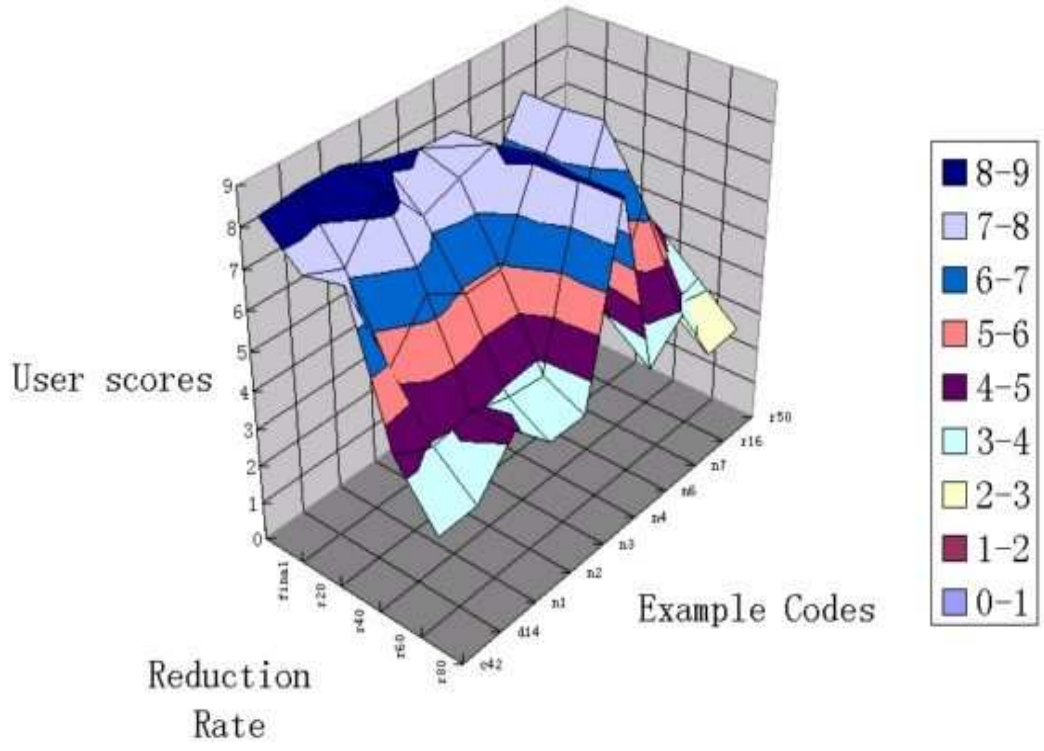

Figure 5.5: This chart shows the user satisfaction scores over the five LoD of the progressively transmitted maps for the ten study areas

satisfied with the simplified data and what the difference was between individual user scores for the same map at the same levels. From the experimental results, we concluded our analysis as follows :

- **Result 1**. In the Figure 5.5 and the corresponding table in Table 5.2 the results showed that user satisfaction has a non-linear positive relationship with data reduction size in general. Satisfaction only diminishes slightly when the data is reduced by 20% or 40% (as expected from work such as [Jenks, 1981])(i.e. $n1$ and $d14$ in the table). However, satisfaction reduces sharply for the further simplified versions (60% and 80%) for most cases.

- **Analysis 1**. The sharply decreasing users scores in the 60% and 80% reduced LoD maps indicate that some of the most simplified versions are not looked upon favourably by users whilst highly detailed maps are preferred by users in general. There are shapes which are under-represented and can cause significant "popping effects" between levels, such as the example $n1$ in Figure 5.6 demonstrates. This might be the source of the disagreements amongst individual users in their satisfaction with these particular maps

- **Result 2.** Some over-represented examples (i.e. $N4$ with 1351 nodes in the table Figure 5.1) exhibit greater user satisfaction at greater data reduction percentages while under-represented map examples (i.e. $R16$ with 46 nodes in the Table 5.1) represent low user satisfaction over the maps with higher simplification rates (60% and 80%). When one calculates other metrics for the dataset and the study areas, we can gain further insights from the results. Datasets $r16$ or $r50$ have $Kmean = 0.413$ and 0.468 respectively (referring to the definition of $Kmean$ in Section 3.2 on 51), but also have lower user satisfaction levels and are considered as under-represented maps.

- **Analysis 2.** Users do not recognize additional refinements to the map during the progressive transmission even after a sufficiently good shape representation has been obtained as the example $N4$ in Figure 5.7 indicates. The results indicate that if a polygon is over-represented, simplification is possible to reduce the data size before transmission to a mobile device with limited capabilities.

- **Result 3.** The satisfaction amongst users of the single polygon examples ($D14$, $R16$) is very positive for the full version and $r20$, but quickly degrades for other versions. It can be seen in the standard deviation table (Table 5.3) of user satisfaction scores that most users agree with each other about the score (with a low standard deviation of approximately 1 for $R80$ for example). However, we find that in the multi-polygon maps such as ($N2$, $N7$), the users have expressed greater satisfaction with the low LoDs as seen in the Figure 5.5 and the Table 5.2.

- **Analysis 3.** For the map represented by a single polygon object, it may be easier for users to observe the difference when progressive updates are applied to map visualisation with LoDs. However, in the situation of multi-polygon objects, the changes of a single object or a few can be less obvious with regarding to the overall map characteristics. Users may not easily identify the difference when being shown a series of similar progressive LoDs. This is an interesting problem, but somewhat outside the scope of our research. Identification of subtle differences between a series of similar progressive LoDs may be heavily influenced by users' spatial reasoning and cartographic understanding skills.

This study increased the need to find suitable solutions to the issue of providing smooth transmission of LoD whilst attempting to ensure that users are satisfied with the results. We investigated single polygon examples. However, almost exclusively, in reality the map usually consists of a set of polygons. It is often the case that there are variations in representation amongst the polygons/polylines in VGI data. For VGI datasets such as OSM, this is certainly the case. This brings us to the position where we can introduce the problem for the next chapter. The experimentation above considered a rigid selection of LoD (at the specific percentage reductions in spatial detail). In practice 'jumps' from the original map datasets $L5$ to $L4$ (with 20% of detail removed) are not common. However, depending on the representation of the objects in the datasets the move between intermediate levels between $L5$ and $L4$ could display some 'popping' effects or significant visual changes. How do we choose suitable LoD to maintain a smooth transmission effect and keep map changes under control? Motivated by

this we develop an enhanced strategy called "selective progressive transmission" in the next section to address this issue.

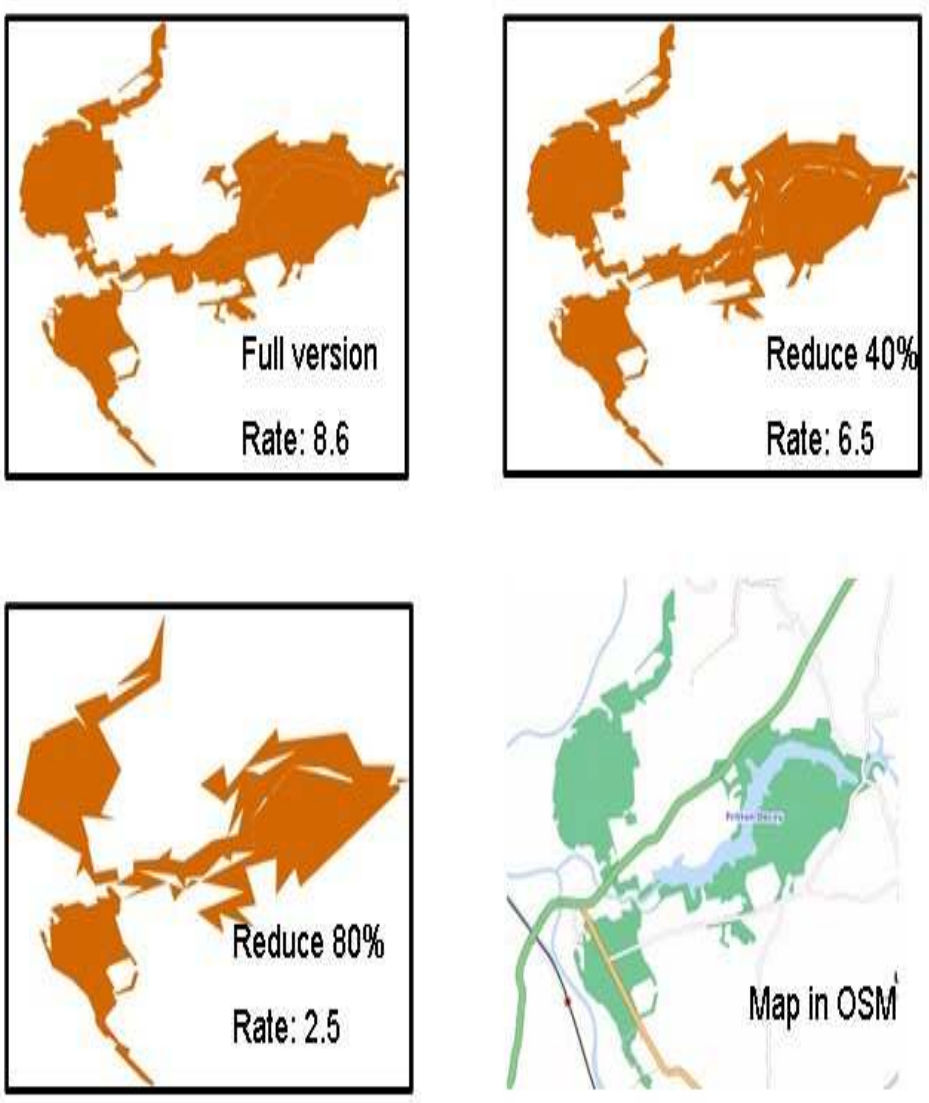

Figure 5.6: For under-represented examples the users did not like over-simplified maps (with 80% reduced data). Significant "popping" effects are easily seen between the LoD chosen for the experiments here

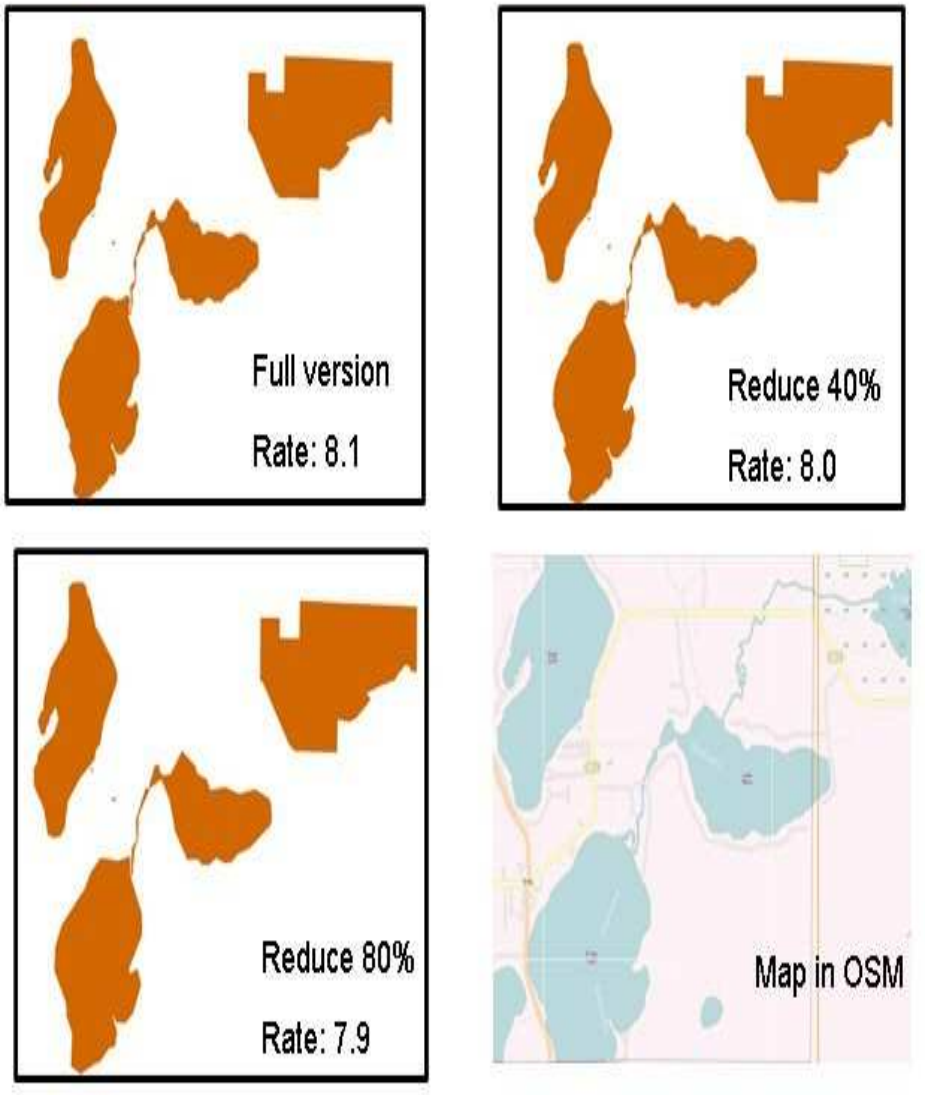

Figure 5.7: For over-represented examples, the user satisfaction ratings show similar satisfaction opinions over the progressive transmission. In this example, the mean user score was high for three different LoD

5.4 Selective progressive transmission

Map visualisations help users to explore the spatial information in a geographic region. Transmitting high quality map data to mobile devices constrained by the low bandwidth network is difficult. When sending this map to mobile users in increasing LoDs by using progressive transmission, users may not be satisfied with the simplified versions. We have seen results of this in the experimental analysis in the previous section. In a dataset such as OSM, as we discussed in Section 2.2.2, there are issues of non-homogeneous representation of the geographic features [Mooney et al., 2010]. In a small geographic area there could well be some features which are over-represented and other features which are under-represented. If we consider simplification and then subsequent rebuilding of these features, the over-represented objects will retain their shape characteristics (before and after) whilst any detail removed from under-represented features is replaced quickly.

To obtain the best quality map representation when user are interacting with LBS applications, it is necessary to select the most suitable candidate objects for an immediate delivery and visualization of the maps on client devices. The non-selective progressive transmission strategy introduced earlier is simply a reversal of generalisation and simplification. This is reasonably straightforward to understand and it works well in our implementation. However, it considers that all objects are similar (in characteristics and user perception of these objects). This is not an accurate reflection of cartographic or geographic reality. In this section, we introduce an extension to this progressive transmission scheme called 'selective progressive transmission'.

We first perform another user survey of the OSM maps by assessing the general complexity of the map objects contained. We then develop a set of metrics to quantify the dissimilarity of map objects. These perceptual metrics are used to derive our selection strategy which is used to adjust LoDs adaptively among complex multi-polygons. This will help to maintain a uniform data rate in transmission and will provide a smoother progressive transmission. 'Complex' objects will be given priority over non-complex objects. Higher priority objects will be supplied with spatial detail in LoD earlier than non-complex objects.

5.4.1 Preparing for selective progressive transmission

Transmitting large-scale LoDs for each requested scale is time consuming. From a user's perspective, it is not always necessary if zoom and pan interactions are needed, for example, to navigate to an area of interest (AOI). As this task does not require a map at the highest resolution, it is reasonable to send less detailed maps first. In order to define these representations, such that characteristic features are preserved, automatic guidance methods are needed. During a line simplification process, the user's perception of differences may change remarkably [Paolino et al., 2011]. To obtain smooth progressive maps, a simplification algorithm is needed to select a subset of vertices which best represents the geometric properties of a polyline while eliminating the remaining points [Fatto et al., 2008]. The generated LoD from the simplification should maintain some optimality between successive levels of resolution [Marteau and Menier, 2009].

The idea of continuously refining low-resolution vector data in Internet and mobile applications has been discussed by some researchers during the last decade. Buttenfield [2002] described a method to refine the maps based on the DP algorithm which are topologically consistent. Ai et al. [2005] proposed a "changes accumulation model" to describe the incremental details as additional change patches are applied to successive LoDs. In this way, as Ai et al. explains, how the difference between each scale can be described as an accumulation of changes and could be easily reversed. Follin et al. [2005b] built a multi-resolution model by applying metric and topological operators. These maps usually consisted of several polygonal objects. However, given a certain number of vertices to refine the simplified version of this map, one cannot simply reverse the process of simplification to each polygonal object, as the representation quality is variable for different spatial objects in an area. There are many scenarios encountered with inhomogeneous representations found in datasets such as OSM.

In this section, we describe how we have integrated a measure of the polygon shape complexity of features into the progressive transmission scheme. This is extended to selective progressive transmission. Features which are designated as 'complex' are 'selected' and provided with more detail as the increasing LoD are assembled in sequence. 'Simple' objects (usually those which are over-represented

or those represented by simple shapes from Euclidean geometry such as circles, squares, rectangles, etc) are not selected early in the LoD process. These 'simple' objects are only provided with additional detail later in the sequence of LoD. Before introducing formal mathematical notation for measurements in the shape complexity, we describe a user trial to understand a map-user's perception of 'complex' and 'simple' objects.

5.4.2 Human perception of shape complexity

In this section, we will discuss a small experiment we carried out to assess map-users' perceptions of 'complex' and 'simple' objects. The results of a user study on the shape complexity features of OSM datasets are discussed. The idea behind this experimentation is to lay the foundations for using shape complexity as a dissimilarity measurement to describe the changes of shapes features during the progressive transmission process.

Human visual perception can identify complex polygons very quickly when presented with a cartographic representation of the polygons [Corcoran et al., 2010]. More generally, the human eyes can determine that a shape is either "simple" (i.e. a circle, a triangle, a rectangle) or "complex" (a multi vertex polygon) quickly. When we look at buildings without any other information except our visual perception, we can say 'that is a very simple design' (for maybe a bungalow) whilst 'that is a very complex and intricate building') (for a medieval castle or modern office block).

We designed the experiment by providing users with a specially selected set of 65 individual polygons from OpenStreetMap. These 65 were drawn randomly from a larger set of individual polygons using a self-weighting sample approach EPSEM (Equal Probability of Selection Method). This was then used as a ground-truth training set required to establish rules on how to automatically distinguish shapes based on the characteristic of being "simple" or "complex". Some examples are shown in Figure 5.8. The polygons are taken from Open-StreetMap for Ireland and Wales. This set contains a good overall distribution of different types of polygons representing different natural phenomenon. As stated the final set of polygons P were chosen from a larger set of polygons Q. The set

Q comprises all qualifying polygons in Ireland and Wales. For a polygon to be a member of set Q, it must represent a water feature (lakes, ponds, reservoirs), a natural feature (forests, parks, open space, green areas), or a mixture of these two features (reclaimed land, quarries) and it must have an area greater than $0.25km^2$.

Ten people were chosen as test subject participants and were required to indicate their visual evaluation of the shape. The ten people chosen for this task comprised of: six researchers in the current research group, another person who is an expert in GIS, and finally three people with no GIS or IT backgrounds. All participants were shown the 65 shapes in the same order on a smart-phone and were asked to indicate if they thought the shape was "simple" or "complex". When all ten people had taken part in the experiment, the majority vote for each polygon was taken. So for each polygon p_i in the set of polygons P a value of 1 indicated if p_i was deemed "complex" otherwise a value of 0 indicated that p_i was "simple". In the 65 shapes, 34 were voted as "complex" while 31 were voted as "simple".

Using the results of the user trials, we can apply the shape metrics to the "complex" and "simple" objects and try to cluster the shapes based on the values of various shape complexity metrics. We will describe some mathematical formulations of shape complexity, which can be implemented computationally in the next section.

5.4.3 Automation of shape complexity assessment

In the previous work [Ying et al., 2010a], we implemented a number of shape description measures (related to characterstics such as Area, Perimeter, Number of nodes and so on) in an attempt to identify the polygons with specific features for use as input for testing generalisation algorithms. The shape complexity measures we used attempted to calculate characteristics for each polygon. These quantitative characteristics can then be used to give an overall measurement of the complexity of a polygon. This allows automatically description of the complexity of a polygon without human visual evaluation [Rosin and Mumford, 2006]. Obviously, the process is quicker if we do not have to rely on human visual

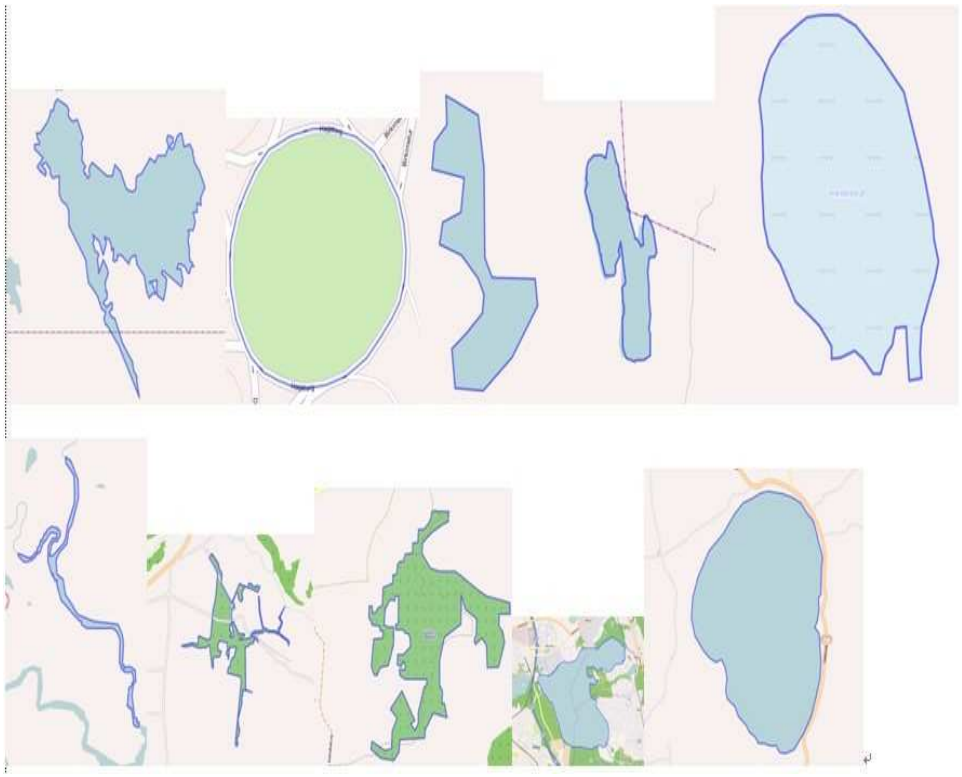

Figure 5.8: Examples of the polygons taken from OSM which were used in the user trials. For the 65 shapes 34 were voted as “complex” while 31 were voted as “simple” by our user participants

evaluation. As all polygons in the training set represent real-world geographic objects, it is important to focus on shape description measures which exploit the spatial characteristics of the shape itself [Brinkhoff et al., 1995]. Each polygon p in the set of polygons P extracted from the OSM XML has n vertices labelled $0...(n-1)$. A number of shape complexity measures are calculated for each polygon p_i in the test set within which we will outline each of the complexity measures. This approach attempts to compute a measure of the polygon complexity in the smallest number of steps which are outlined as follows. For these steps, we choose Circularity C_p and Area Ratio A_r.

Circularity C_p is important shape characteristic [Brinkhoff et al., 1995; Žunić and Hirota, 2008]. It is very well known and is easy to calculate. The value of C_p is normalized to the range of $[0 \dots 1]$. The equation for C_p is given as follows:

$$C_p = \frac{4\pi \times A_p}{P_r} \tag{5.1}$$

where A_p is the area of the polygon shape and P_p is the perimeter.

The second important shape complexity metric we chose was the well known Area Ratio A_r. A_r is a little more difficult to compute as we need to have knowledge of convex hull of the polygon p. The convex hull of a set of points (or polygon) in N dimensions is the intersection of all convex sets containing the polygon. In $2-D$ geometry the convex hull is easily interpreted as the smallest convex set containing that polygon object. One way to think of the convex hull is as follows: stretch an elastic band to make a giant circle, so big that the entire polygon fits inside. Now allow the elastic band to shrink until it just contains the polygon. The outline of the elastic band marks the boundary of the convex hull [Costa and Cesar, 2000]. A_r investigates the relationship between the size of a polygon and the size of its convex hull and is expressed as a ratio:

$$A_r = \frac{A_{cp} - A_p}{A_{cp}} \tag{5.2}$$

where A_cp is the area of convex hull of the polygon and A_p is the area of polygon. The convex hull of a polygon can be computed using several different approaches [Chen, 1989]. We used the Quick Hull algorithm. In Figure 5.7 (on page 112)

a "complex" shape is displayed. The area ratio for this shape is 0.442 while the circularity measure is 0.092. The area ratio measure is high indicating that the convex hull has a very large area in comparison to the area of the polygon itself. The circularity value is almost 0 indicating that the polygon has no circular characteristics.

In section 5.4.2, we described the results of a visual analysis of 65 polygons. The values of Circularity C_p and Area Ratio A_r corresponding to the polygons in the user trials were computed and these values are displayed in the scatter plot in Figure 5.9.

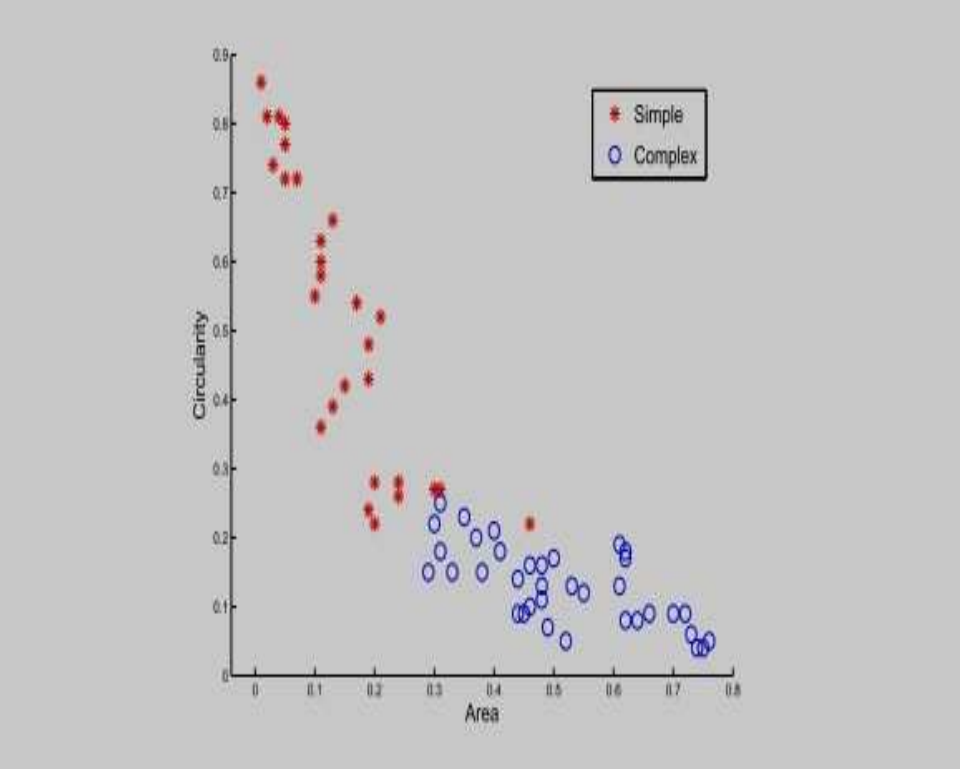

Figure 5.9: Results from the test-set of polygons are plotted (Circularity C_p and Area Ratio A_r). Two distinct clusters of polygons are evident. We have labelled the polygons as classified by the user trials with different colours representing simple and complex shapes.

In Figure 5.9, the polygons in P that were classified as "Simple" by the ten test participants are shown with an asterisks while polygons in P that were clas-

sified as "Complex" are shown with a circle. Two clear clusters are evident from the datasets. The combination of A_r and C_p provides us with an accurate classification of how our human test participants classified the polygons into "complex" and "simple". Consequently we can use these results as a ground truth dataset. Using the two clusters formed here, we classified a larger set of different test polygons using a nearest neighbour classification algorithm [Zhang et al., 2006]. The means of both clusters in Figure 5.9 are taken: for "simple" polygons this is $({x^s}_i, {y^s}_i)$ and for complex polygons this is $({x^c}_i, {y^c}_i)$. Then, for some unknown polygon k with (Circularity C_p and Area Ratio A_r) values (x_k,y_k) we used the Euclidean distance formula to estimate which cluster mean that (x_k,y_k) is closest to. On the test set P, this algorithm achieved classification accuracy of over 90% (the p value for statistical significance is $p < 0.05$) which is more than sufficient for the requirements of this project. Similar results are reported in the work of Brinkhoff et al. [1995].

5.4.4 Using shape complexity for decision making during simplification

Many approaches have been proposed to measure the dissimilarity between shapes. Rosin [1997] surveyed many polygon approximation techniques and used the measure of integral square error(ISE) to measure the difference of the original polygon and approximating polygon. He also showed that when adding more detail to a shape it makes the shape much smoother and closer to the original shape feature. [Cheung and Shi, 2004] has proposed a model to handle positional uncertainty in the process of line simplification by using a shape dissimilarity measure. This approach uses the sum of the length of the vertices and the turning angle as a parameter to measure the changes to the shape. Similar approaches can be seen in work such as Latecki and Lakamper [1999] who use an approach called tangent space measurement.In this approach, we are using the results directly from our experiments to model the metrics of user perception, which are designed for the recognition of map representation and changes in those representations.

When one of our test OSM datasets k is submitted to the simplification process it generates a sequence of LoD $\{ l_0 , l_1, \ldots, l_n \}$. We use the complexity (c_i,a_i)

(where c_i= Circularity C_p and a_i = Area Ratio A_r) to measure the complexity of the LoD after the i^{th} simplification step where $n - i \gg 3$ and $i < n$. We then use the distance $Dist$ to measure the dissimilarity between the LoD l_i and l_{i-1} with the pair of (c_i,a_i) and the original shape in l_n with (c_n,a_n). We label this as the cost distance for the refinements. It gives us a measure of how the overall complexity of the shape is changing between sequential levels of details. The dissimilarity function $Dist1_i$ of the LoDs LoD_i and the original map LoD_n is formulated as follows as a Euclidean distance metric:

$$Dist1_i = \sqrt{(x_i - x_n)^2 + (y_i - y_n)^2} \tag{5.3}$$

Brinkhoff et al. [1995]'s complexity measurements are also used where the composite expression for complexity is $Comp$: Brinkhoff et al.'s complexity measurement is established also using ground-truth datasets and the computation of $Comp$ is reasonably straightforward.

$$Comp = 0.8 \times P_r \times N_r + 0.2 \times A_r \tag{5.4}$$

where P_r is Perimeter ratio, which is defined as $P_r = \frac{P_a - P_c}{P_a}$ where P_a is the perimeter for the polygon and P_c is the perimeter for the convex hull of this polygon. A_r is the Area ratio as in the previous subsection. N_r is the Notch ratio which was introduced by Brinkhoff et al. after extensive empirical testing. A notch is any non-convex part of a polygon. N_r is given as follows: $N_r = 16 \times (N - 0.5) \times 4 - 8 \times (N - 0.5) \times 2 + 1$. where N is the normalized notch number and is expressed as $N = \frac{notch}{vertices-3}$. One must iterate over the chain of vertices in a polygon to calculate the number of notches. $Comp$ can be used as a complexity measurement for the LoD pairs. The complexity of each LoD can be represented as $comp_i$ and the dissimilarity function of $Dist2_t$ between the current LoD l_i and original shape l_n is

$$Dist2_i = comp_i - comp_n \tag{5.5}$$

We evaluated these approaches on four different datasets. The generated similarity distances $Dist2_t$ and $Dist1_t$ for these two measurements are shown in

Figure 5.11 for the progressive refinements of one example shape from the Figure 5.10. Similar results were observed with other over-represented test features in the datasets. We found that the simplified map has a very large dissimilarity to the original map which has the longest $Dist$. After a few steps of refinements, the polygon is now shaped with more significant features (possibly more non-convex features) during the progressive refinements where the shape is more similar to the original shape which has smaller $Dist$. The method of Brinkhoff et al. $Dist2$ has similar results with the approach, but does not show the changes in shape representation as explicitly as our measurements do. Using these measurements we can potentially describe the changes of the shape during progressive transmission where the shape can obtain the most relevant details in the early refinement steps and then obtain less significant refinements after this as shown in the figure 5.11. In this case, the two similarity distances $Dist2_t$ and $Dist1_t$ will begin to converge and tend towards zero dissimilarity.

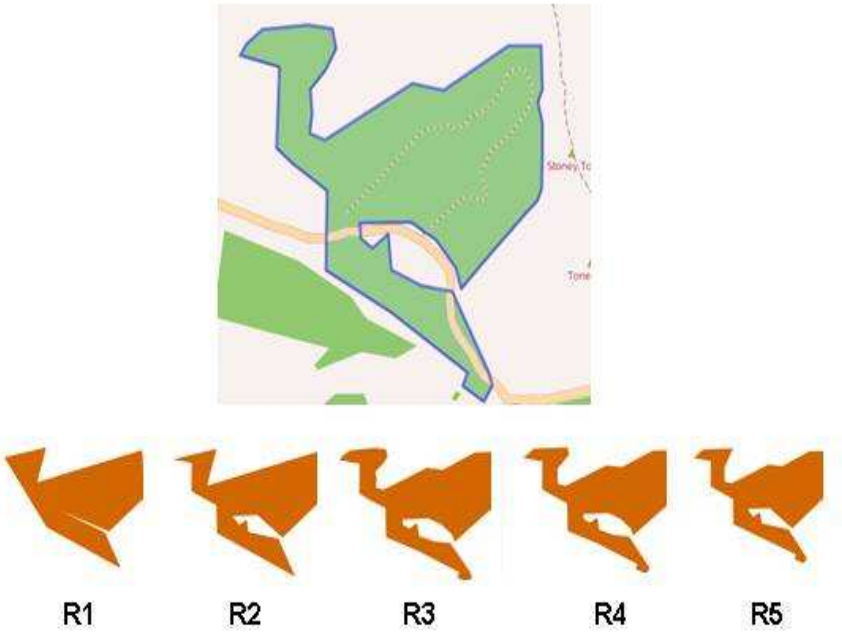

Figure 5.10: This is the sample original polygon shape to which progressive transmission was applied

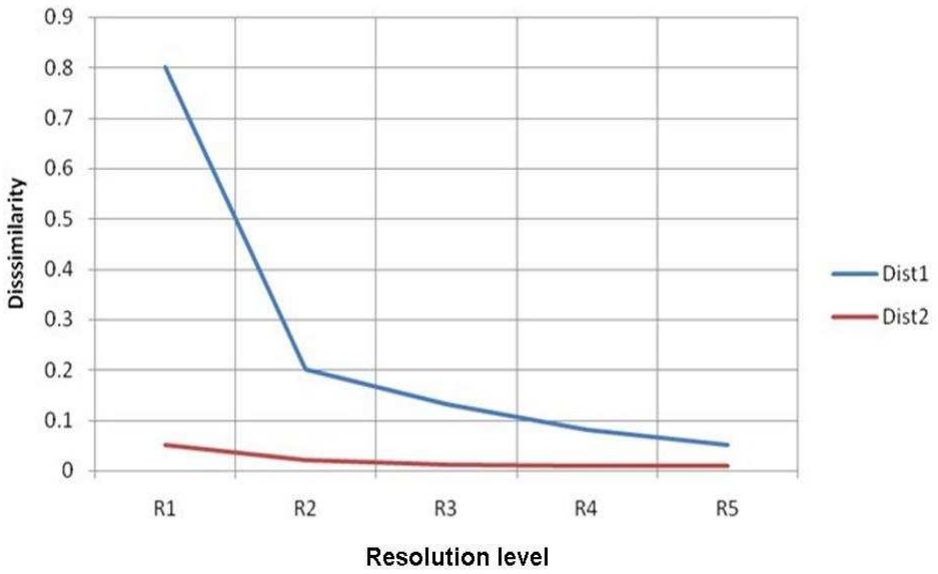

Figure 5.11: This figure shows the two similarity measurements performance over the progressive transmission based on shape complexity of the polygon shape shown in Figure 5.10 on page 122

5.4.5 Enhancing selective progressive transmission

When mobile users use their mobile devices to view spatial information on maps, transmitting a large amount of vector data through low bandwidth networks is challenging. One of the many challenges is due to the file size increasing unpredictably depending on the complexity of feature geometry Buttenfield [2002]. It is important to attempt to keep the amount of spatial data for delivery relatively constant. Because of network constraints, sending too much spatial detail could cause problems due to low bandwidth. However, transmitting too few details to mobile users might cause the slow down of visualization during user interactions and lead to an unsatisfying user experience. In this case the LoD sent by a progressive transmission scheme should be delivered "*on - demand*" [Cecconi et al., 2002]. However, there is always a trade-off between download speed and generalization quality when a user requests a map or map data from a server [Cecconi and Galanda, 2002]. The pre-computed application may fail in the case of sending desired details on the fly due to the complex generalization process [Bertolotto and Egenhofer, 1999] while other real time approaches may find it difficult to provide users with all of the data and information that matches their request specifications. In simple terms, we would like to send the user as much spatial data and information as possible based on their selection or viewing area on the map. However, we should only send spatial data which closely corresponds to the area of interest on the mobile visualisation of the early LoD which the user can see. The selective progressive transmission approach described above sought to provide "complex" objects with more spatial data than "simple" objects. The "complex" objects were given priority in the early stages of the selective progressive transmission process. As the LoD l_i begin to approach, l_n (the original dataset with full resolution) then spatial data is added more frequently to "simple" objects.

In the next subsection, we propose an extension to selective progressive transmission using a heuristic strategy to provide a good balance for these scenarios. We provide a cost-benefit function as a heuristic to guide the selection strategy. This heuristic selection strategy will help us to adapt and balance amongst the amount of LoDs to be transmitted to the client device and the visual contour

details which need to be enhanced and in which priority occurs. We call this extension "view-dependent progressive transmission". Spatial indexing techniques are used to restrict the delivery of the spatial data which is within the user's current viewing or request area on the visualisation on the device.

5.4.6 Using heuristics for selection of objects for selective progressive transmission

When users interact with maps, the complexity of the portion of the large scale map visible to mobile users can be highly variable. Depending on screen size, zoom position, pan position, etc there might be tens of thousands of vertices simultaneously visible from one user's selected view whereas a relatively small number of vertices can be seen from another user's view. There are very often examples of variation of the data representation quality from region to region and potentially within the same urban area in spatial data. Location-based Services applications that simply render all potentially visible polygons with some predetermined quality may generate frames at highly variable rates. There is no guaranteed upper bound on any single frame delivery time. In this case, progressive transmission smoothness or absence of "popping effects" is a vital goal. In order to provide smooth transitions between LoDs, we perform a constrained optimization routine that selects a level of detail to render containing each potentially visible object. This approach will produce the map within a specified number of LoD as a good representation as possible despite possibly being constrained by the network bandwidth. We will now specify this concept more formally.

We can define a spatial object as a tuple (P, L, T) which is the spatial object P, provided at LoD L within the transmission process T. We then define two heuristics for this object $Cost(P, L, T)$ and $Benefit(P, L, T)$. The $Cost$ heuristic estimates how many more LoDs remain before delivering the full resolution spatial object in tuple (P, L, T). The $Benefit$ heuristic estimates the dissimilarity between the current generalized map and the original full resolution map. We define S to be the set of objects which contains the tuples that are to be delivered. To take the network bandwidth restrictions into consideration, we set an upper threshold on the amount of data which can be successfully transmitted in

a frame. This is represented as SDS. Using these formalised approaches, the choice of LoD and subsequent transmission for each potentially visible object (in the mobile user's current) can be stated as follows:

Maximize:

$$\sum_{S} Benefit(P, L, T) \tag{5.6}$$

Subject to:

$$\sum_{S} Cost(P, L, T) \leq SDS \tag{5.7}$$

The equation 5.6 attempts to quantify the benefit (or reward) of sending a specific set of objects during this transmission. Then equation 5.7 attempts to optimise the contents of this set of object tuples by ensuring that we do not exceed the specified data threshold limit represented by SDS. This selection formulation can be used to derive the best progressive transmission strategy given the current circumstances and deliver the best quality LoDs. The limitation of bandwidth is taken into account by the contraint of SDS so that the formulation should deliver the LoDs as fast as possible within the limited bandwidth. The maximisation of equation 5.6 is designed to ensure that even objects with varying levels of data representation quality are combined to ensure that a consistent rate of data flow is maintained and transmitted over the network. The formalisation is an optimisation problem. Each time we select the most valuable configuration of objects in the LoD as the next candidate candidate for transmission. This will prevent excessive waiting time for generalization to complete while retaining good quality map visualisation. The user's request for increased detail in their map visualisation is also responded to as quickly as possible.

A good combination of $Cost$ and $Benefit$ for the evaluation of the map transmission based on this heuristic approach should exhibit two key features. Firstly their formulation should ensure that we can easily identify the map changes during the transmission. These changes to the map during transmission should ensure that the visual quality of the generated maps improve over the transmission period. To control the rate of map changes we choose the $Dist$ as one key parameter for $Benefit$. This measure can determine whether a single spatial object obtains enough representation details during the progressive transmission

of LoD refinements. However, other $Benefit$ factors have been identified in the literature and can be considered and integrated into this approach such as: the area importance [Haunert et al., 2009; Oosterom, 1993], or semantics, color and so on. In the experiments, for simplicity and efficiency of the computation we have only chosen the most relevant factors as the factors to derive $Benefit$. This is outlined in equation 5.8 which represented by a multiplication of $Area$ by $Dist$. This reflects a possible reduction in the object tuple benefit to the overall area. Equation 5.8 is then represented as follows with the overall Benefit heuristic a product of the two selected factors:

$$Benefit = Area \times Dist \tag{5.8}$$

The concept of equation 5.8 is easily explained. Larger and more complex objects are given higher values of $Benefit$. These objects, when they are refined, will lead to a greater contribution to the perception of the map by the user. A $Cost$ measure can be the remaining transmission time or the data amount required to achieve the full resolution of the shape. For example, if the spatial object obtained from the OSM data is over-represented it is unlikely be refined. This polygon object will use up large amounts of data $Cost$. However, its overall $Benefit$ to the process of visualisation will be small. The additional LoDs will contribute very little $Benefit$ to the overall map. This constraint strategy can be very useful when integrated with progressive transmission. This is because it will enhance the selection process of a group of spatial objects by iteratively selecting the most important area in the dataset to deliver the LoD until all areas have sufficient representation.

Practically this constrained optimization solution is a NP-complete problem. It is the Continuous Multiple Choice Knapsack Problem and has been well studied in the literature [Pisinger, 1994; Sinha and Zoltners, 1979]. It is a well known knapsack problem, in which one chooses the optional elements from participated options to put into a knapsack K and maximize the value of profit while still keeping the overall weight under or equal to a given threshold. The problem in our case is to select the objects that have the maximum visual contribution benefit but whose transmission cost fits into the constraint of potentially limited

bandwidth (the threshold of SDS). We have implemented a greedy approach to this problem, where we chooses the object with highest $V = (\frac{Benefit}{Cost})$. This will run with complexity $O(n \log n)$ where n is the number of objects. We maintain the spatial objects in a heap data structure (sorted by the V criteria). The highest values of V are at the top of the heap. When the user is interacting with the map and moving around the map viewing area, there is a requirement for frame-to-frame coherence during user interaction with maps. If the user changes their viewing area, there will be new visible objects assigned to the lowest LoD. Some existing objects will be assigned to the higher levels of LoD.

In this situation, we can have two options for delivery of additional spatial details. The first option is that we could send the LoDs required to refine the existing objects and include many insignificant nodes (the scenario here is higher $Cost$ with lower $Benefit$). The second option is that we could decide to send the spatial detail to the new objects. Because these LoDs are positioned early in their overall sequence of LoDs, any additional spatial detail to these new objects will be a significant addition to this shape (the scenario here is higher $Benefit$ with lower $Cost$). In this cost-benefit model, we direct the optimization approach to choose the objects which will maintain the higher $Benefit$ with lower $Cost$. In the case of the visualisation on mobile devices, it is necessary to provide a sufficient number of refinements to lower resolution maps. Because of the requirements of frame-to-frame coherence, when the map is visualised and displayed, the approach outlined here provides a good selection of refinements for the low detail maps. However, for a larger scale map, it costs more, in terms of time, to perform the selection of candidates from a larger set of polygons. It is necessary then to reduce the number of selections and restrict selection to the set of objects within the users' current view.

5.4.7 View-dependent progressive transmission approach in context

In the previous section, we described an optimisation model for the view-dependent progressive transmission approach. Rather than the crude simplistic approach of sending LoDs for the entire area requested by the user our, view-dependent ap-

proach optimises LoDs for the area of viewing. When the user is viewing the map on their mobile device they can pan around the map freely. We track the AOI that the user is moving through. As proposed in the section 2.2.6 the OSM server stores vector based maps composed of OSM objects that have various representation quality. The OSM database is very large and very dynamic. In the OSM database, some regions may contain hundred of objects and it is not possible to transmit all the objects to the mobile client immediately. Even for standard desktop-based web browsing of the OSM website, it has constraints placed on the number of objects that can be displayed on web-based maps. When one views live objects on the OpenStreetMap, 'data view' part of the OSM map most web-browsers become overwhelmed if more than a few hundred objects are requested for display. The user must then zoom in to a more specific area or cancel the requested operation. However, in practice, a user usually needs to only view a small portion of data in detail rather than the entire dataset (i.e. when looking for a bus station beside a national motorway). When we move from the desktop to the mobile, the storage capacity in the browser is even less. The data storage facility in mobile devices is limited, it is necessary to restrict maintenance to only the spatial objects within the visible area. We discard any irrelevant spatial objects outside of this area. In our example of the bus station, we only deliver the detail of objects connected to the national motorway object for the region. Other superfluous objects to this specific query are omitted.

We feel that this approach is well placed within the state of the art in the literature. Multi-resolution LoD approaches are favourable techniques "used to bridge between complex spatial datasets and low-capacity client side visualizations" [Follin et al., 2005a; Hamid and Ahmed, 2010; Li, 2009; Yang, 2005; Zhou and Bertolotto, 2004]. These approaches send the coarsest datasets as initial datasets to the mobile clients when the user begins to interact with the map. The users can download the simplified datasets at level LoD_0 (Base map) very quickly at a reasonable resolution to view and understand the global context before performing any further queries or requesting more detailed spatial interaction. The server progressively transmits LoD refinements to reconstruct the map with the user-specified scale. In simplistic approaches the generalized maps are transmitted to the client in the opposite order to which they were created [Ai

et al., 2005]. This transition between scales is achieved through a process of refinement which transmits only the additions required to compute the current scale from the previous [Corcoran et al., 2011a]. He also stated that such a scale-based progressive transmission approach introduces major inefficiencies when the client requests only a small region of a large map. For the mobile devices with limited screen size, users usually only can view a small area of the map. A scale-based approach may cause an "information over-load" by loading excessive amounts of data into memory.

As described in Section 5.4.5 and section 5.4.6 above view-dependent progressive transmission is a technique to transmit the data depending on where the user is currently viewing on the map visualisation. It has been implemented in raster-based map tiles [Tanner et al., 1998]. The idea for view dependent progressive transmission is also widely used in the computer graphics field for efficiently rendering progressive meshes [Cheng and Ooi, 2008; Hoppe, 1997; Luebke and Erikson, 1997]. In computer gaming, for example, rendering complex geometric models at rates which support interactive game-play, the most serious restriction is the bottleneck of bandwidth of the network. To avoid sending irrelevant data, only the LoDs of visible spatial objects will be selected and sent to the client for rendering. The view-dependent strategy selectively transmits the LoD and refines the view according to changing view parameters [Hoppe, 1997].

In many Web mapping applications, view-dependent strategies can be achieved by introducing spatial indexing techniques. Several well-known spatial indexing techniques have been used in many GIS applications to generate hierarchical structure for representing spatial objects, such as R-Tree [Guttman, 1984], Quadtrees [Samet, 2005], Octree [Meagher, 1982] and gap tree [Oosterom, 1993]. In Augusto et al. [2009] the authors use Quad-trees to select the spatial objects intersecting the query windows and progressively send the LoD updates for refining the user viewing area. Antoniou et al. [2009] proposed tile-based approaches for dividing the map into tiles and only those tiles contained within the user's view are transmitted. Corcoran et al. [2011a] integrated the R-tree technique to maintain the topological consistency of the spatial objects only within the user's current view to reduce the computational expense. For mobile client, mapping applications integrating such spatial indexing techniques results in significant savings

in data transmission and storage for a client without generating any noticeable visual differences for the map which the user is currently viewing.

For the purposes of illustration in Figure 5.12, we see two polygons in a large region. The user starts to zoom in and pan. The standard scale-based approach sends LoD for refining all of these polygons (tab 2 in the illustration). The view-dependent approach selects the visible polygon(s) for refinement (tab 3 in the illustration). The later approach leads to a significant reduction of the LoD data volume while there is no difference in the actual user screen view. In this work, for the purposes of accuracy and ease of implementation, we have used an Octree to organize the spatial objects in the map and facilitate the progressive transmission of a group of vector based spatial objects within the query window (or AOI). The level LoD_0 (Base map) is sent to mobile clients with a larger scale. Then the objects only within the user's current view are refined with subsequent LoDs. Therefore, the window query performed, based on the user actions, selects polygonal objects intersecting with the query window. These objects are extracted from the relevant datasets for delivery "on demand".

5.5 Implementation example and discussion

We now illustrate an implementation example of this cost/benefit optimization approach for the proposed view-dependent progressive transmission scheme. The key requirement of this approach is to generate a more uniform size of delivered datasets, which takes network bandwidth issues into account. The spatial data "load" is spread more evenly across all of the LoD from L_0 (Base map) to the full resolution dataset L_n. For this example, we conduct a test run to simulate the user interaction with a real-world mapping application on a mobile device. Within the software code, we placed timing points and collected information about the data structures and the current state of the processing of the LoD from $L_0 \ldots L_n$. We implemented the three approaches in the application as follows:

1. **Full scale progressive transmission:** This is the simplest approach which sends the datasets to the map with large scale by reversing the generalization process. This process sends the details from a low LoD L_0 to

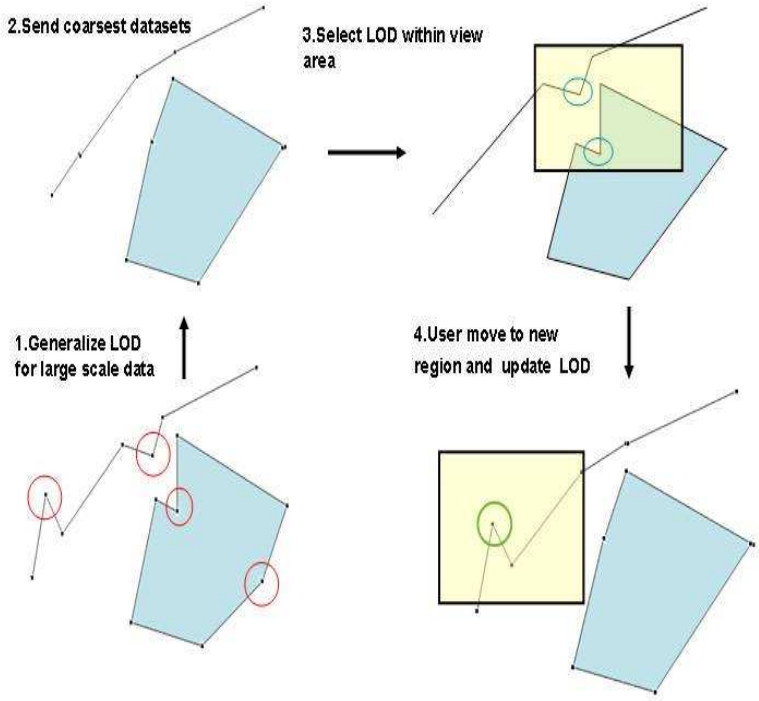

Figure 5.12: This is a simple example of the view-dependent approach where selection of the visible LoD to refine objects within the user viewing area is performed instead of sending LoD at full scale for all of the objects in the viewing window area

high level of detail (towards L_n) until all details of the entire map are sent. If the user moves to a different area of the map, outside the current viewing window, then this entire process must be repeated.

2. **View dependent progressive transmission:**. This approach sends the lowest level of detail to the full scale map. After L_0 the decision making process ensures that only objects within the user's current window view can be selected for additional LoDs for incremental refinement.

3. **Optimization approach:** This is proposed optimised version of the view dependent progressive transmission. This approach, as outlined above, transmits the LoD based on the $Benefit$ (i.e. Dissimilarity between LoD) and the $Cost$ (i.e. data transfer based on the network characteristics). Those objects with a higher profit (greater $Benefit$) are chosen provided that they can be transmitted at a reasonably low $Cost$.

The implementation of these transmission techniques for the OSM vector datasets are performed over the Internet to mobile devices on a standard mobile broadband connection. All software was written in Java and the client mobile device was an Android-based mobile device. The datasets were chosen to ensure that it was complex enough (total size is 8.56MB) to be sent in a limited bandwidth of 100 kb per second. This is also small enough for the phone to reside entirely in main memory so as to eliminate the effects of memory management in this test. Figure 5.14 shows the dataset from the map in Figure 5.13. The user interaction follows the path as shown in Figure 5.15 with the red dashed line. The user begins browsing the map at region A and keeps on moving to right from region B to C. This is the typical panning behaviour of user interactions with LBS maps. The 3 regions in the map are chosen for the following reasons. "A" has a large and complex polygon, "B" has a few small and simple polygons, while "C" is the place the user is moving towards with a similarly complex polygon as in the region "A".

The Figure 5.14 shows a simplified version of the map in the Figure 5.13. We send this map to user as the base map (L_0) to start browsing and visualisation. The users could follow the path in Figure 5.15 on page 136 to browse the map. We use purple colors to render the high detail objects and use green colors to render

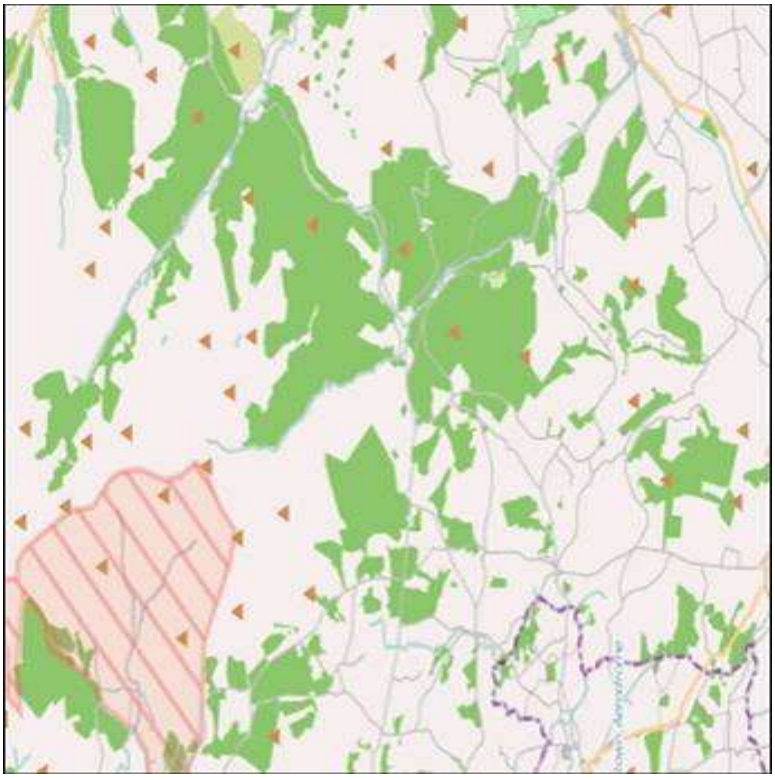

Figure 5.13: This an OSM map of the example spatial dataset used for the illustrations in this Chapter. The map shows the region of Aghavannagh, Co. Wicklow, Ireland

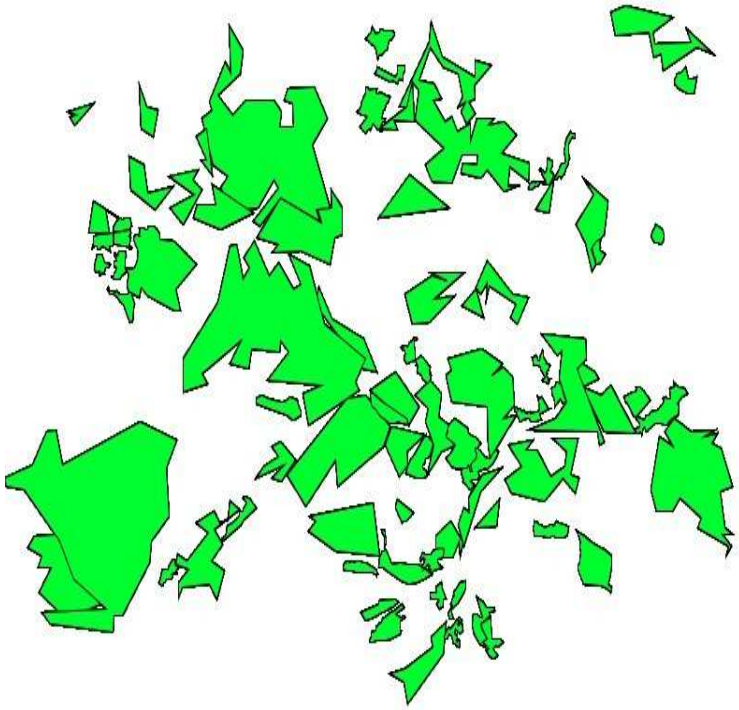

Figure 5.14: This test dataset, from Figure 5.13, has 71 polygons with varying complexity and area sizes. This dataset is used to illustrate the implementation of our optimised view-dependent progressive transmission approach. The total number of nodes in this dataset is 6649. When simplification has been performed, this is reduced to 1733 nodes.

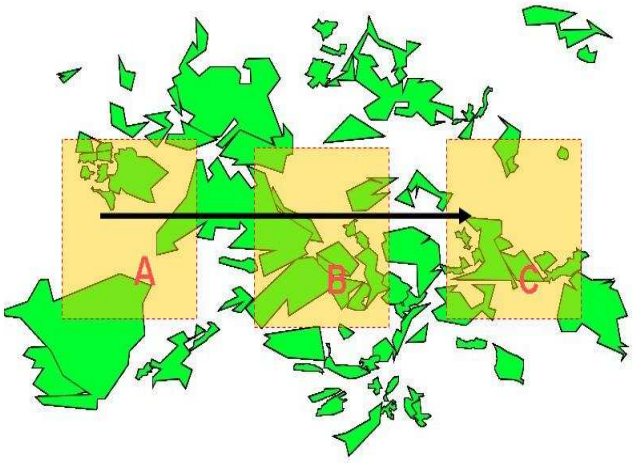

Figure 5.15: The red line crossing the map is the simulated path of the user's interaction with the map. The user starts their interaction with the map from region A. The user then moves to region B and finally stops in their target region C. This path is used in the tests for all of the three models. The original visualisation of this spatial dataset is shown in Figure 5.14

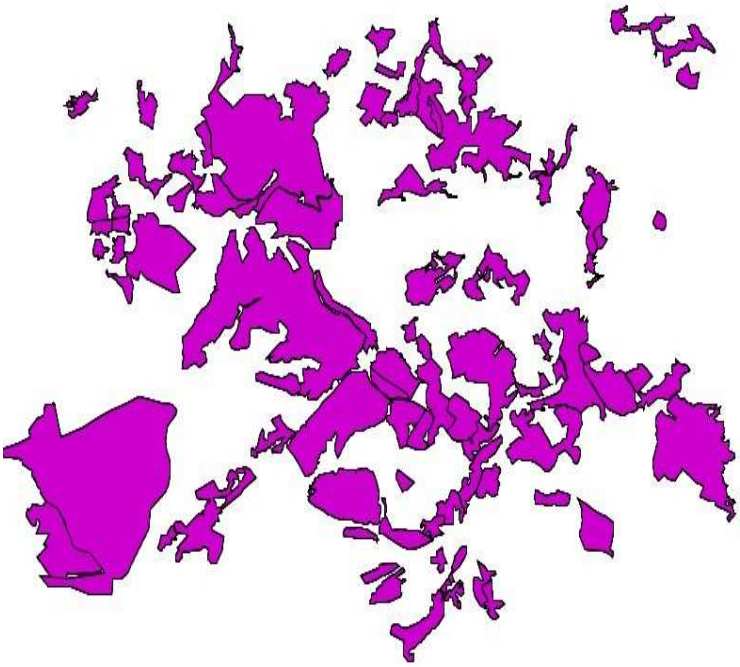

Figure 5.16: This figure shows the full scale transmission which transmits the data from Figure 5.14 with high detail (in purple color)

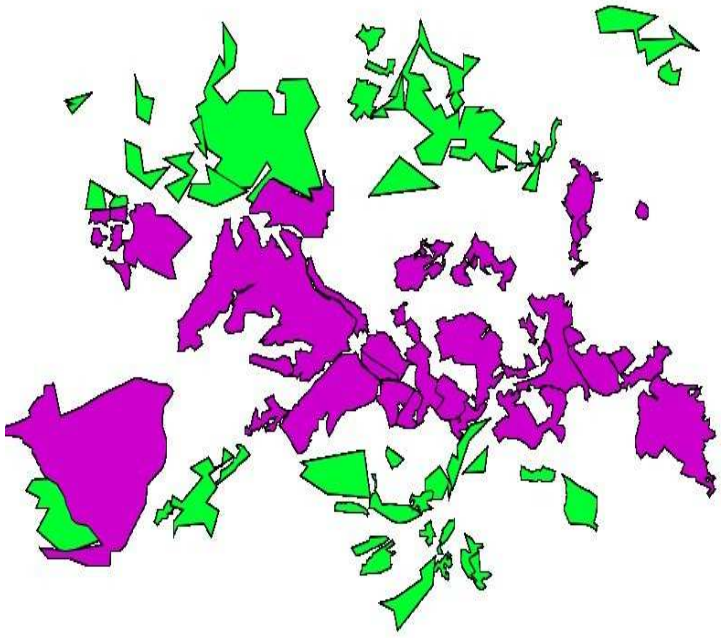

Figure 5.17: This figure illustrates the concept of the view dependent transmission. In this case, we only send the spatial data corresponding to the user required regions. Only the polygons within the path of interaction (with purple color) will obtain the refinements and additional data. The polygons with green coloring, which are not in this path, receive no additional refinements. As before the map data corresponds to Figure 5.14

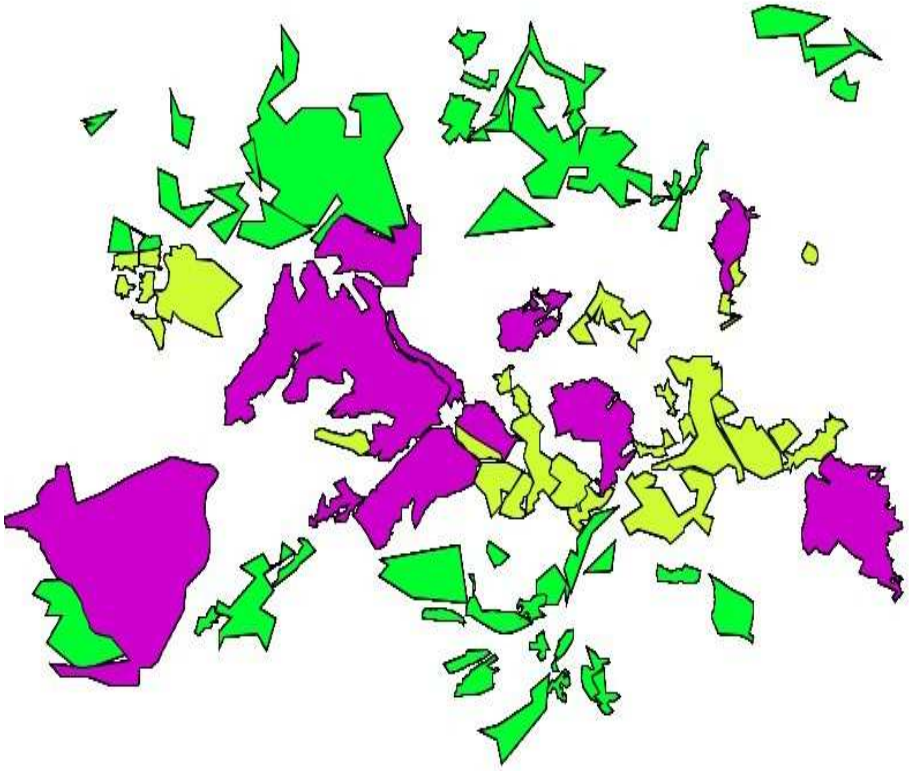

Figure 5.18: This figure shows the visualisation of the transmission provided by the proposed optimization approach. In this approach, we only send the data to regions required by the user. In the path, only the polygons with high *value* (that is high $Benefit$ with low $Cost$ shown with purple color) are assigned the highest priority to obtain more refinements. Other polygons with lower *value* (low $Benefit$ but with high $Cost$ shown in yellow color) may obtain additional refinements if there is enough bandwidth available. As before, in Figure 5.17 the polygons of green color, which are not in the path, receive no additional refinements.

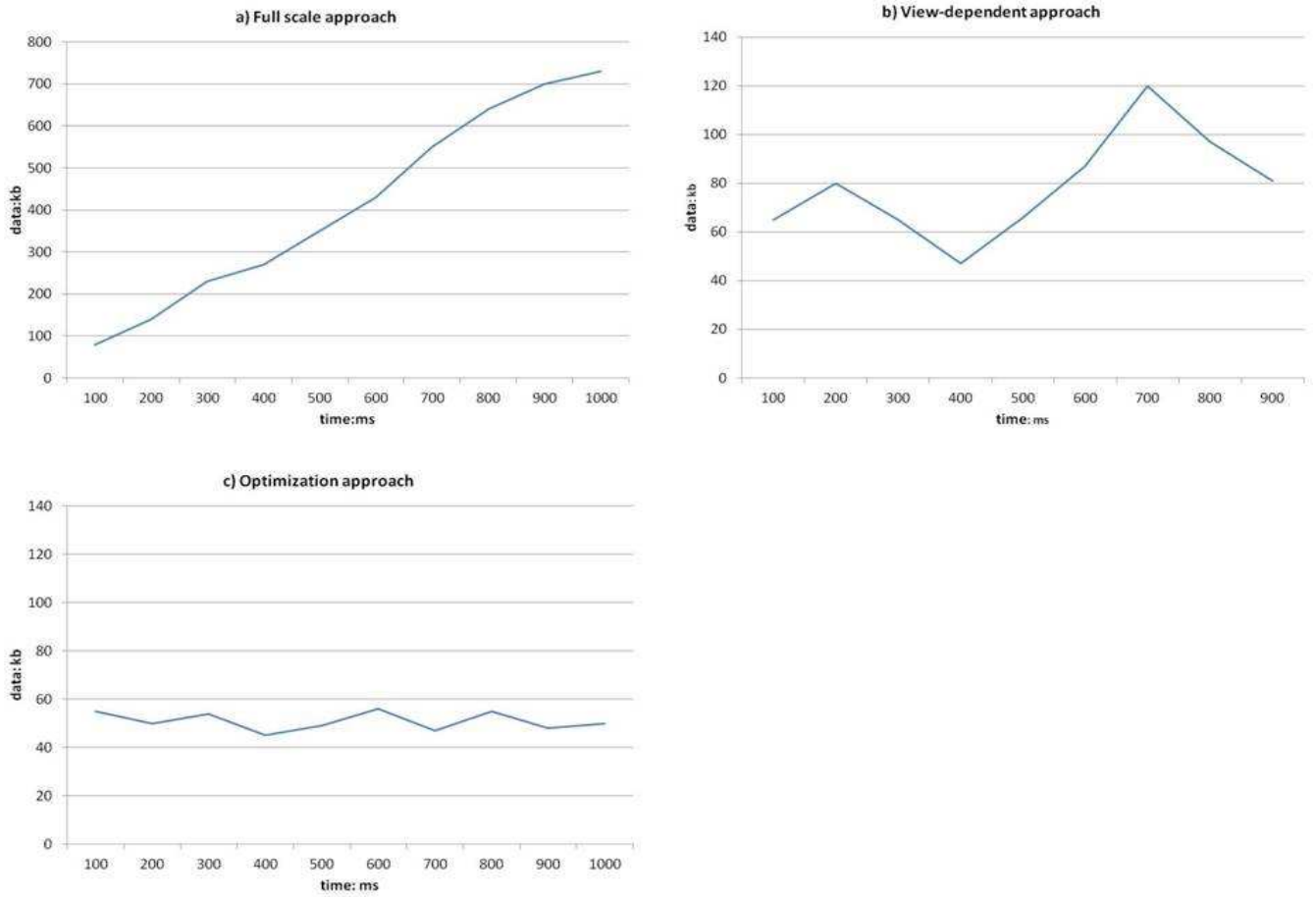

Figure 5.19: This figure shows the results of the overall time and dataset size per LoD for the three models outlined above in this section (5.5). (a).Full scale progressive transmission. (b).View-dependent progressive transmission. and (c).Optimization progressive transmission

the low detail objects in the map for clarity for these examples. The objects with intermediate level of detail will be rendered in a yellow color. The full scale transmission will send the LoD refinements to the objects in the full scale of the map without considering the user's viewing path. This map is then refined with a large amount of vertices with high detail (in purple color). The view-dependent transmission only sends the LoD refinements to those objects within the user's visible area as shown in the Figure 5.17 on page 138. However, in situations of low bandwidth, our optimization approach can select the objects, which have high visual contribution $value$ (high $Benefit$ with low $Cost$) within the user visible area. These objects are given higher priority to transmit additional data. In the limited bandwidth situation, we take these objects as priority candidates to refine the map. However, we make the best usage of the available bandwidth. We also send details to the other objects within the user's visible area if it is estimated that there is additional bandwidth available.

Figure 5.19 shows three plots of time against data size of the LoD for the three approaches above. The full scale progressive transmission (Figure 5.19a) sends the data from L_0 onwards which at first contains small data quantities. As the map visualisation approaches, full resolution larger volumes of data are delivered towards L_n. There is a greater overall data burden on this approach. While the view-dependent approach, as seen in the Figure 5.19b, can significantly reduce the data amounts transmitted by sending the data within the user view area. The first peak in the time series relates to the data needed as the user moves through region "A". As the user moves to the region "B", where some simple and small polygons on the map, the size of data in the LoD is greatly reduced. Finally as the user moves from "B" to "C", the rates of data transmission is increased as the user is viewing more complex objects with larger areas. In the full scale transmission, we send all LoD with $6,649$ nodes. In the view dependent transmission we only send $3,386$ nodes, but there is no real visual difference to the user when they are browsing the map. In the optimization approach, we send $3,313$ nodes and obtain very similar visualization quality. Figure 5.19c gives the time series results of the optimization approach which selectively sends the data within a uniform data transfer rate. The trend of this time series line is very different to that in (a) or (b) in the Figure. The rate of data transfer is very

consistent for our optimisation approach.

The optimization approach is a very positive development. It provides the user with a satisfactory representation quickly. Full scale progressive transmission is slower than the latter two approaches and has the longest waiting time. The proposed optimisation method generates appropriate LoD and fully utilises the available bandwidth with maximum benefit in order to improve the map representation and visualisation on the client device. As a result, the data transmission is performed at a uniform rate and can be rendered in a more predictable frame time. However, what if the user moves around the map frequently and in particular moves over complex geographic regions? The spatial data transmitted to the mobile devices will require increased usage of the client device's on-board memory if this process is not well managed. This brings us to the introduction of the next chapter where we will discuss the data management at the client side in more detail.

5.6 Chapter summary

This chapter has introduced progressive transmission, selective progressive transmission, view dependent progressive transmission, and then finally an optimised view dependent progressive transmission scheme. The goal of the optimised approach is the balance out the trade-off between a high resolution map representation on the mobile device against the constraints of the network bandwidth. We have provided mathematical formulations for these approaches and for the shape complexity measures we have implemented and used. The chapter ended with a discussion of an implementation example of these schemes.

In the next chapter, we discuss the processes behind the rendering of the map visualisations on the mobile device. This involves a discussion of the framework for the storage of the spatial data on the mobile device, the access to this spatial data, and the rendering of the spatial data into a cartographical representation. We also describe an enhancement to our optimised view-dependent progressive transmission scheme where we include a 'user prediction' component. Based on the movement of the user within the current map view can we predict where the user will move to next? If we can predict this with a high degree of certainty,

the framework for the application could pre-fetch LoD for the predicted area in anticipation of the user moving their AOI to that region of the map.

Chapter 6

Rendering and visualisation of progressively generated maps

In the previous chapter, we provided a rigorous outline of progressive transmission schemes for spatial data. This discussion involved three parts: full scale progressive transmission, view dependent progressive transmission, and finally a scheme to optimise the amount of spatial data transmitted for each LoD to ensure an optimal cost-benefit approach to the progressive transmission. Up to this point in this thesis we have focussed almost entirely on the delivery of the spatial data to the mobile client device. There has been little discussion regarding how the spatial data is visualised on the client device except for some brief examples in the form of screenshots. The formulation of the structures for preparing and then delivering the LoDs have been the subject of our focus.

In this chapter, we move to the client device and discuss the issue of rendering and visualising the spatial data received from the progressive transmission scheme. We have argued that the progressive transmission of LoD provides a solution that enhances a user's experience when accessing large scale geospatial information and data, and in particular dynamic data such as VGI. The user always has access to the most up-to-date data. There are issues that must be addressed in terms of ensuring that the user's experience of using these maps and these applications is good. When users are interacting with maps on mobile devices, they may switch the spatial context frequently. The resulting overhead in

terms of mobile device rendering, on-board memory usage, and network latency are major obstacles to achieving optimal map interactivity. On slower Internet connections, we have all had experiences of using tile-based mapping systems such as Google Maps and waiting for some time until all of the tiles download completely. In this chapter, we describe the development of a multi-thread based data management strategy for visualisation in the mobile client with regard to these constraints.

A client view-dependent updating approach is proposed to maintain the data in the mobile device memory during continuous user interaction. This prevents repeated downloading of irrelevant datasets into memory to support the visualisations of the user's current view. We also describe the synchronizing mechanism and dynamic adjustment mechanism to enhance the visualization process. The data query processing run in parallel to offset the effects of any potential network latency. This synchronization mechanism is implemented using a "look-ahead model" to pre-fetch the datasets of the user's view area for the next frame while the user is still visualizing the current view. An experimental study is presented to outline the performance of this parallel approach. The findings indicate that there are benefits to using this strategy in managing the data in mobile clients.

6.1 Introduction

Visualization of spatial data in mobile clients can help users perform analysis and decision-making in LBS. When one reflects on the last decade, it is only a few years since Google Maps appeared on desktop-browsers and many in GIS and the IT industry knew that a major turning point in web-based cartography had been reached. Google's mapping and direction-finding services were a stunning improvement upon any other systems available. The maps were aesthetically pleasing and readable. Using Javascript and AJAX technologies they filled as much with the screen (or the printed page) as the user provided them. Scrolling the map worked in the most natural way by dragging the image. With the mobile device revolution of the last few years, much of the effort of industry and research is focussed on "maps on mobile devices". The development of these technologies have brought a revolution in respect to the widespread online access

to geographic data through map-based interfaces [Wilson et al., 2008]. People can use their mobile devices to access maps anywhere at any time. Many of these LBS applications are often characterized by high interactivity and strong end-user participation. For example, user can quickly browse a map area and select a restaurant to obtain corresponding review information. Such tasks, associated with rapid map context switching, usually require large volumes of data exchanging and updating between the client and server. Comparatively slow network transmission is considered as a constrained resource of the system for remotely accessing data [Schmalstieg and Gervautz, 1996]. This is where a progressive transmission scheme is best applied: to deliver the LoD of the multi-resolution data to the mobile clients resulting in faster visualization and resource savings for the mobile application. In mobile clients when multi-resolution datasets are used the application must regulate the amount of spatial detail delivered to the client.

However, as indicated in the Section 2.2.5, the visualization of spatial data is challenged by the constraints of low-end mobile clients. They usually have a very tight budget in computing resources and data storage. It is not always necessary to maintain the data with high LoD in the several scales in the mobile device, since the majority of the high-resolution datasets are not always used during user interactivity. One must carefully manage the data to deliver it "on-demand" [Foerster, 2010] to the mobile clients. Increasing the amount of data on the client device will cause problems of overloading with respect to data storage [Schmalstieg and Gervautz, 1996]. Each time a user changes their area of interest on the map the contents of the client memory must be updated by requesting relevant data from the server and discarding any unnecessary information already residing there [Augusto et al., 2009].

The graphical rendering component is a unique resource in the mobile client, which must render the maps after all the newly received LoDs are integrated correctly. Multi-resolution approaches in mobile clients may keep updating the map representation when continuously receiving the incremental LoDs datasets from the server. We believe that it is possible to schedule the rendering and transmission of the spatial data in parallel to deliver a speed-up in the visualization process. This will also allow us to manage the graphical resources in the client.

We aim to provide a sophisticated data management framework for visualization of LoDs on the mobile device whilst dealing with constrained computational resources on the mobile clients. We begin our discussion with a description of interactive rendering of the spatial data from LoDs in the mobile client device. In particular we focus on the three principal components for interactive rendering: rendering itself, updating the map data, and then the pre-fetching of the next LoDs.

6.2 Interactive rendering of spatial data in mobile client

In the interactive rendering model we propose here, the selection and transmission of suitable LoD updates to the client are based on a selective progressive transmission scheme. The client has three components, which are working together in parallel for performing efficient map visualization on the mobile device. These components include: the Rendering component (R), the Updating component (U), and the Prefetching component (P). R handles the user interactivity and associated events and sends parameter requests to the server. R also renders the map in the "canvas" which is a graphic API available in most smartphone client devices. U manages the reconstruction of the spatial objects while continuously receiving the LoD updates from the server and deciding upon the correct rendering according to the map features (e.g. a lake should be rendered using a blue colour whilst an object with type park should be rendered in green). P will check whether there are any reusable datasets in the client and attempts to predict future user motion for the next area of interest. Pre-fetching, as performed by P, can help to reduce the effect of network latency by essentially having spatial data "ready" before the user actually formally requests it (by moving to the area P predicts they will move to).

This sequential approach is shown in Figure 6.1. All tasks are processed sequentially, where the client sends the request and waits to receive the updated LoD response before refining the initial maps with LoD and finally displaying the newly rendered map. At the points where the user moves the map or interacts

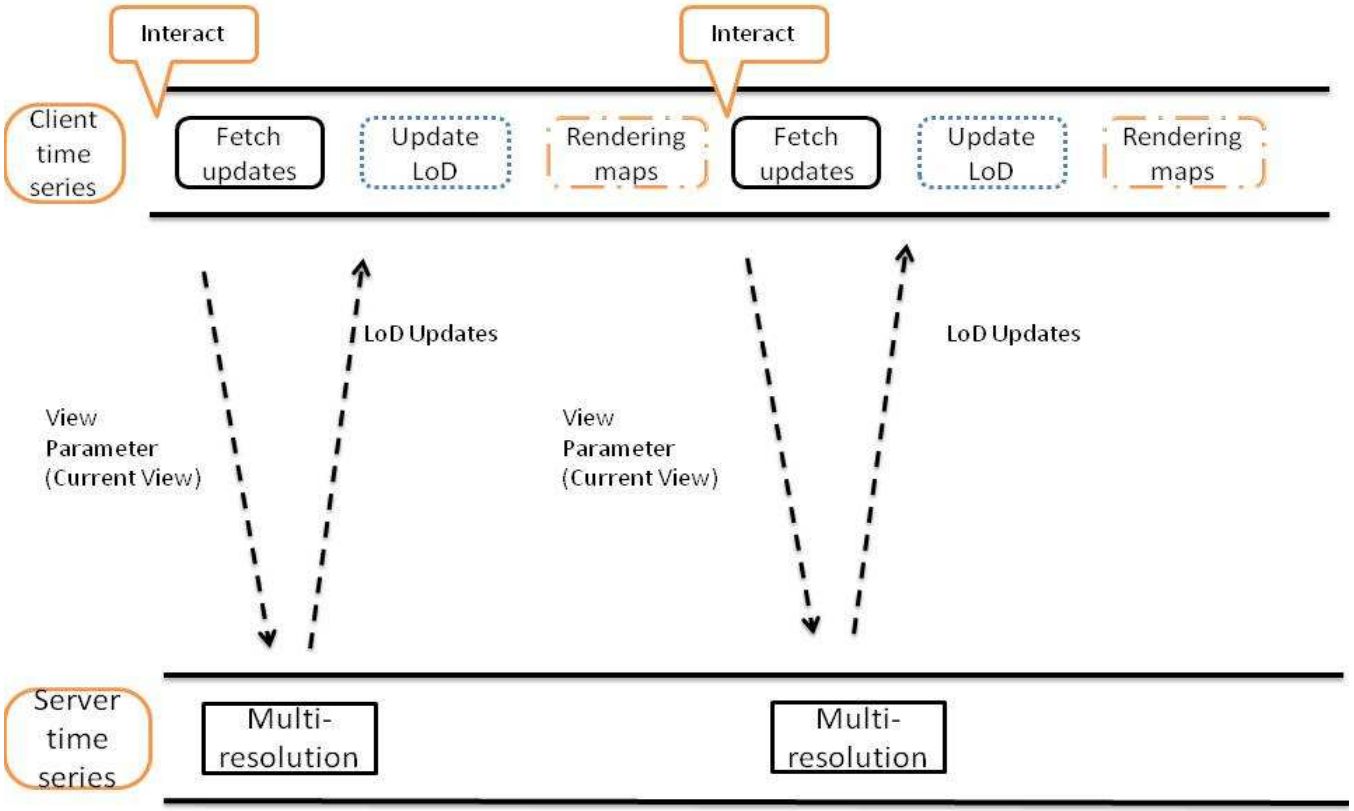

Figure 6.1: This is an example of the sequence of events which occur between server and client in order to deliver and then render a map in client device

with the map in some way this sequential process is then repeated. Similar approaches have been proposed in the early computer graphics literature. Floriani et al. [2000] considered the problem of transmitting huge triangle meshes in the context of a Web-like client-server architecture. Floriani et al. compute approximations of the original meshes which are transmitted by applying selective refinement operators. To et al. [1999] present an efficient multi-resolution method that allows progressive and selective transmission of multi-resolution models. This is achieved by reducing the neighboring dependency to a minimum. They allow visually important parts of an object to be transmitted to the client at higher priority than the less important parts and progressively reconstructed there for display. These works, as is the case with our work, used the earlier work of Hoppe [1996] on rendering progressive meshes as a platform for further work. Progressive meshes define a lossless, continuous-resolution representation of arbitrary meshes, and support scalable rendering.

In Figure 6.1, the sequential approach begins with the initialization step. The client P sends a query for a map area to the server. The server then streams and simplifies a map within this bounding rectangle from the OSM database and sends the coarsest map L_0 in this scale as the base map. When the user is presented with map visualisation of L_0 this interaction triggers the P to send the view parameters of the map to the server with incremental LoD update requests. The application software code on the server then traverses the multiresolution hierarchy for this request and produces a stream of refinements according to the received view-window parameters. As outlined in Figure 5.1, a GeoJSON format object is used as a light weight data exchange format in our data transmission process. There will be more discussion on this later in the section. The updated LoDs for each polygon are sent to the client and subsequently reconstructed in the client by U. The selected visible datasets are sent to R and drawn on to the "canvas" component of the graphic API. The updated map is then displayed to the user.

In terms of the client, the process of displaying one frame of the map to the user includes 3 major component procedures:

1. **Query processing:** This component includes sending requests and retrieving the spatial data from the server responses;

2. **Update processing:** This component involves the reconstruction of the spatial data in a graphical representation on the mobile device and integration into the existing rendered map;

3. **Rendering:** This component is where the software on the mobile device invokes the graphic API such as "canvas" to draw the spatial data contained in the on-board memory. The graphic API in mobile device only allows the "canvas" to be updated once all the update LoDs in one frame are fully integrated.

To formally introduce the concept of a sequence of operations which occur as the user interacts with the map and the server is performing Query and Update processing, let T_q be the time for the "Query process", let T_u be the time for updating the LoD of the map and T_r the time required for rendering. In this sequential formulation, the overall frame time is $T = T_q + T_u + T_r$. This sequential approach is very easy to understand. It also mirrors the way traditional client-server interactions are performed.

The problem with this sequential approach is that the waiting time is too long and can compromise the interactivity on the client side. Using a selective or view-based progressive transmission approach or the cost-benefit approach from section 5.4.6 means that the "Query process" time used by the server has been minimized. In summary, the sequential approach is not efficient. Its individual parts are efficient, but when we consider a summation of these, in sequential order, there is a detrimental effect on the overall performance. In the next section we will introduce an enhancement to this sequential approach with a parallel strategy for the interactive rendering.

6.3 A parallel strategy for monitoring interactive rendering

In many client-server system applications, the server interacts with the client without taking account of the resource budget available to the client. Most LBS are time critical tasks where the user requires the services be of acceptable rep-

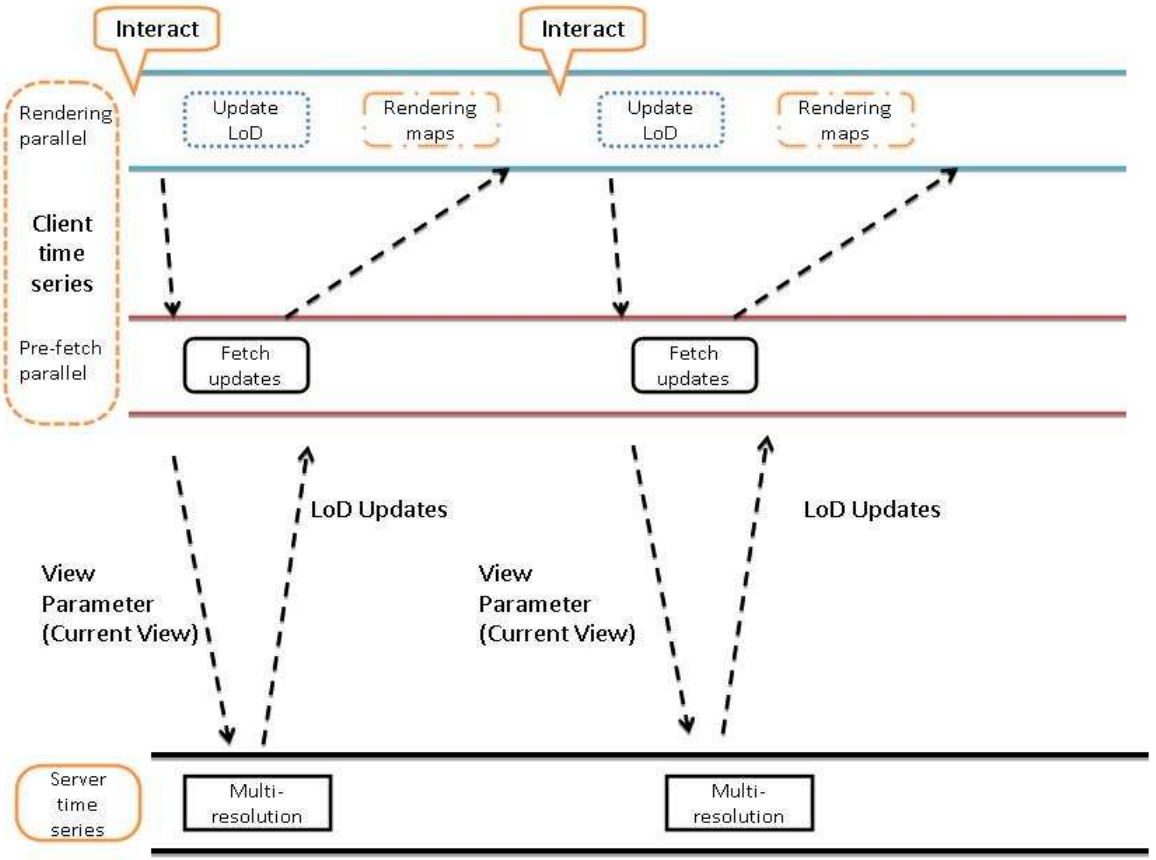

Figure 6.2: A parallel strategy for rendering the map in the client. This is an improvement on the sequential processing outlined in Figure 6.1

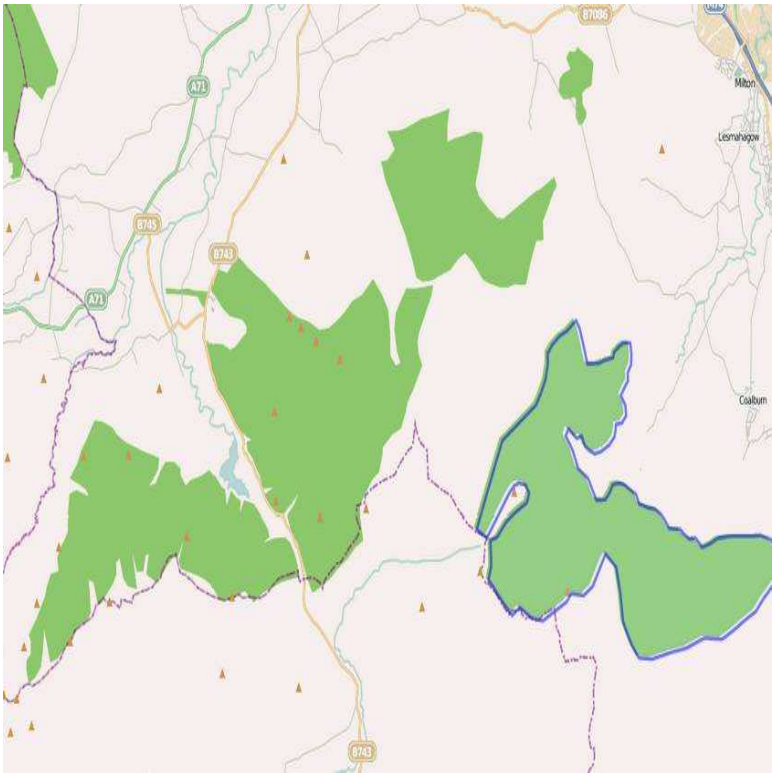

Figure 6.3: The is an example of one of the OSM datasets we have used for development of this strategy

resentation quality. The low memory capacity of mobile devices will have to be managed correctly in order to achieve these goals. The progressive transmission schemes we proposed in the previous chapter (for example section 5.4.6) manage efficient transmission in terms of the time and the data sizes. It goes without saying that it will take a longer time to update and render the spatial data in the mobile client when the mobile client has limited computing resources. Since the time required by the "Query process" is mainly a function of the waiting time for the server response, it does not have a significant resource competition with the other two processes. The updating process occurs in memory and the rendering process relies on the performance of the graphics component. Consequently there exists the potential to improve the overall running time by using an asynchronous rendering strategy. This asynchronous rendering strategy can be used to parallelize the "Query process" for the next frame and the other two processes of updating and rendering for current frame. The run times of these procedures can be overlapped to achieve a better overall frame rate. The time cost of the "Query process" can be greatly diminished and the mobile client will still receive the necessary *LoD* updates in advance for the next AOI (Area of Interest) the user will view. In this way, the previous $T = T_q + T_u + T_r$ will now change to the total cost of the frame time as $T = T_u + T_r$. This parallel strategy is highlighted in the schematic in Figure 6.2. One should clearly see that the strategy is different to that of the sequential approach illustrated earlier in Figure 6.1.

We will now discuss the operation of this parallel strategy Figure 6.2. The sequential method in the previous section sends the parameters for the current view to the server as in the sequential method. In this parallel strategy, before each frame is rendered, the client attempts to predict the view parameters for the next frame when the user will interact with the map. These predicted view parameters are sent to the server as part of the next LoDs update request. (The user motion prediction, which determines the AOI the user will examine at the next step during their interaction, will be explained in the next section.) Receiving the LoD updates for the next frame is handled by the rendering component of the current frame. The improvement provided by this strategy becomes apparent. When the next frame is required for rendering, the necessary LoD updates will

have already been delivered to the client and are ready to be integrated.

The updating and rendering processes are explained in: the Rendering Process in Algorithm3,4 and 5. The *QueryProcess* is designed to send the current view parameters to the server and to receive the LoD_i updates after the base map LoD_0. In the Rendering Process instead of setting the view parameters to the current view, we use *predictview* for the *currentview* and the map will be rendered after all data is updated. We use "NOT APPLIED" to check if all the updates from the server have been performed before rendering the map (in Algorithm 3). The Query Process uses "NOT SENT" (line 3 in 4) to ensure that all of the view parameters are sent to the server only once.

As stated above, we have used a modified view-dependent strategy to manage the data in mobile clients for the *UpdatingProcess* in Algorithm 5. However, the modifications make this approach slightly different to the one outlined in Section 5.4.5. As shown in Algorithm 5 we use a priority queue to organize the temporary *LoD* datasets received from the server (lines $1-2$ in Algorithm 5). The priority queue is used to manage the spatial data in memory. We are not increasing the *LoD* in the user's current viewing area. However, as the user moves around the map area, we are decreasing the spatial detail in irrelevant areas. If the object P_i is in the screen area, we apply the "increasing LoD" process (lines $3-5$ in 5). Otherwise, we will decrease the details which are now considered to be irrelevant data (lines $9-11$ in Algorithm 5). Meanwhile, this process will continue to check that the spatial data stored in the memory are still necessary. This step seeks to maintain the spatial data in the limited storage available. If there is the data, which is seen as unnecessary, then it is removed from memory storage. By using these checks, we can achieve an efficient usage of system resources while speeding up the overall rendering process.

We provide an example of this process in operation in Figure 6.4. The example in Figure 6.4 uses real data from OSM as shown in the screen shot of the map in Figure 6.3. The generalized map L_0 is shown in Figure 6.4A. In Figure 6.4B, we see the sequence of operations on the map where the user's current viewing area is outlined (receiving increasing LoD). The previous view is being updated to remove some of the now unrequired data in some of the objects. Using this user motion prediction, the data for *LoD* for the next view are being pre-fetched.

The "previous view" will decrease detail by applying $UpdatingProcess$ when users are looking at "current view" where the LoD are being increased. The $Pre-fetchingProcess$ is now fetching new spatial data from server for the "next view" which currently is still rendered in low resolution.

Algorithm 3: This is the algorithmic procedure for the rendering thread working in the mobile client application as the user interacts with the map visualisation

```
1  Map initialization: receive basemap (LoD_0);
2  Calculate currentview of the map;
3  set view ← currentview;
4  repeat
5  |  interact;
6  |  if Updates NOT APPLIED then
7  |  |  set view ← predictview;
8  |  |  apply updates U;
9  |  end
10 |  render the map;
11 until;
```

Algorithm 4: This is the high level algorithm for the 3 components working in parallel: sending, receiving, and updating

```
1  q = ∅ (q is a priority queue for storing LoD) ;
2  repeat
3  |  if view NOT SENT then
4  |  |  send view;
5  |  |  receive LoD;
6  |  |  set q ← updateLoD;
7  |  end
8  until ;
```

Algorithm 5: The algorithm describes the "updating" where the decision is made to increase or decrease LoD and to discard objects which are no longer relevant

```
1  while q ≠ ∅ do
2  |  Dequeue q;
3  |  if P_i within currentview and L_i is lower than FullLoD_i then
4  |  |  Increase LoD;
5  |  end
6  |  else
7  |  |  Discard P_i;
8  |  end
9  |  if Q_i not within currentview and L_i is higher than L_0 then
10 |  |  Decrease LoD;
11 |  end
12 end
```

As discussed above and depicted in Figure 6.4, our approach on the client attempts to predict where on the map the user will move to next. If we can predict with satisfactory accuracy, the location the user is most likely to move to next then the prefetching process can begin to download increasing LoD to enhance the base-map L_0 for that area. The strategy for user motion prediction will now be described.

6.4 User motion prediction and prefetching of updated LoDs

When the users browse online maps, they usually have own favourite ways to move around the map. Some users prefer to use the controls on the map container to pan and zoom to a specific area of the map. Others prefer to take advantage of the "slippy" map and slide and drag the map around. However, when we are performing a task such as locating a specific building on a web-based map or trying to plan a route along a set of roads our movements are usually much slower and carefully planned. In this section, we take advantage of this behaviour

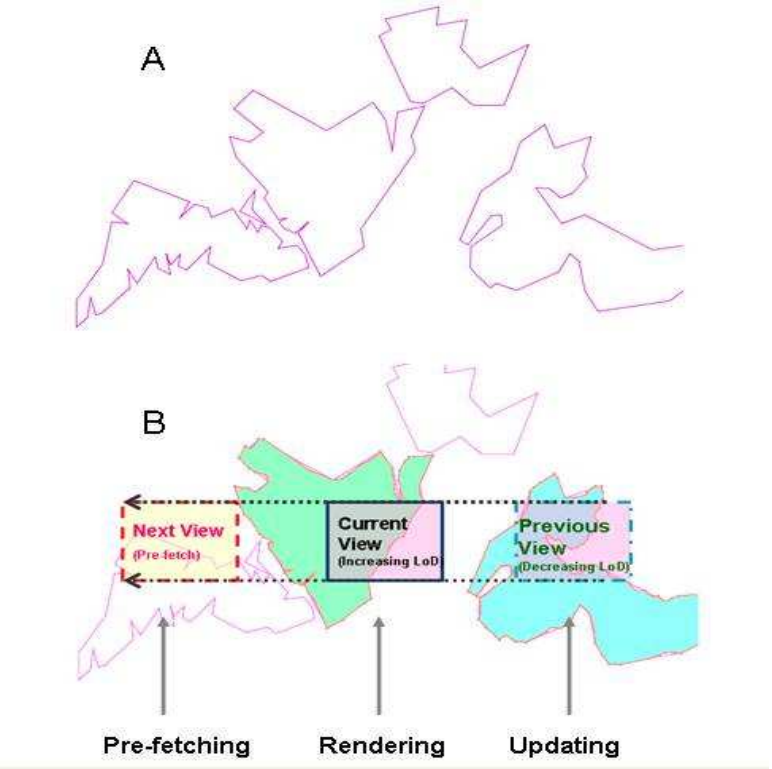

Figure 6.4: This example shows that after the user received a generalized map as a L_0 (in A), they can move the view from the "previous view" to the "current view" (as in B). The $RenderingProcess$ is rendering the current view at this time. In parallel the $UpdatingProcess$ is decreasing the LoD in the "previous view". The $PrefetchingProcess$ is requesting the data for the "next view" according to the prediction of the next location that the user will move to

in attempting to predict the motion of the user movement of the mobile map. Given the user's current AOI, there is a great benefit to be obtained in predicting their next movement and then pre-fetching the spatial detail required for the corresponding area on the map. As most interaction is performed with the mouse (excluding the increasing number of touch-pad interfaces), authors such as Tahir et al. [2011] have studied the movement of the user's mouse during spatial tasks. They automatically identify and validate a number of spatial tasks forming complex mouse trajectories. Using these trajectories Tahir et al. speculate that these could help in enhanced map personalization.

In our implementation, when a user interacts with the map on the mobile client, the client software sends a request to the server with view parameters *view*. These parameters are encoded in JSON format. The view parameters typically include screen position, view direction, the zoom level, and the screen-space error allowance. Since each zoom level has a different scale corresponding to a fixed error allowance (see section 3.4 for an explanation of the error transformation), we only consider the parameters for screen position and view direction in this section. These two parameters are constantly changing when the user interacts with the mobile map. The mobile screen can refresh once per second or faster when rendering the map data. This is usually called the "framerate" in computer graphics [Funkhouser and Séquin, 1993]. The interaction pattern of viewing a mobile map can be approximated as a piecewise set of straight lines connected by halts and abrupt changes in movement of speed or direction. Within a very short time (several frames), the interaction pattern can be approximated as uniform motion in a straight line. The display for future frames can be extrapolated based on the screen view of the current frame and that of the previous frame. This method is adapted from the first-order dead-reckoning algorithm used in distributed simulations for digital environments [Chan et al., 2001, 2005; Duncan and Gracanin, 2003]. The formulation which has been adapted from the work of Chan et al. [2001, 2005] is outlined as follows.

$$F_{view} = C_{view} + (C_{view} - P_{view}) \tag{6.1}$$

The next future view F_{view} can be calculated by the first order dead-reckoning

algorithm with the direction from the previous view P_{view} to the current view C_{view}. Due to temporal coherence this prediction algorithm works well for regular movement patterns during interaction [Scherzer et al., 2011]. Temporal coherence (TC) is the correlation of contents between adjacent rendered frames, it exists across a wide range of scenes and motion types in practical real-time rendering. By taking advantages of TC, we attempt to save redundant computation and improve the performance of the rendering task significantly, even if there is an abrupt change in speed or viewing direction resulting in a large prediction error, this error will be corrected in the next few frames as long as the proceeding changes in the view parameters stay relatively stable for a few frames. Dix et al. [2003] demonstrates that this is usually the case in human interaction with digital environments. Additionally, equation 6.1 is easy to calculate and it provides a good trade-off for resource constrained mobile clients.

The traversal constraint on the LoD update in each frame can limit the influence of any large prediction errors. When interaction is quick, the view-dependent progressive refinement/simplification caused by the error of the predicted view parameters will be limited. The human vision system cannot resolve small details changing during very fast motion [Luebke et al., 2002]. Figure 6.5 shows an example of this user motion prediction. In the figure, the user is currently at position 2. This was an abrupt change from their previous position 1 and the predicted screen position (position 2 as a square icon) deviates from the actual position. However, the prediction accuracy increases in the following frames as the screen position changes steadily. This type of user interaction is handled well by this simple predictive model.

In this section, we have described a user motion prediction model. Earlier in the chapter we also described in detail our parallel approach to prefetching LoD and also removing irrelevant objects from the memory of the client device. In the next section, we will outline some examples of how this interactive rendering approach of using a multi-threaded architecture works in practice and what the actual quantitative benefits are.

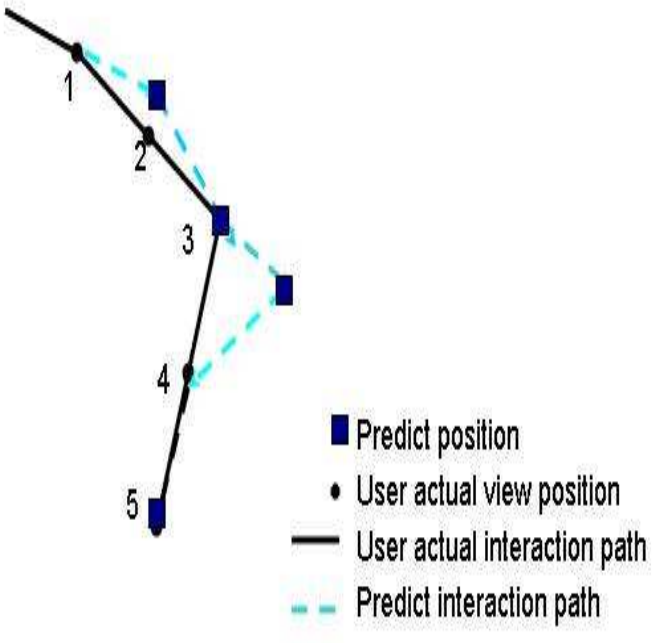

Figure 6.5: This is a viewpath example for the prediction of the user motion interaction path. This is the practical implementation of the view prediction model outlined in equation 6.1. In the figure, user movement starts at 1 and moves towards position 5.

6.5 Practical evaluation of this approach

In this section, we provide a practical evaluation of the performance of the strategy proposed in Section 6.3. The first experiment is designed to compare the sequence approach and the parallel strategy in terms of overall map rendering time. In the second evaluation, we will show an example of data usage in the client side view-dependent approach when the updating component is in operation. This will demonstrate the efficient data management strategy employed and we will discuss the amounts of data received by the client.

Our implementation setup is the same as was described and used in the previous chapter (see section 5.5 on page 131). The server runs on a Pentium 2GHz PC and the client runs on an Android 2.2 HTC Desire smart mobile phone. The client and the server are set up on a wireless network with a 100 kb per second connection. The OSM datasets are extracted as specified in a rectangular bounding box and we have set $n = 3$ steps for LoD. This is visible on the $x-$axis of the graph, as shown in the Figure 6.12. The corresponding OSM dataset used is displayed in Figure 6.6. A mobile visualization example for this map is shown in the Figure 6.14.

We use the same example map dataset to help illustrate the step-by-step concept of our rendering approach in a consistent manner. The initial map in Figure 6.8 on page 165 is simplified from original datasets (in Figure 6.7 on page 164). This map is sent to the mobile user to start the interaction. The data is transmitted by the progressive transmission approach which we proposed in the previous chapter (Chapter 5). The user starts to browse the map from the top left of the map as shown in Figure 6.9, on page 166, and follow the preset path. The panning interaction is finished at the right bottom of the map and there are two turns or changes in the direction during the time the user is following this path. As shown in the Figure 6.9, on page 166, additional spatial detail is sent to the user's area of interest along the path. Along this path, the corresponding polygons obtain significant refinements when they in this area visible to the user. However, as long as the user's interacting with the map, the data is being received continuously. The amount of this data is increasing in the main memory of the mobile device as these additional refinements are received.

This is where the proposed approach can more efficiently manage the *LoD*. As shown in the Figure 6.10, on page 167, when the user begins to pan and move around in the map area, this approach not only increases the spatial data in the user's current view, but at the same time, decreases or removes spatial data from the old or previous view. In this way increasing *LoD* will not cause an overload of the main memory when the user performs multiple interactions with the map visualisation. Moreover, this approach discards the refinements of *LoD* in the historical path if it is sufficiently far from the user's viewing area as shown in the Figure 6.11 on page 168. Overall this approach can significantly enhance the efficiency with which data management is carried out in low-end mobile clients.

In this practical example, the overall processing time is measured against the total amount of data (in terms of nodes). This is plotted and illustrated in Figure 6.13 on page 170. It is clear in this illustration that the removal of *LoD* from areas now considered irrelevant has a very beneficial effect. These areas are no longer required at high detail because the user has moved to a different area on the map. The rate of change of data caused by user movement means that the client side of our view dependent approach must maintain the data in memory. In Figure 6.12, on page 169, the frame sequence is plotted against the overall time to process the frame for both the sequential default method and then the parallel rendering method. This plot shows that the parallel strategy can render a frame in a much shorter time during progressive transmission while the sequential method requires a longer frame processing time. It is clear from the plot that the parallel method takes less overall time to render the frames than the sequential method. The progression of the *LoD* increases from the server when there is interaction from the user changing their context on the map screen view (such as performing a panning operation like Figure 6.4 on page 157). The sequence method must wait for the complete download of the *LoD* to complete. The parallel method can prefetch the datasets potentially within the next view. This allows the forward prefetching of data meaning that the data for the next frames the user requires will be delivered and ready for user viewing.

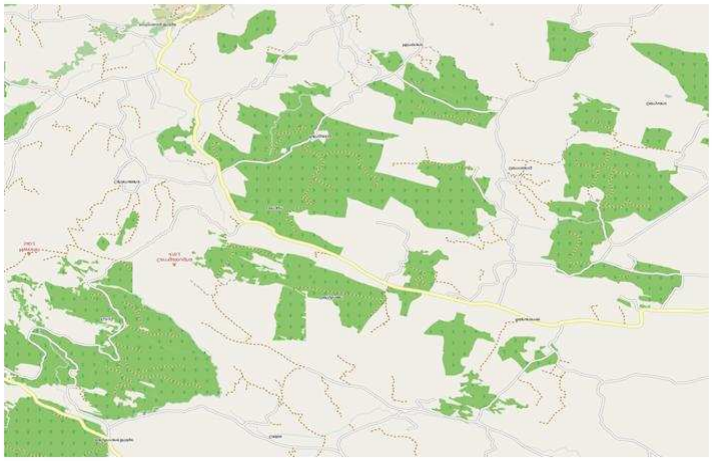

Figure 6.6: This is a screenshot of the OSM area used for the analysis in Figure 6.12 and Figure 6.13

Figure 6.7: This figure shows the full detail vector map generated by our progressive transmission approach for the map example in Figure 6.6

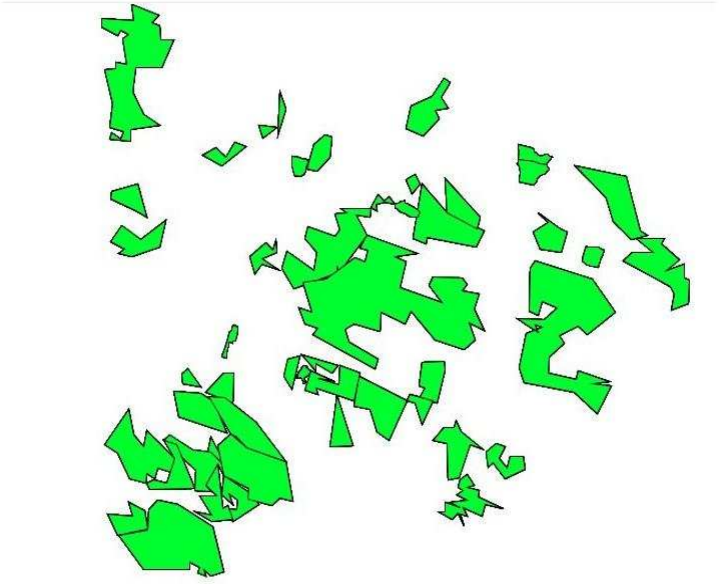

Figure 6.8: This figure shows the initial map with simplified data, which is sent to the user. This map is simplified from the data in the Figure 6.7

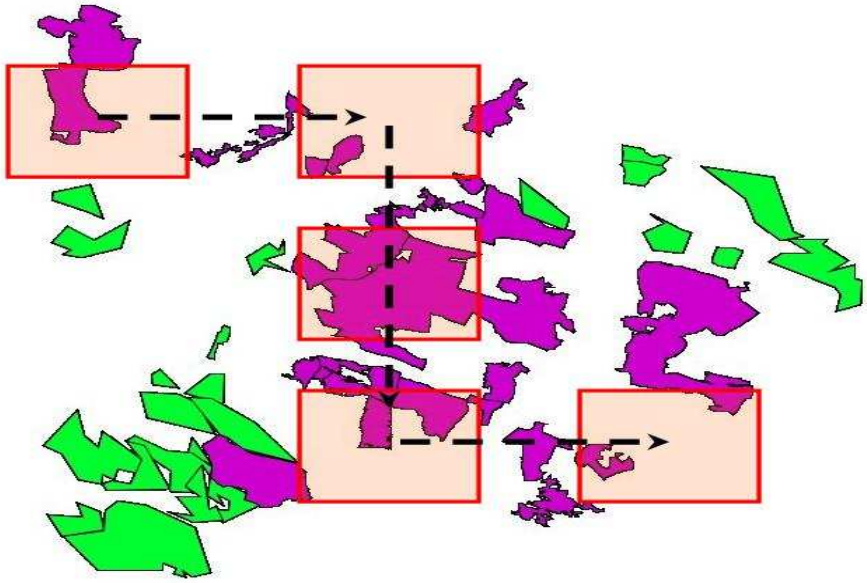

Figure 6.9: This figure shows the user panning the map along the pre-set path (the red box). The details are obtained within the path (shown in the light red color). This additional spatial data is received from the view-dependent progressive transmission scheme which was described in Chapter 5. The other areas, currently not relevant, do not obtain additional details and are shown in green color

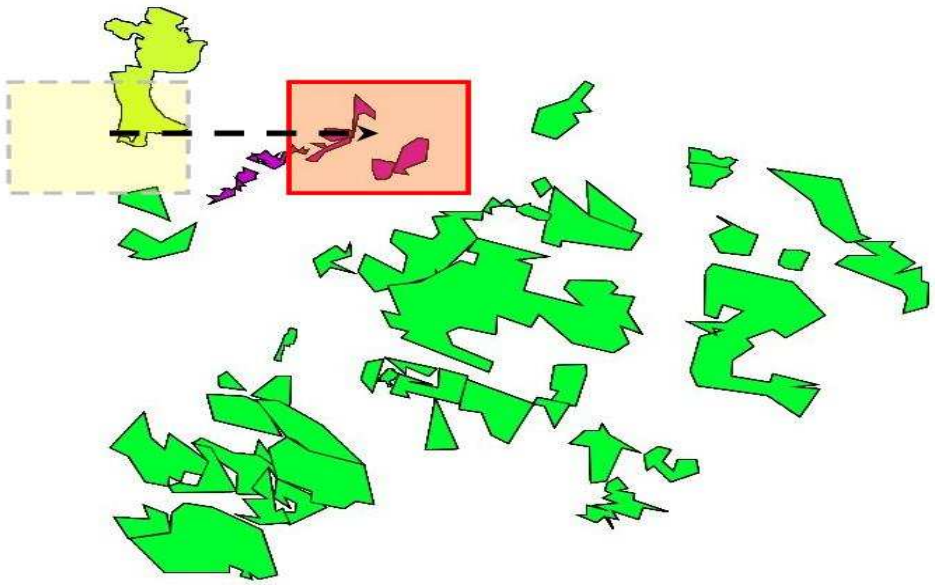

Figure 6.10: This figure shows the effect of the proposed rendering approach. Not only can it increase the details in the user's current viewing area (in the red box), but it also decreases the details in the previously viewed areas (in the yellow box). The polygons in the yellow box (in an orange color) are maintained with less spatial detail than the current view. The current view will continue to gain more spatial detail (in the red color)

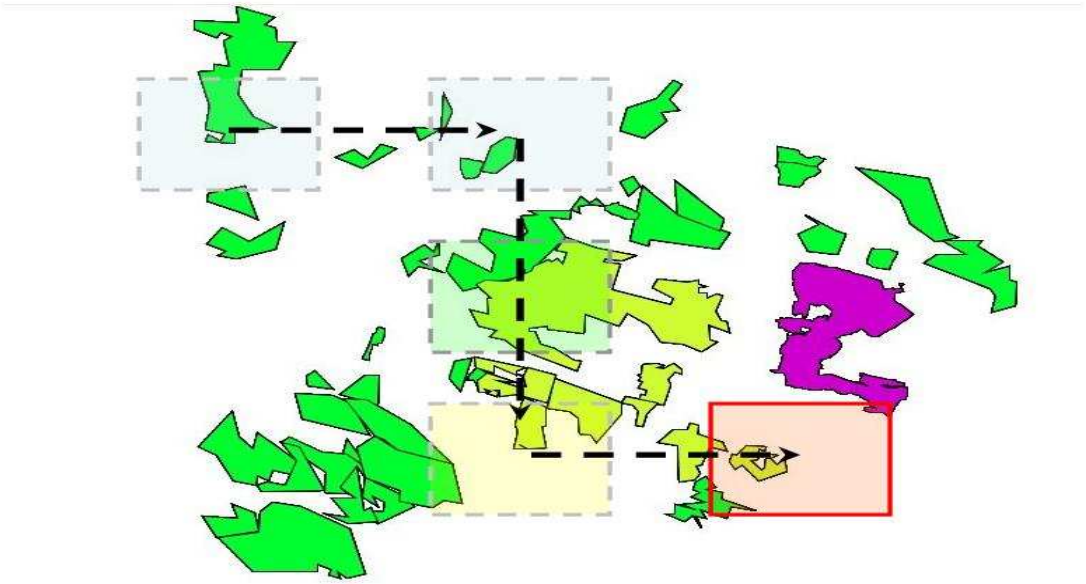

Figure 6.11: As the user pans around the map, they leave a historical trail of areas which they have visited or viewed on the map. This figure shows that this approach can also discard the details (in the gray boxes) in the historical path which are now sufficiently far away from user's current view.

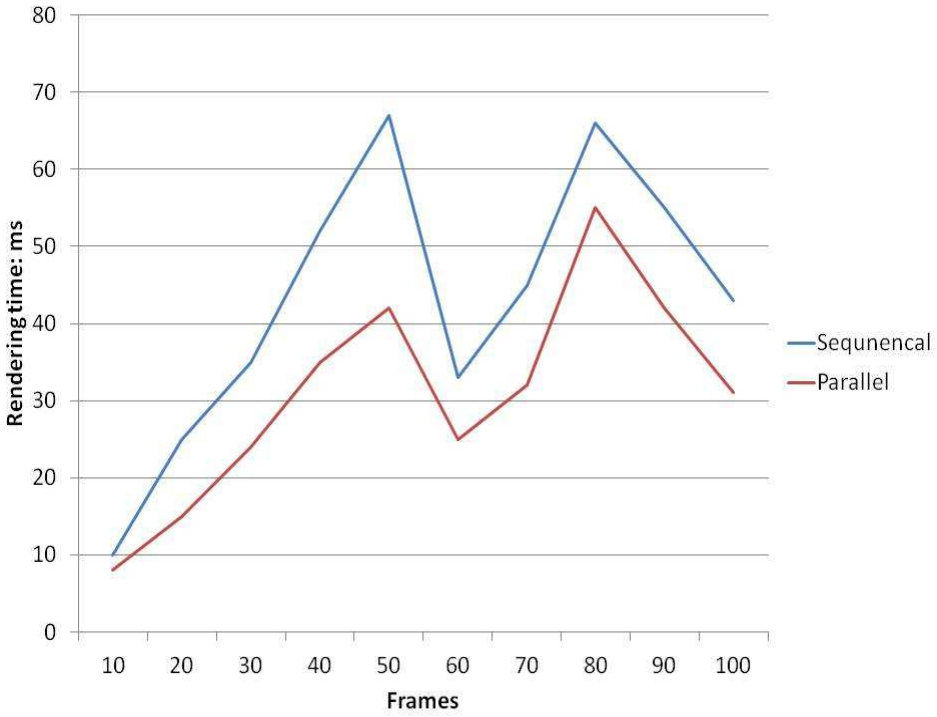

Figure 6.12: In this plot, we see a comparison of the sequential method of *LoD* delivery against the parallel method of prefetching data for the predicted area that the user will eventually move (pan) to in the next few frames.

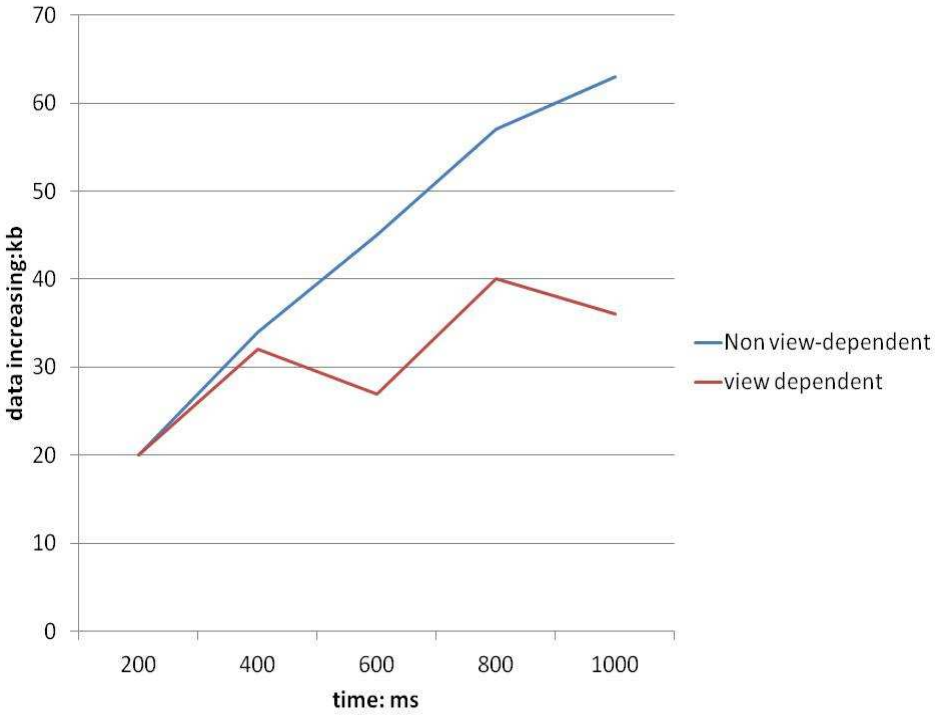

Figure 6.13: This figure shows a plot of the overall data consumption rates against the overall time required for rendering. There are two user movement direction changes in the example where the user moves to another area of the map. In this case, the resolution of now irrelevant areas is decreased.

Figure 6.14: This is an example of the actually rendering process running on a real mobile device. This photograph is of the implementation of this approach for map of in Figure 6.13 on page 170

6.6 Styling the map visualisation

As shown in Figure 6.6 the map is a simple map in terms of the types of features involved. The most frequently occurring features in Figure 6.6 are green spaces, agricultural fields, and forestry. There are also some water features. These features are coloured 'green' and 'blue' respectively. We can easily access the semantics with the OSM data. For each object in OSM, there are 'tags' or attributes attached to the spatial data describing that object. These tags can be used to add specific styles to the map by changing the colouring of features. This is based on the rules generated by analysing the types of features contained in the map dataset.

In our software implementation, the the complete collection of data features in OSM data is transmitted in JSON formats. JSON format is a common standard for structured data, which is more compact than XML. However, it is a light-weight data format for exchanging data between clients and servers in web applications using Javascript. GeoJSON allows geographic data to be stored in a human-readable way. GeoJSON generators and parsers are available for most leading databases and programming languages. This enables the application server to send spatial data and associated semantic metadata in this format extracted from the features in the OSM XML (see Figure 6.15 on 174). Figure 6.15 shows a very simple piece of OSM XML transformed to JSON which is then transformed and bound to Java objects. The main features of the OSM data contained in this example include: way id (polygon or polyline), location information, OSM feature and name. The rendering function in the mobile clients will convert this format directly to Java objects for integration into our Java code for the reconstruction of the map in visualisation on the mobile clients. As the simple OSM example in Figure 6.15 shows the data in OSM, in the left hand side, is transferred (after the generalisation) to JSON formatted objects. The JSON is then sent to the client and rendered in the mobile client's visualisation application software.

All the spatial information encoded in the JSON objects will be converted, or bound, to Java objects. The attributes of the features of the map are encoded into the JSON object. As discussed these are available in the form of tags in

the OSM XML which are expressed in value pairs such as k = 'amenity' and v='public building'(discussed in Section 4.1 in Chapter 3). These attributes become attributes of the objects in Java. The proposed software tool then uses these attribute values to perform rule-based rendering to the corresponding features (i.e. "Lake" should be rendered in a blue color). However, the focus of this work is not to generate very aesthetically-pleasing map visualisations. There are many freely available style layers available which could be used to make this visualisations "look-and-feel" like an OpenStreetMap map rending. We have used a reasonably simple rule-based rendering approach which can be easily extended if necessary. An example of two different methods of map styling is shown in Figure 6.16.

6.7 Discussion and potential applications

Up to this point in this thesis our primary concern was the delivery of the spatial data from a source (dynamic database or Internet API) to the server for further processing. On the server, the processing is performed to prepare the selected data package for progressive transmission to the client. This chapter has focused on the client side rendering techniques as the end point for the progressive transmission of the spatial data from the dynamic database or Internet API. We introduced an efficient strategy for the parallel rendering of the map visualisation in the client while at the same time progressively transmitting data for the next LoD or next area of interest. This parallel strategy involved the development of a prediction algorithm to determine, with high accuracy, where on the map the user was likely to move to next. This concept was illustrated in Figure 6.5 and its evaluation was discussed in Section 6.5. This parallel scheduling process was used with a view-dependent progressive transmission. It was shown to have similar performance to a scale based approach while reducing the amount of data sent in each frame. This makes this approach very suitable for the visualisation of large spatial datasets in low capacity mobile clients. In Section 6.6, we outlined methods to style the map visualisations by accessing the attribute information of the spatial features transferred to the mobile client during the progressive transmission process.

In Chapter 7, we will outline a summary of the contents of thesis. We will

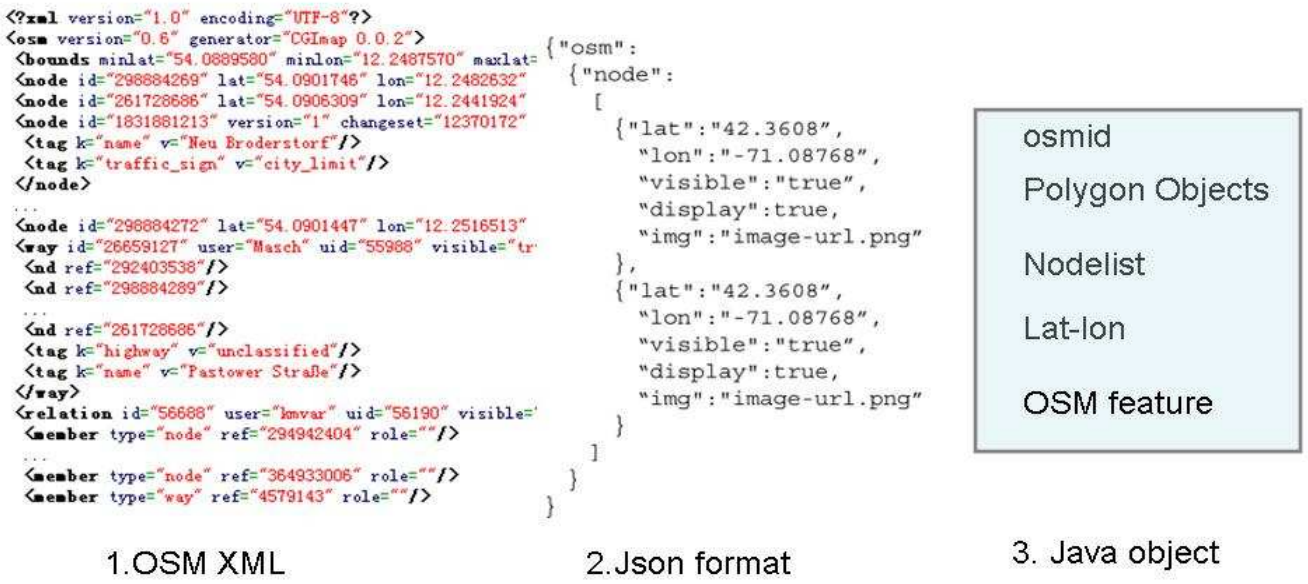

Figure 6.15: This figure shows a very simple piece of OSM XML transformed to JSON which is then transformed and bound to Java objects

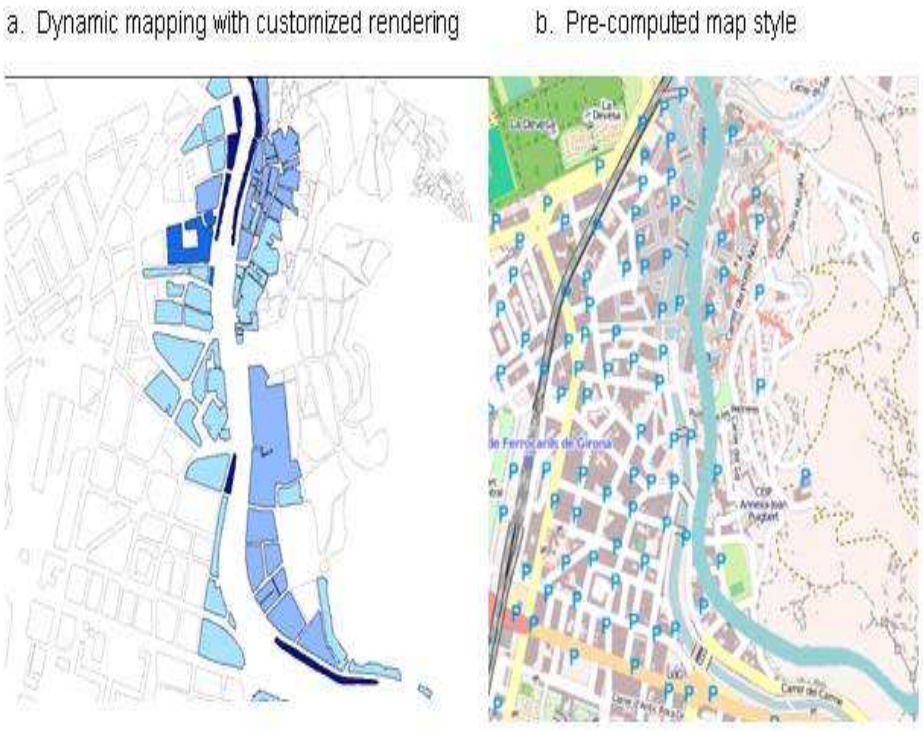

Figure 6.16: This is an example of flood analysis in the city river of Girona Spain, which is visualised using (a.) dynamic visualization with customised styling and (b.) pre-computed visualization with tile-based maps

investigate how we have successfully delivered the research contributions outlined in the first chapter in section 1.3. This will allow us to put these results in the context of the current state of the art in the literature. We close the thesis by outlining some opportunities for the further research work on this topic for the immediate and long-term future.

Chapter 7

Conclusions and discussions

In this thesis, we developed a framework to facilitate the management of vector-based spatial data both in server and mobile clients for the visualization in LBS. The development of this framework has taken account of the constraints inherent in mobile devices (e.g. low network communication, limited storage, small screen display). Our framework focusses on dynamic spatial data sources such as VGI data. In Chapter 1, we have outlined the key research questions related to the visualization of dynamic spatial datasets in mobile applications. To address these questions, we have conducted a series of research includes:

- Identified the key challenges related to managing dynamic spatial data (e.g. VGI data) in LBS and investigated current solutions related to vector based map visualisation in chapter 2;
- Proposed a server side data processing framework to handle the vector data in incremental LoD in Chapter 3;
- Implemented an opensource software based on the data processing framework of Chapter 3 and used OSM datasets as a case study in Chapter 4;
- Developed a selective progressive transmission strategy based on a heuristics model in Chapter 5; Also integrated the view-dependent technique to obtain a smoother transmission of suitable LoD refinements;
- Developed multi-thread-based data management strategy associated with

the algorithms to speed up the data visualization and to prevent the overload of the irrelevant data in low-end mobile client in Chapter 6.

In this chapter, we will summarise the main achievements of this thesis and provide some insights from user trials and experimental evaluations. The limitations of this work are also discussed. Finally, we outlook the future work with a number of interesting research directions.

7.1 Overview of research achievements

The vector-based spatial data visualization as an alternative solution to traditional tile-based solutions is emerging in LBS applications. Dynamic crowd sourcing data sources (e.g. VGI data) are potential data resources to provide timely and accurate spatial information for LBS. However, for most mobile users, it is still difficult to access the large amount of online spatial data due to many constraints of mobile devices. In order to address these issues, this thesis has developed a server- client data management framework for handling the vector data as LoD on both server and client side. The main contributions of this work can be described as follows:

1. **A framework for spatial data processing:** Our framework provides an improved solution for processing vector-based spatial data on an application server. Processing vector data is a complex process comprising of many steps and components. We subdivided this complex process into individual components to perform: spatial data simplification; coordinate transformation; general data handling, and consistency checking. We apply "on-the-fly" data representation checking and data simplification. By carefully considering the representation of the spatial data, it applies data simplification only if necessary. This results in significant reductions in the data volumes. This also provides users with access to their chosen spatial datasets without an unacceptable reduction in the overall quality of those datasets. To ensure that the visualization of these datasets on the mobile device screen is of acceptable cartographical standard we apply scale trans-

formation metrics to transform these datasets to the mobile screen coordinate system. These metrics formularized the transformation between the error threshold in data simplification and the user's visual error allowance within the available mobile screen area. To support this framework, a low redundancy data structure was designed to handle the spatial data in incremental LoD. A priority queue is used in the management of LoD after data simplification. Crucially for multi-layer datasets, our framework supports the integration of topology consistency checking which ensures enhanced data accuracy after generalization processing [Corcoran et al., 2012; Zhou et al., 2005].

2. **An implementation of the software tool:** A software architecture is described in the implementation the data processing framework. The software can handle vector data extraction directly from a database or spatial data which is being streamed "live" from a dynamic data source (e.g. OSM databases). Data simplification techniques are implemented in this software to reduce the amount of data in the processed datasets. The geographical data features in the datasets are examined by this software tool. In the example of using OSM data, this software tool can extract the spatial information of the OSM dataset "on-the-fly". This approach is generic, which allows one to use vector data of other formats such as: geoJSON, ESRI Shapefile, KML, etc. A comparison with traditional tile-based map processing demonstrates the advantages of this software tool for handling spatial data on the application server without the requirement to manage extra caching of map tiles. The software provides a solution to enable quick access to dynamic vector datasets.

3. **Selective progressive transmission:** We proposed and implemented this progressive transmission scheme based on a series of user studies. We conducted user trials based on the user's satisfaction ratings with maps transmitted by a progressive transmission technique. This progressive transmission technique was a straightforward reversal or unwrapping of the order of which generalization had processed the map dataset. Many authors in the literature show that human vision plays a very important role in map us-

ability. The users satisfaction results in user trials demonstrated that there was potential to improve the map visualisation during progressive transmission by performing a selection on the set of LoD. A number of shape complexity metrics were investigated in the user trails of OSM datasets in order to be able to automatically assess the complexity and representation of the dataset before transmission. To extend this approach, we reviewed various techniques from the literature and attempted to optimize the progressive transmission process. We developed a view-dependent progressive transmission approach based on efficient spatial-indexing techniques to select and deliver spatial data within the user's requested view area. This approach significantly reduces resource consumption on both client and server when compared to full scale progressive transmission [Corcoran et al., 2011a]. Building on this view-dependent approach, we then described and implemented a cost-benefit heuristic model by altering our approach based on shape complexity metrics and the view-dependent technique. The cost-benefit approach optimizes the LoD data delivery by sending the most significant map features with priority in a limited bandwidth situation. The cost-benefit approach attempts to maximise the smoothness of transition between LoD whilst minimising the amount of data to be transmitted. An experimental test is conducted in the context that user is viewing the area with many complex spatial objects with various representations. The experimental results demonstrated the advantages of this approach over other types of progressive transmission when delivering spatial data over low bandwidth communication networks.

4. **A parallel strategy for data management on the client mobile device:** The design of this client-side data management strategy has taken account of the limitations of low-end mobile clients (e.g. limited data storage capacity, tight computational resource budget and low network bandwidth). To overcome the tight computational resource budget, the sequence of processes involved in visualization are optimized into a multi-threaded parallel approach. This multi-threaded data management scheme has been designed for the mobile client to schedule data downloading and data rendering in

parallel in order to achieve improved processing time for data visualization on the client. A set of algorithms have been developed corresponding to the threads in this multi-threaded approach. These algorithms correspond to: visualization, data updating, and fetching data of the next LoDs. This algorithmic approach is used in conjunction with a view-dependent progressive transmission scheme in the client in order to decrease the amount of data transmitted for irrelevant areas of the map visualisation while also avoiding the overloading of the phone memory and disk space with spatial data. The pre-fetching thread helps overcome network latency while also speeding up the data visualization process. The updating thread maintains the relevant LoD in memory. This prevents the overload of memory resources when continuously downloading the increasing LoDs. The rendering thread can handle user interactions and render maps based on user preference on the fly (e.g. customization of the color style). This multi-threaded parallel approach can be potentially applied to many geoprocessing applications to visualize dynamic geoprocessing data effectively.

7.2 Suggested improvements

In practice, the LBS applications will be operated in more complex scenarios and the mobile users are performing many parallel tasks rather than just looking at the maps [Steiniger et al., 2006]. It is difficult to satisfy all the requirements in many dimensions of user requirements. For this thesis work, we only focus on development of a framework for facilitating the data management for visualization in mobile devices. This work provides an enormous opportunity to use vector based data as a data resource to derive more complex and interactive LBS applications. Nevertheless, this research work has the following limitations:

In this work, we proposed a framework for efficient processing of vector-based spatial data (Chapter 3). There are four main constraints of generalization: shape constraints, semantic constraints, metric constraints, and topological constraints Weibel [1996]. We have made efforts to balance the trade off of the visual importance of the shape of features (shape constraints) and the simplification algorithm

performance (metric constraints). We have not considered other complex data generalization techniques such as those involving semantic intelligence with the semantic process of Kulik et al. [2005] being one such example. This is because there are different types of vector data sources, the semantic rules for performing generalization according to the non-spatial attributes (metadata) are very much unique to each dataset. Consequently these semantic rules must be considered on a case-by-case basis depending on whether the data source is OSM or a National Mapping Agency or a sensor network. We have examined the semantic features of OSM in metadata. Other authors have also looked at this problem as Ballatore and Bertolotto [2011] finds that the rich non-spatial features are very complex. This makes having a uniform generic strategy in map generalization according to a specific semantic attribute or feature difficult.

In the mobile contexts, users may not be able to focus on the maps and the resolution of the screen is reasonably low. We performed data simplification to reduce data volume, which allows users to access the data more efficiently on the understanding that it might not be the highest quality resolution. This was shown to work effectively. However, we discussed topological consistency issues arising in generalization. Efficient topological consistency checking is still a difficult question in this research area [Bertolotto and Egenhofer, 2001; Han et al., 2004]. In the development of the selective progressive transmission approach (Chapter 6), we examined a number of criteria (e.g. area importance, semantic importance, etc). In the end, we choose the metrics of shape complexity as the criteria for the heuristic model. We considered that these features can directly affect the user's experience and perception of the map particularly during the transmission of complex spatial objects. We could consider more potential criteria to improve the map usability based on different user preferences in the future.

The proposed mobile client data management strategy has considered the many limitations of mobile devices. However, there are more constraints, which must be considered in practice for devices operating in a ubiquitous computing environment [Franklin and Wilhelm, 2000]. To achieve better visualization results our user's motion prediction scheme could potentially consider more complex gestures (e.g. multi-touch, grab, etc). A user-friendly interface could be design to assist the user to navigate to the area of interest efficiently.

7.3 Conclusion and future work

In this thesis, a framework has been developed to better manage vector-based spatial data for the visualization in LBS applications on both the server and client side. Our framework is examined and assessed with real-world datasets from OSM. Our experimental analysis indicate that the framework works very well with low-end mobile clients. We feel that this research work has made valuable contributions in the area of managing dynamic vector data in the server side and LBS mapping and visualisation on the client side. Despite the discussion of the achievements of this work, there are some areas which will require future research work.

In this thesis, our data reduction techniques focus on the geographical objects and their shape analysis. However, data reduction techniques could be extended to include the personalization of spatial contents by taking user profiles and user interactions into account [Mac Aoidh et al., 2012]. Irrelevant spatial contents could be omitted before the transmission which would also result in a more efficient generalization process [Bertolotto and McArdle, 2011]. In VGI, metadata contains various non-spatial information (e.g. names, tags, etc). The data simplification process could be extended to provide a personalization solution allowing the selection of spatial objects matching user preferences before transmission. For example, tourists require maps which highlight landmarks and the points of tourist interests on maps. A personalized map could deliver more readable spatial contents in this types of context [Agrawala and Stolte, 2001]. These personalization techniques should be integrated into the generalization process to provide more relevant spatial contents to the mobile users in LBS while also benefiting from improved performance.

In chapter 5, we investigated a number of shape analysis metrics and we selected the metrics, which are most closely related to the human visual system as the criteria for the selective progressive transmission. More complex user studies could be designed to examine the more complex metrics.

As we discussed in chapter 6, mobile client applications associated with our framework could potentially work with spatial data which is generated in real time such as that from a geosensor network. Our real time rendering mechanism

can provide customized rendering styles for the different features. A user interface could be developed in the future to allow users to configure their preferences. This approach could be operated in real world geoprocessing applications to assist in development of more meaningful and interactive spatial information. HTML5 has shown its potential, as a cross-platform technique, to visualize vector-based spatial data effectively in the web browser of any mobile devices [Boulos et al., 2010; Corcoran et al., 2011b]. As $4G$ Internet becomes widespread, the mobile visualization of spatial data is predicted to offer more interaction with semantically rich spatial data. As a consequence, the data generalization processes on the server will also become more complex to adapt to these changes. In our work here we emphasised that visualisation of spatial data on mobile devices was fundamentally different to the same task on desktop computers and in GIS. The new generation of tablets such as the iPad have a relative larger screen and much high specifications than smartphones. This must be considered in the future work, where client devices will come with a variety of different screen sizes.

With the continued success of geovisualization in LBS, vector-based mapping techniques are becoming a very popular solution. These techniques have many advantages over the traditional mapping techniques for providing interactive and high quality spatial information. This work has achieved a framework to address several key issues related to the visualization of vector-based spatial data in LBS. This thesis provides a strong treatment of this topic, discusses promising research results, and outlines future research directions for the topic. Research efforts should continue in the provision of solutions for visualization of spatial data LBS as the technology of client devices changes rapidly for the better.

References

Maneesh Agrawala. *Visualizing Route Maps*. PhD thesis, Stanford University, 2002. xi, 39, 40

Maneesh Agrawala and Chris Stolte. Rendering effective route maps: improving usability through generalization. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 241–249, New York, NY, USA, 2001. ACM. ISBN 1-58113-374-X. 5, 40, 183

Tinghua Ai. Constraints of progressive transmission of spatial data on the web. *Geo-spatial Information Science*, 13(2):85–92, May 2010. ISSN 1009-5020. 47

Tinghua Ai, Zhilin Li, and Yaolin Liu. Progressive transmission of vector data based on changes accumulation model. In *Developments in Spatial Data Handling: 11th International Symposium on Spatial Data Handling*, 2005. 114, 129

Paul Anderson, Mark Hepworth, Brian Kelly, and Randy Metcalfe. What is Web 2.0? Ideas , technologies and implications for education. Technical report, JISC, 2007. 21

Gennady L. Andrienko and Natalia V. Andrienko. Interactive maps for visual data exploration. *International Journal of Geographical Information Science*, 13(4):355–374, 1999. 28

Vyron Antoniou, Jeremy Morley, and Mordechai M. Haklay. Tiled Vectors: A Method for Vector Transmission over the Web Web and Wireless Geographical Information Systems. In James D. Carswell, A. Stewart Fotheringham,

and Gavin McArdle, editors, *W2GIS 2009*, volume 5886 of *Lecture Notes in Computer Science*, pages 56–71, Berlin, Heidelberg, 2009. Springer Berlin / Heidelberg. ISBN 978-3-642-10600-2. 5, 14, 47, 130

OSM API. Openstreetmap has an editing api for fetching and saving raw geodata from/to the openstreetmap database, 2012. URL http://wiki.openstreetmap.org/wiki/API. 79

Apple. How apple's new vector-based maps leave google maps looking jittery, 2012. URL http://appleinsider.com/articles/12/08/03/inside_apples_new_vector_based_maps_in_ios_6. 31

Lars H Arrie, L Tiina S Arjakoski, and Lassi L Ehto. A Mapping Function for Variable-Scale Maps in Small-Display Cartography. *Journal of Geospatial Engineering*, 4(2):111–123, 2002. 40, 41, 42

José Augusto, Sapienza Ramos, Claudio Esperança, Esteban Walter, and Gonzales Clua. A progressive vector map browser for the web. *Architecture*, 15(1): 35–48, 2009. 5, 47, 97, 130, 146

Andrea Ballatore and Michela Bertolotto. Semantically enriching vgi in support of implicit feedback analysis. In *Proceedings of the 10th international conference on Web and wireless geographical information systems*, W2GIS'11, pages 78–93, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-19172-5. 182

Thomas Barkowsky, Longin Jan Latecki, and Kai-Florian Richter. Schematizing maps: Simplification of geographic shape by discrete curve evolution. In *Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, pages 41–53, London, UK, UK, 2000. Springer-Verlag. ISBN 3-540-67584-1. xi, 53, 54, 56, 76

Ronen Basri, Luiz Costa, Davi Geiger, and David Jacobs. Determining the similarity of deformable shapes. *Vision Research*, 38(15 C 16):2365 – 2385, 1998. ISSN 0042-6989. 53

Michael Batty, Suchith Anand, Andrew Crooks, Andrew Hudon Smith, Mike Jackson, Richard Militon, and Jeremy Morley. Data mash-ups and the future of mapping. Technical report, JISC, 2010. 2

Bruno Becker, Hans werner Six, Fernuniversitat Hagen, and Peter Widmayer. Spatial priority search: An access technique for scaleless maps. In *In Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 128–137. ACM, 1991. 70

M. Bertolotto. *Geometric Modeling of Spatial Entities at Multiple Levels of Resolution*. PhD thesis, Department of Computer and Information Sciences, University of Genova, Italy, 1998. 73

Michela Bertolotto. Progressive Techniques for Efficient Vector Map Data Transmission: An Overview. In Alberto Belussi, Barbara Catania, Eliseo Clementini, and Elena Ferrari, editors, *Spatial Data on the Web*, pages 65–84. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-69878-4. 5, 31, 43, 96

Michela Bertolotto and Max J. Egenhofer. Progressive vector transmission. In *Proceedings of the 7th ACM international symposium on Advances in geographic information systems*, GIS '99, pages 152–157, New York, NY, USA, 1999. ACM. ISBN 1-58113-235-2. 97, 124

Michela Bertolotto and Max J. Egenhofer. Progressive Transmission of Vector Map Data over the World Wide Web. *GeoInformatica*, 5(4):345–373, dec 2001. xi, 5, 10, 44, 69, 97, 182

Michela Bertolotto and Gavin McArdle. Data reduction techniques for web and mobile gis. In Bert Veenendaal, editor, *Advances in Web-based GIS, Mapping Services and Applications*, chapter Chapter 8, pages 139 – 152. Taylor & Francis, 2011. 5, 183

Maged N Kamel Boulos, Jeffrey Warren, Jianya Gong, and Peng Yue. Web GIS in practice VIII : HTML5 and the canvas element for interactive online mapping. *International Journal of Health Geographics*, 9:1–13, 2010. 5, 28, 29, 40, 184

Thomas Brinkhoff, Hans Peter Kriegel, Ralf Schneider, and Ralf Schneider. Measuring the complexity of polygonal objects. In *Proceedings 3rd ACM International Workshop on Advances in Geographic Information Systems*, pages 40–49, Baltimore, MD, US, Dec 1995. 118, 120, 121, 122

Stefano Burigat and Luca Chittaro. Visualizing the results of interactive queries for geographic data on mobile devices. In *Proceedings of the 13th annual ACM international workshop on Geographic information systems*, GIS '05, pages 277–284, New York, NY, USA, 2005. ACM. ISBN 1-59593-146-5. 4, 44

Barbara P. Buttenfield. Transmitting vector geospatial data across the internet. In *Proceedings of the Second International Conference on Geographic Information Science*, GIScience '02, pages 51–64, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-44253-7. 5, 14, 44, 114, 124

Barbara P. Buttenfield and Robert Brainerd McMaster. *Map Generalization: Making Rules for Knowledge Representation*. London: Longman, 1991. 5, 39

Davide Carboni, Sylvain Giroux, Eloisa Vargiu, Claude Moulin, Stefano Sanna, Alessandro Soro, and Gavino Paddeu. E-mate: An open architecture to support mobility of users. In *Proceedings of the Baltic Conference, BalticDB&IS 2002 - Volume 2*, pages 227–242. Institute of Cybernetics at Tallin Technical University, 2002. ISBN 9985-894-40-5. 23

Cartegan. Cartegan - create vector maps with html5 & canvas, 2010. URL `http://cartagen.org/`. xi, 30, 91

Alesandro Cecconi and Martin Galanda. Adaptive Zooming in Web Cartography. *Computer Graphics Forum*, 21(4):787–799, December 2002. ISSN 0167-7055. 29, 39, 40, 69, 124

Alessandro Cecconi. *Integration of Cartographic Generalization and Multi-Scale Databases for Enhanced Web Mapping*. PhD thesis, University of Zurich, 2003. 10, 39, 69

Alessandro Cecconi, Robert Weibel, and Mathieu Barrault. Improving Automated Generalisation for On - Demand Web Mapping by Multiscale Databases.

In *10th International Symposium on Spatial Data Handling*, Ottawa, Canada, 2002. 39, 124

Addison Chan, Rynson W. H. Lau, and Beatrice Ng. A hybrid motion prediction method for caching and prefetching in distributed virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '01, pages 135–142, New York, NY, USA, 2001. ACM. ISBN 1-58113-427-4. 158

Addison Chan, Rynson W. H. Lau, and Beatrice Ng. Motion prediction for caching and prefetching in mouse-driven dve navigation. *ACM transactions on internet technology.*, 5(1):70–91, February 2005. URL http://dro.dur.ac.uk/612/. 158

Chern-Lin Chen. Computing the convex hull of a simple polygon. *Pattern Recognition*, 22(5):561–565, 1989. ISSN 0031-3203. 118

Wei Cheng and Wei Tsang Ooi. Receiver-driven view-dependent streaming of progressive mesh. In *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, NOSSDAV '08, pages 9–14, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-157-6. 37, 130

Chui Kwan Cheung and Wenzhong Shi. Estimation of the positional uncertainty in line simplification in gis. *Cartographic Journal, The*, 41(1):37–45, 2004. 120

Blazej Ciepluch, Peter Mooney, Ricky Jacob, and Adam C. Winstanley. Using openstreetmap to deliver location-based environmental information in ireland. *SIGSPATIAL Special*, 1(3):17–22, November 2009. ISSN 1946-7729. 33

Cloudmade. Cloudmade, 2012. URL http://cloudmade.com/. 33

David J. Coleman, Yola Georgiadou, and Jeff Labonte. Volunteered geographic information : The nature and motivation of producers. *International Journal of Spatial Data Infrastructures Research*, 4:332–358, 2009. 24

JBIG Committee. Coding of still pictures. Technical report, Hewlett-Packard Company, 11000 Wolfe Road, MS42U0, Cupertino CA 95014, USA, 1999. URL http://www.jpeg.org/public/fcd14492.pdf. 42

Padraig Corcoran, Adam Winstanley, and Peter Mooney. Segmentation performance evaluation for object-based remotely sensed image analysis. *Int. J. Remote Sens.*, 31(3):617–645, April 2010. ISSN 0143-1161. 115

Padraig Corcoran, Peter Mooney, Michela Bertolotto, and C.Winstanley Adam. View- and Scale-Based Progressive Transmission of Vector Data. In *The 11th International Conference on Computational Science and Its Applications (ICCSA 2011)*, pages 1–13, Santander, Spain, June 2011a. 7, 10, 14, 47, 130, 180

Padraig Corcoran, Peter Mooney, Adam C Winstanley, and Michela Bertolotto. Effective vector data transmission and visualization using html5. In *GIS Research UK*, 2011b. 40, 184

Padraig Corcoran, Peter Mooney, and Michela Bertolotto. Line simplification in the presence of non-planar topological relationships. In *15th AGILE International Conference on Geographic Information Science*, pages 24 –27. Springer, April 2012. 73, 179

Luciano da Fontoura Da Costa and Roberto Marcondes Cesar, Jr. *Shape Analysis and Classification: Theory and Practice*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2000. ISBN 0849334934. 118

Jeremy W. Crampton. Online mapping: Theoretical context and practical applications. In P. Peterson and G. F. Gartner, editors, *W. Cartwright*. Springer-Verlag, 1999. 39

Jeremy W. Crampton. Interactivity types in geographic visualization. *Cartography and Geographic Information Science*, 29(2):85–98, 2002. 32

Bertrand De Longueville, Alessandro Annoni, Sven Schade, Nicole Ostlaender, and Ceri Whitmore. Digital earth's nervous system for crisis events: real-time

sensor web enablement of volunteered geographic information. *International Journal of Digital Earth*, 3(3):242–259, 2010. 2

Alan Dix, Janet E. Finlay, Gregory D. Abowd, and Russell Beale. *Human-Computer Interaction (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2003. ISBN 0130461091. 159

David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, Oct 1973. 45, 58

Thomas P. Duncan and Denis Gracanin. Algorithms and analyses: pre-reckoning algorithm for distributed virtual environments. In Stephen E. Chick, Paul J. Sanchez, David M. Ferrin, and Douglas J. Morrice, editors, *Winter Simulation Conference*, pages 1086–1093. ACM, 2003. ISBN 0-7803-8132-7. 158

Robert I. Dunfey, Bruce M. Gittings, and James K. Batcheller. Towards an open architecture for vector gis. *Comput. Geosci.*, 32(10):1720–1732, December 2006. ISSN 0098-3004. 29, 44

Alistair Edwardes, Dirk Burghardt, Matthias Bobzien, Lars Harrie, Lassi Lehto, Tumasch Reichenbacher, Monika Sester, and Robert Weibel. Generalisation technology: Addressing the need for a common research platform. In *Proceedings 21st International Cartographic Conference*, Durban, South Africa, 2003. 39

Sarah Elwood. Volunteered geographic information : future research directions motivated by critical , participatory , and feminist GIS. *GeoJournal*, 72(3 -4): 173 –183, 2008. 17

Facebook, 2012. URL `http://www.facebook.com/`. Facebook is a social utility that connects people with friends and others who work, study and live around them. 17

Vincenzo Fatto, Luca Paolino, Monica Sebillo, Giuliana Vitiello, and Genoveffa Tortora. Spatial factors affecting user's perception in map simplification: An

empirical analysis. In *Proceedings of the 8th International Symposium on Web and Wireless Geographical Information Systems*, W2GIS '08, pages 152–163, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-89902-0. 46, 102, 114

OpenStreetMap Features. Map features, 2012. URL http://wiki.openstreetmap.org/wiki/Map_Features. 33

Andrew J. Flanagin and Miriam J. Metzger. The role of site features, user attributes, and information verification behaviours and the credibility of web-based information. *New Media & Society*, 9:319–342, 2007. 23

Flickr. Flickr- photo sharing, 2012. URL www.flickr.com/. 21

Leila De Floriani, Paola Magillo, Franco Morando, and Enrico Puppo. Dynamic view-dependent multiresolution on a client-server architecture. *CAD Journal*, 32:805–823, 2000. 149

Theodor Ernst-Christoph Martin Joachim Foerster. *Web-based architecture for on-demand maps : integrating meaningful generalization processing.* PhD thesis, Universiteit Twente, 02 2010. 146

James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes, editors. *Computer graphics : Principles and practice.* Addison-Wesley, 1996. 36

Jean-Michel Follin, Alain Bouju, Frédéric Bertrand, and Patrice Boursier. Multi-resolution extension for transmission of geodata in a mobile context. *Comput. Geosci.*, 31(2):179–188, March 2005a. ISSN 0098-3004. 5, 10, 129

Jean-Michel Follin, Alain Bouju, Frédéric Bertrand, and Arunas Stockus. An increment based model for multi-resolution geodata management in a mobile system. In *Proceedings of the 5th international conference on Web and Wireless Geographical Information Systems*, W2GIS'05, pages 42–53, Berlin, Heidelberg, 2005b. Springer-Verlag. ISBN 3-540-30848-2, 978-3-540-30848-5. 45, 69, 114

FourSquare. Foursquare, 2012. URL https://foursquare.com/. 17

Michael Franklin and Reinhard Wilhelm. Challenges in ubiquitous data management. In *Informatics*, 2000. 182

Thomas A. Funkhouser and Carlo H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 247–254, New York, NY, USA, 1993. ACM. ISBN 0-89791-601-8. 158

GeoTools. Geotools the open source java gis toolkit, 2012. URL http://www.geotools.org/. 85

Michael F. Goodchild. Data models and data quality: Problems and prospects. In Michael GoodChild, Bob Parks, and Leon Steyaert, editors, *Environmental modeling with GIS*, pages 94 –103. Oxford University Press, 1993. 23, 24

Michael F. Goodchild. Citizens as sensors : the world of volunteered geography. *GeoJournal*, 69(4):211–221, 2007. 2, 17, 26

Michael F. Goodchild. Neogeography and the nature of geographic expertise. *Journal of Location Based Services*, 3(2):82–96, 2009. 17, 33

Michael F. Goodchild and Alan Glennon. Crowdsourcing geographic information for disaster response: a research frontier. *International Journal of Digital Earth*, 3(3):231–241, 2010. 2

Inc. Google. Google maps, 2012. URL http://www.maps.google.com. 1

Alexander Grossmann and J. Morlet. Decompostion of hardy functions into square integrable wavelets of constant shape. *Society for Industrial and Applied Mathematics*, 15:723 – 736, 1984. 43

Antonin Guttman. R-trees: a dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, SIGMOD '84, pages 47–57, New York, NY, USA, 1984. ACM. ISBN 0-89791-128-8. 130

Mordechai (Muki) Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7:12–18, 2008. ISSN 1536-1268. 24

Ahmed Abdel Hamid and Yehia Ahmed, Mahmoud andHelmy. Enhanced progressive vector data transmission for mobile geographic information systems (mgis). In Tarek Sobh, editor, *Innovations and Advances in Computer Sciences and Engineering*, pages 61–66. Springer Netherlands, 2010. 129

Mark Hampe and Monika Sester. Real-time integration and generalization of spatial data for mobile applications. Technical report, Institute for cartography and geoinformatics, University of Hannover, Germany, 2001. 40

Qiang Han, Michela Bertolotto, and Joe Weakliam. A conceptual model for supporting multiple representations and topology management. In *Conceptual Modeling for Advanced Application Domains*, 2004. 182

Lars Harrie and Hanna Stigmar. An evaluation of measures for quantifying map information. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(3): 266 – 274, 2010. ISSN 0924-2716. 41, 101

Lars Harrie, L. Tina Sarjakoski, and Lassi Lehto. A variable-scale map for small-display cartography. In *Proceedings of the Joint International Symposium on Geo- Spatial Theory*, 2002. 39

Jan-Henrik Haunert, Arta Dilo, and Peter van Oosterom. Constrained set-up of the tgap structure for progressive vector data transfer. *Computers and Geosciences*, 35(11):2191 – 2203, 2009. ISSN 0098-3004. 5, 45, 127

Christian Heipke. Crowdsourcing geospatial data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6):550 – 557, 2010. ISSN 0924-2716. 4, 26

Hugues Hoppe. Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 99–108, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. 36, 149

Hugues Hoppe. View-dependent refinement of progressive meshes. In *Proceedings of the 24th annual conference on Computer graphics and interactive tech-*

niques, SIGGRAPH '97, pages 189–198, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. 130

Andrew Hudson-Smith, Michael Batty, Andrew Crooks, and Richard Milton. Mapping for the masses. *Soc. Sci. Comput. Rev.*, 27(4):524–538, November 2009. ISSN 0894-4393. 21, 23

Sergio Ilarri, Eduardo Mena, and Arantza Illarramendi. Location-dependent query processing: Where we are and where we are heading. *ACM Comput. Surv.*, 42(3):12:1–12:73, March 2010. ISSN 0360-0300. 15, 32

Amaud E. Jacquin. Image coding based on a fractal theory of iterated contractive image transformations. *Image Processing, IEEE Transactions on*, 1(1):18 –30, jan 1992. ISSN 1057-7149. 42

George F. Jenks. Lines, computers, and human frailties. *Annals of the Association of American Geographers*, 71(1):1–10, 1981. 100, 109

Bin Jiang and Xiaobai Yao. Location-based services and gis in perspective. *Computers, Environment and Urban Systems*, 30(6):712 – 725, 2006. 20

JOSM. extensible editor for openstreetmap, 2012. URL `http://josm.openstreetmap.de/`. 19

Jesper Kjeldskov and Connor Graham. A review of mobile hci research methods. In J Kjeldskov and C Graham, editors, *In Proceedings of Mobile HCI 2003*, pages 317–335, Udine, Italy, 2003. 46, 101

Alexander Kolesnikov. Vector maps compression for progressive transmission. In *Digital Information Management, 2007. ICDIM '07. 2nd International Conference on*, volume 1, pages 81 –86, oct. 2007. 46, 73

Alexander Kolesnikov and Pasi Franti. Data reduction of large vector graphics. *Pattern Recognition*, 38(3):381 – 394, 2005. ISSN 0031-3203. 58

Ourania Kounadi. Assessing the quality of OpenStreetMap data. Master's thesis, University of College London, 2009. 24

Sven Kratz, Ivo Brodien, and Michael Rohs. Semi-automatic zooming for mobile map navigation. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, MobileHCI '10, pages 63–72, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-835-3. 41, 101

John Krumm and Steve Shafer. Data store issues for location-based service. *IEEE Data Engineering Bulletin*, 28:36–43, 2005. 15

Lars Kulik, Matt Duckham, and Max Egenhofer. Ontology-driven map generalization. *Journal of Visual Languages and Computing*, 16:245–267, 2005. 182

Longin Jan Latecki and Rolf Lakamper. Convexity Rule for Shape Decomposition Based on Discrete Contour Evolution. *Computer Vision and Image Understanding*, 73(3):441–454, March 1999. ISSN 10773142. xi, 53, 56, 58, 59, 120

Talia Lavie, Tal Oron-Gilad, and Joachim Meyer. Aesthetics and usability of in-vehicle navigation displays. *International Journal of Human-Computer Studies*, 69(1 C 2):80 – 99, 2011. ISSN 1071-5819. 41

Lassi Lehto and L. Tina Sarjakoski. Real time generalization of XML encoded spatial data for the Web and mobile devices. *International Journal of Geographical Information Science*, 19(8-9):957–973, September 2005. ISSN 1365-8816. 5, 29, 40, 69

Chao Li. User preferences, information transactions and location-based services: A study of urban pedestrian wayfinding. *Computers, Environment and Urban Systems*, 30(6):726 – 740, 2006. ISSN 0198-9715. 41

Qingquan Li. Variable-scale representation of road networks on small mobile devices. *Computers & Geosciences*, 35(11):2185–2190, November 2009. ISSN 00983004. 129

Ruth Rosenholtz Yuanzhen Li and Lisa Nakano. Measuring visual clutter. *Journal of Vision*, 7(2):1 –22, 2007. 104

David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 199–208, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. 37, 130

David Luebke, Benjamin Watson, Jonathan D. Cohen, Martin Reddy, and Amitabh Varshney. *Level of Detail for 3D Graphics*. Elsevier Science Inc., New York, NY, USA, 2002. ISBN 1558608389. 159

Visvalingam M. and Whyatt J.D. Line generalisation by repeated elimination of points. *Cartographic Journal, The*, 30(1):46–51, 1993. 58

Eoin Mac Aoidh, Michela Bertolotto, and David Wilson. Towards dynamic behavior-based profiling for reducing spatial information overload in map browsing activity. *GeoInformatica*, 16(3):409 –434, 07 2012. 38, 183

Mapnik. Mapnik: Free toolkit for developing mapping applications, 2012. URL http://mapnik.org/. 19

Mapquest. Mapquest, 2012. URL http://www.mapquest.com/. 1, 33

Pierre F. Marteau and Gildas Menier. Speeding up simplification of polygonal curves using nested approximations. *Pattern Anal. Appl.*, 12(4):367–375, October 2009. ISSN 1433-7541. 114

Asif Masood. Optimized polygonal approximation by dominant point deletion. *Pattern Recognition*, 41(1):227 – 239, 2008. ISSN 0031-3203. 58

Jacob Matell, Michael S.and Jacoby. Is there an optimal number of alternatives for likert scale items? : Reliability and validity. *Educational and Psychological Measurement*, 31:657–674, 1971. 104

Robert Brainerd McMaster. Automated line generalization. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 24:74 – 111, 1987. 39

Robert Brainerd McMaster and K. Stuart Shea. Generalization in digital cartography. In *Monograph Series*, volume Monograph Series, 1992. 5, 39

Donald Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129 – 147, 1982. ISSN 0146-664X. 130

Nirvana Meratnia and Rolf A. de By. Spatiotemporal compression techniques for moving point objects. In *Proceedings of the 9th International Conference on Extending Database Technology (EDBT 2004)*, Crete, Greece, March 2004. Springer-Verlag. 51

Esmond Mok and Geoffrey Shea. Geolocation positioning with wireless cellular network in hong kong. *The Hong Kong Institute of Surveyors Journal*, 15(Issue 2):23–30, 2004. 35

Peter Mooney and Padraig Corcoran. How social is OpenStreetMap ? In *The 15th AGILE International Conference on Geographic Information Science*, 2012a. 104

Peter Mooney and Padraig Corcoran. Characteristics of heavily edited objects in openstreetmap. *Future Internet*, 4(1):285–305, 2012b. ISSN 1999-5903. 19, 33, 53

Peter Mooney, Padraig Corcoran, and Adam C Winstanley. Geospatial data issues in the provision of Location-based Services. In *Proceedings of the 7th International Symposium on LBS & TeleCartography at South China Normal University (SCNU)*, pages 1–14, 2010. 17, 20, 33, 113

Peter Mooney, Huabo Sun, and Lei Yan. Vgi as a dynamically updating data source in location-based services in urban environments. In *Proceedings of the 2nd international workshop on Ubiquitous crowdsouring*, UbiCrowd '11, pages 13–16, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0927-1. 20

N. Mustafa, S. Krishnan, G. Varadhan, and S. Venkatasubramanian. Dynamic simplification and visualization of large maps. *International Journal of Geographical Information Science*, 20(3):273–302, March 2006. ISSN 1365-8816. 5, 47, 76

Pascal Neis, Dennis Zielstra, and Alexander Zipf. The street network evolution of crowdsourced maps: Openstreetmap in germany 2007. *Future Internet*, 4 (1):1–21, 2011. ISSN 1999-5903. 19

Andreas Neumann. Using svg for online digitizing and editing of geographic data. Online at http://www.svgopen.org/2004/papers/ - last checked November 2012, 2004. 28

Andreas Neumann, Andre M. Winter, and Tirol Atlas. Chapter 12 - webmapping with scalable vector graphics (svg): Delivering the promise of high quality and interactive web maps. In Michael Peterson, editor, *Maps and the Internet*, pages 197 – 220. Elsevier Science, Oxford, 2003. ISBN 978-0-08-044201-3. 5

Moritz Neun and Dirk and Burghardt. Automated processing for map generalization using web services. *GeoInformatica*, 13:425 – 452, 2009. 5

Peter Van Oosterom. *Reactive Data Structures for Geographic Information Systems*. PhD thesis, Department of Computer Science, Leiden University, 1990. 70

Peter Van Oosterom. The GAP-tree , an approach to On-the-Fly Map Generalization of an Area Partitioning. In *GISDATA Specialist Meeting on Generalization*, pages 15 – 19, France, December 1993. 5, 46, 69, 127, 130

OpenStreetMap. URL http://www.openstreetmap.org/. 2

OpenStreetMap. The openstreetmap wiki. Online at http://wiki.openstreetmap.org/wiki/Main_Page - last checked November 2012, 2012. 31

Antti Oulasvirta, Mikael Wahlstrom, and K. Anders Ericsson. What does it mean to be good at using a mobile device? an investigation of three levels of experience and skill. *International Journal of Human-Computer Studies*, 69 (3):155 – 169, 2011. ISSN 1071-5819. 41

Luca Paolino, Monica Sebillo, Genny Tortora, and Giuliana Vitiello. A perception based selection of vector map lods for progressive transmission. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on GeoStreaming*,

IWGS '11, pages 29–32, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-1036-9. 96, 100, 101, 114

Theodosis Pavlidis and Steven L. Horowitz. Segmentation of plane curves. *Computers, IEEE Transactions on*, C-23(8):860 – 870, aug. 1974. ISSN 0018-9340. 58

Arie Pikaz and Its'hak Dinstein. An algorithm for polygonal approximation based on iterative point elimination. *Pattern Recognition Letters*, 16(6):557 – 563, 1995. ISSN 0167-8655. 58

David Pisinger. A minimal algorithm for the multiple-choice knapsack problem. *European Journal of Operational Research*, 83:394–410, 1994. 127

Inc. PKWARE. Application note on the .zip file format. official white paper, Dec. 2012. URL http://www.pkware.com/documents/casestudies/APPNOTE.TXT. 57

Brandon Plewe. *GIS Online: Information Retrieval, Mapping, and the Internet*. OnWord Press, 1st edition, 1997. ISBN 1566901375. 28

P. M. van der Poorten, Sheng Zhou, and Christopher B. Jones. Topologically-consistent map generalisation procedures and multi-scale spatial databases. In *Proceedings of the Second International Conference on Geographic Information Science*, GIScience '02, pages 209–227, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-44253-7. 73

Edward Pultar, Martin Raubal, Thomas J Cova, and Michael F Goodchild. Dynamic gis case studies: Wildfire evacuation and volunteered geographic information. *Transactions in GIS*, 13(s1):85–104, 2009. 2, 34

Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244 – 256, 1972. ISSN 0146-664X. 45

Jonathan Raper, Georg Gartner, Hassan Karimi, and Chris Rizos. Applications of location-based services: a selected review. *J. Locat. Based Serv.*, 1(2):89–111, June 2007. ISSN 1748-9725. 14, 15

RARLAB. Winrar, a powerful tool to process rar and zip files, 2012. URL http://www.rarlab.com/. 57

Tumasch Reichenbacher. *Mobile Cartography of Adaptive Visualisation of Geographic Information on Mobile Devices*. PhD thesis, Technischen Universitat Munchen, 2004. 4, 29, 32, 33, 39

Paul L. Rosin. Techniques for assessing polygonal approximations of curves. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(6):659 –666, jun 1997. ISSN 0162-8828. 58, 62, 120

Paul L. Rosin and Christine L. Mumford. A symmetric convexity measure. *Computer Vision and Image Understanding*, 103(2):101 – 111, 2006. ISSN 1077-3142. 116

Alan Saalfeld. Topologically consistent line simplification with the douglas-peucker algorithm. *Cartography and Geographic Information Science*, 26(1): 7–18, 1999. 46, 76

Marc Salotti. An efficient algorithm for the optimal polygonal approximation of digitized curves. *Pattern Recognition Letters*, 22(2):215 – 221, 2001. ISSN 0167-8655. 59

Hanan Samet. *Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. ISBN 0123694469. 130

Daniel Scherzer, Michael Wimmer, and Werner Purgathofer. A survey of real-time hard shadow mapping methods. *Comput. Graph. Forum*, 30(1):169–186, 2011. 159

Jochen Schiller and Agnes Voisard. *Location-based services*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. 27

Dieter Schmalstieg and Michael Gervautz. Demand-Driven Geometry Transmission for Distributed Virtual Environments. *Computer Graphics Forum*, 15(3): 421–432, August 1996. ISSN 0167-7055. 37, 146

Monika Sester and Claus Brenner. Continuous Generalization for Visualization on Small Mobile Devices. In *In Proc. Conference on Spatial Data Handling*, 2004. 41, 46, 69, 101

P. Greg Sherwood and Kenneth Zeger. Progressive image coding on noisy channels. In *Data Compression Conference, 1997. DCC '97. Proceedings*, pages 72 –81, mar 1997. 42

H.Z. Shu, L.M. Luo, J.D. Zhou, and X.D. Bao. Moment-based methods for polygonal approximation of digitized curves. *Pattern Recognition*, 35(2):421 – 434, 2002. ISSN 0031-3203. 58

Prabhakant Sinha and Andris A. Zoltners. The multiple-choice knapsack problem. *Operations Research*, 27:503 –515, 1979. 127

Richard Southern, Simon Perkins, Barry Steyn, Alan Muller, Patrick Marais, and Edwin Blake. A stateless client for progressive view-dependent transmission. In *Proceedings of the sixth international conference on 3D Web technology*, Web3D '01, pages 43–50, New York, NY, USA, 2001. ACM. ISBN 1-58113-339-1. 37

Stats. Places to find information about openstreetmap's database, 2012. URL http://wiki.openstreetmap.org/wiki/Stats. 23

Erik Steiner, Alan Maceachren, and Diansheng Guo. Developing lightweight, data-driven exploratory geo-visualization tools for the web. In *Advances in Spatial Data Handling: Proc., 10th Int. Symposium on Spatial Data Handling*. Springer-Verlag, 2002. 29

Stefan Steiniger, Moritz Neun, and Alistair Edwardes. Foundations of Location Based Services. Lecture notes on lbs, Department of Geography, University of Zurich, Winterthurerstrasse 190, CH-8057 Zurich (Switzerland), 2006. URL http://ftp.heanet.ie/mirrors/sourceforge/j/project/ju/jump-pilot/Documentation/articles/lbs_lecturenotes_steinigeretal2006.pdf. CartouCHe. x, 1, 15, 16, 181

Daniel Z. Sui. The wikification of GIS and its consequences: Or Angelina Jolie's new tattoo and the future of GIS. *Computers, Environment and Urban Systems*, 32(1):1–5, January 2008. ISSN 01989715. 17

Ali Tahir, Gavin McArdle, and Michela Bertolotto. A web-based visualisation tool for analysing mouse movements to support map personalisation. In Jianliang Xu, Ge Yu, Shuigeng Zhou, and Rainer Unland, editors, *Database Systems for Adanced Applications*, volume 6637 of *Lecture Notes in Computer Science*, pages 132–143. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-20243-8. doi: 10.1007/978-3-642-20244-5_13. URL http://dx.doi.org/10.1007/978-3-642-20244-5_13. 158

Christopher C. Tanner, Christopher J. Migdal, and Michael T. Jones. The clipmap: a virtual mipmap. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 151–158, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. 130

Hock-Hai Teo, Lih-Bin Oh, Chunhui Liu, and Kwok-Kee Wei. An empirical study of the effects of interactivity on web user attitude. *International Journal of Human-Computer Studies*, 58(3):281 – 305, 2003. ISSN 1071-5819. 100

Danny S. P. To, Rynson W. H. Lau, and Mark Green. A method for progressive and selective transmission of multi-resolution models. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '99, pages 88–95, New York, NY, USA, 1999. ACM. ISBN 1-58113-141-0. 149

Bradley Tonder and Janet Wesson. Design and evaluation of an adaptive mobile map-based visualisation system. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I*, INTERACT '09, pages 839–852, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-03654-5. 41, 101

Nectaria Tryfona and Dieter Pfoser. Data semantics in location-based services. In Stefano Spaccapietra and Esteban Zimanyi, editors, *Journal on Data Semantics III*, volume 3534 of *Lecture Notes in Computer Science*, pages 587–587. Springer Berlin / Heidelberg, 2005. ISBN 978-3-540-26225-1. 16, 20

Andrew Turner. *Introduction to neogeography*. O'Reilly, first edition, 2006. ISBN 0596529953. 19

The Java(TM) Web Services Tutorial. Why stax?, 2012. URL http://docs.oracle.com/cd/E17802_01/webservices/webservices/docs/1.6/tutorial/doc/SJSXP2.html. 85

Joviša Žunić and Kaoru Hirota. Measuring shape circularity. In *Proceedings of the 13th Iberoamerican congress on Pattern Recognition: Progress in Pattern Recognition, Image Analysis and Applications*, CIARP '08, pages 94–101, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85919-2. 118

Robert Weibel. A typology of constraints to line simplification. In M.J. Kraak and M. Molenaar, editors, *Advances in GIS Research II (7th International Symposium on Spatial Data Handling)*, 1996. 181

Daniel Weiner, Trevor M Harris, and William J Craig. Introduction: Community participation and geographic information systems. In Daniel Weiner William J. Craig, Trevor M. Harris, editor, *Community Participation and Geographic Information Systems*, page 416. Taylor & Francis, 2002. 17

David C. Wilson, Heather Richter Lipford, Erin Carroll, Pamela Karr, and Nadia Najjar. Charting new ground: modeling user behavior in interactive geovisualization. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, GIS '08, pages 61:1–61:4, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-323-5. 146

Michael Worboys. Imprecision in finite resolution spatial data. *Geoinformatica*, 2(3):257–279, October 1998. ISSN 1384-6175. 23, 24

Yi Xiao, Ju Jia Zou, and Hong Yan. An adaptive split-and-merge method for binary image contour data compression. *Pattern Recognition Letters*, 22(3 C 4):299 – 307, 2001. ISSN 0167-8655. 58

Yahoo!2009. Flickr blog, 2009. URL http://blog.flickr.net/en/2009/10/12/4000000000/. 23

Bisheng Yang. A multi-resolution model of vector map data for rapid transmission over the internet. *Comput. Geosci.*, 31(5):569–578, June 2005. ISSN 0098-3004. 5, 14, 76, 129

Bisheng Yang and Robert Weibel. Editorial: Some thoughts on progressive transmission of spatial datasets in the web environment. *Computers and Geosciences*, 35:11, 2009. 43

Ling Yang, Liqiang Zhang, Jingtao Ma, Jinghan Xie, and Liu Liu. Interactive visualization of multi-resolution urban building models considering spatial cognition. *Int. J. Geogr. Inf. Sci.*, 25(1):5–24, February 2011. ISSN 1365-8816. 102

Dave Yates and Scott Paquette. Emergency knowledge management and social media technologies: A case study of the 2010 haitian earthquake. *International Journal of Information Management*, 31(1):6 – 13, 2011. ISSN 0268-4012. 26

Peng-Yeng Yin. Polygonal approximation using genetic algorithms. In *Proceedings of the 8th International Conference on Computer Analysis of Images and Patterns*, CAIP '99, pages 175–182, London, UK, UK, 1999. Springer-Verlag. ISBN 3-540-66366-5. 58

Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C Winstanley. Using shape complexity to guide simplification of geospatial data for use in Location-based Services. In *Proceedings of the 7th International Symposium on LBS & TeleCartography at South China Normal University (SCNU)*, pages 1–16, 2010a. 7, 49, 54, 86, 116

Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam C. Winstanley. Polygon processing on openstreetmap xml data. In M. Haklay, J. Morley, and H. Rahemtulla, editors, *the GIS Research UK 18th Annual Conference*, pages 149–154. University College London, 2010b. 7, 86

Fangli Ying, Peter Mooney, Padraig Corcoran, and Adam Winstanley. How little is enough? evaluation of user satisfaction with maps generated by a progressive transmission scheme for geospatial data. In *proceedings of the 14th AGILE International Conference on Geographic Information Science*, 2011. 7, 46, 101

Li Yunjin and Zhong Ershun. A new vector data compression approach for webgis. *Geo-Spatial Information Science*, 14:48– 53, 2011. 14, 47

Hao Zhang, A.C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2126 – 2136, 2006. 120

Liqiang Zhang, Liang Zhang, Yingchao Ren, and Zhifeng Guo. Transmission and visualization of large geographical maps. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(1):73 – 80, 2011. ISSN 0924-2716. 45

Zhi Zheng, Prakash Edmond, and Tony Chan. Interactive view-dependent rendering over networks. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):576–589, May 2008. ISSN 1077-2626. 38

Min Zhou and Michela Bertolotto. A Data Structure for Efficient Transmission of Generalised Vector Maps. In *COMPUTATIONAL SCIENCE*, volume 3039/2004, pages 948–955, 2004. 46, 69, 73, 97, 129

Min Zhou, Michela Bertolotto, Ki-Joune Li, and Christelle Vangenot. Efficiently generating multiple representations for web mapping. In *Web and Wireless Geographical Information Systems*, 2005. 97, 179

Xiaofang Zhou, S. Prasher, and M. Kitsuregawa. Database support for spatial generalisation for www and mobile applications. In *Web Information Systems Engineering, 2002. WISE 2002. Proceedings of the Third International Conference on*, pages 239 – 246, dec. 2002. 35