



Proceedings of the 1st European State of the Map

– OpenStreetMap Conference –



FORSCHUNGSGRUPPE
KARTOGRAPHIE
Research Group Cartography

The 1st European State of the Map Conference took place
at Vienna University of Technology on July 15-17, 2011.

Edited by Manuela Schmidt and Georg Gartner

Processed by Manuela Schmidt and Felix Ortag

For copyright questions, please contact the authors.

Table of Contents

OpenStreetMap: Just a Database, or Catalyst for Cartographic Revolution Steve Chilton	3
Usability of Editors: What to Improve? Patrick Weber, Catherine Emma Jones	14
GEMMA – Making Maps Even Easier Oliver O’Brien, Steven Gray, Andrew Hudson-Smith	35
OSMatrix – Grid-based Analysis and Visualization of OpenStreetMap Oliver Roick, Julian Hagenauer, Alexander Zipf	44
OSMonto – An Ontology of OpenStreetMap Tags Mihai Codescu, Gregor Horsinka, Oliver Kutz, Till Mossakowski, Rafaela Rau	55
Projected Hill Map – Between Virtual-Reality and Physical Models Gregory Marler	65
OpenWebGlobe SDK – An Open Source High-Performance Virtual Globe SDK for Open Maps Martin Christen, Stephan Nebiker	76
Mobile Augmented Geo Data Visualization Based on Open Street Map Alexander Erifiu and Gerald Ostermayer	92
Modeling Railways in OSM: From 2D to 1D Berend Schotanus	112
OpenStreetMap For Traffic Simulation Michael Zilske, Andreas Neumann, Kai Nagel	126
Evaluating Urban Traffic Congestion Using OpenStreetMap and Sparse Public Transport Data Andrei Bejan	135
Using Open Street Map (OSM) for Logistics and Planning: A Case Study Ben Abelshausen	158

OpenStreetMap: Just a Database, or Catalyst for Cartographic Revolution

Steve Chilton

Middlesex University

Abstract. Is OSM just a database, which has a map to show off that data? Does the resulting cartography exemplify good digital mapping? Would slippy maps be so prevalent without OSM? Would OpenLayers have the penetration it has? Would mapnik be so widely used? Is OSM actually driving upwards capabilities for mapping? Did editable map styles result from OSM? These questions and developments from Google (crowdsourcing data) and Mapquest (supporting OSM developments) will be explored and addressed in this paper.

Keywords. *OpenStreetMap, database, cartographic revolution, OSM, OpenLayers*

1. Introduction

Cartography can be traced way back in history and has developed hugely as an art and science since those early days. Soon after the World Wide Web became prevalent in the mid 1990s web mapping interfaces appeared. Typically they would show a single map image. This might be reloaded from a web server after the user clicked to zoom in. Panning might be implemented either by re-centering by clicking on the map image, or by moving stepwise in the direction of one of the cardinal compass directions. One of the early implementations of these techniques was from Mapquest.

2. Some History

2.1. OpenStreetMap's beginnings

OpenStreetMap (OSM) was founded by Steve Coast in the summer of 2004. Its aims were (and still are) “to provide free geographic data to anyone who

wants them. The project started because most maps you think of as free actually have legal or technical restrictions on their use, holding back people from using them in creative, productive and unexpected ways.”¹ Initially OSM had a very simple web map interface, and wasn’t to develop cartographically for a couple of years.

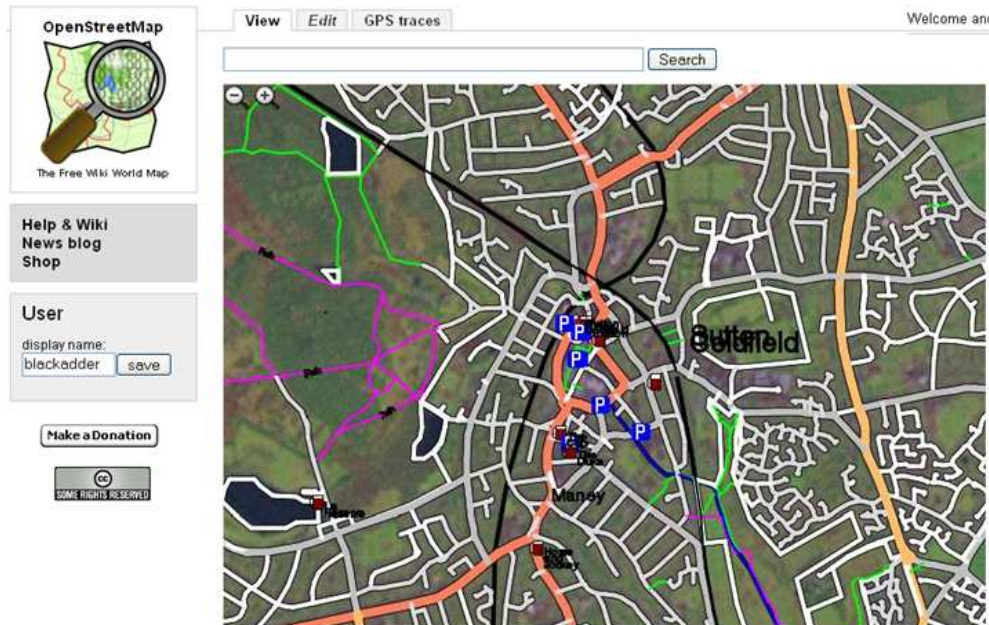


Figure 1. Example of early OSM web map
(Source: <http://svn.openstreetmap.org/misc/lectures/2006/>)

2.2. Slippy Maps

In February 2005 Google Maps was first announced. Continuous panning by dragging was introduced. The map consisted of many small images (tiles) and the web page used Javascript to request new tiles on demand from the web server, without reloading the entire web page. Zooming in or out still required all tiles to be replaced, but the surrounding web page wasn't reloaded. Furthermore, the map covered more than half of the browser window, in contrast to older applications which had a small map window in a large web page. These subsequently acquired the label of slippy maps.

Soon after this Google allowed users to put Google maps on their own websites and mix them with their own data. The latter became known

¹ Aims of OSM directly quoted from: http://wiki.openstreetmap.org/wiki/Main_Page

as mash-ups, a major component of the so-called web 2.0 movement. It was a revolution that over-night made previous web maps look dated.

2.3. OpenLayers

The arrival of Google Maps, and the enthusiasm it generated, prompted Tim O'Reilly to organize a Where 2.0 conference on June 29-30 2005 in San Francisco. Being there inspired MetaCarta to develop an open source equivalent to Google Maps. OpenLayers was first released before the 2nd Where 2.0 conference, which took place on June 13-14, 2006.

It is completely free Open Source JavaScript, released under the FreeBSD licence. It aims to “separate map *tools* from map *data* so that all the tools can operate on all the data sources. This separation breaks the proprietary silos that earlier GIS revolutions have taught civilization to avoid.”²

3. Just a database?

There are those within the project who maintain that OSM is just a geo-database. In fact the aims stated above can be interpreted that way. Others maintain that is only because there is an attractive map display that people then explore that database in order to do creative things with the data. It is significant that the second variant of the project's two main URLs³ goes directly to it's map display. This display, which utilizes OpenLayers, also has several variants of map styles available to display the OSM data. Having said that, it is the multitude of maps being output from the data by project members and by third parties that give rise to some of the claims about OSM's cartographic influence in this paper.

4. The development of OSM's own mapping

4.1. Mapnik

Mapnik⁴ is a toolkit for developing mapping applications. It was developed by Artem Pavlenko shortly after Google maps appeared – roughly concurrently with the development of OpenLayers by Metacarta. Without getting too technical, “mapnik is a library written in the C++ programming

² Aims of OpenLayers directly quoted from: <http://openlayers.org/>

³ <http://www.openstreetmap.org/>

⁴ <http://mapnik.org/>

language. It runs on all major platforms including Linux, Mac OS X, and Windows and it is free – released under a LGPL Licence. It also has Python (another popular high-level language) bindings” (Pavlenko, 2007). The strength of mapnik is the configuration file (in XML format) which is where all styles and data are referenced. These styles can be adapted by the user to influence the map output that is generated.

4.2. OSM moves to mapnik

As has been explained OSM’s map display was initially very poor. The first significant improvement to the look of the web map was the change to using mapnik in November 2006.

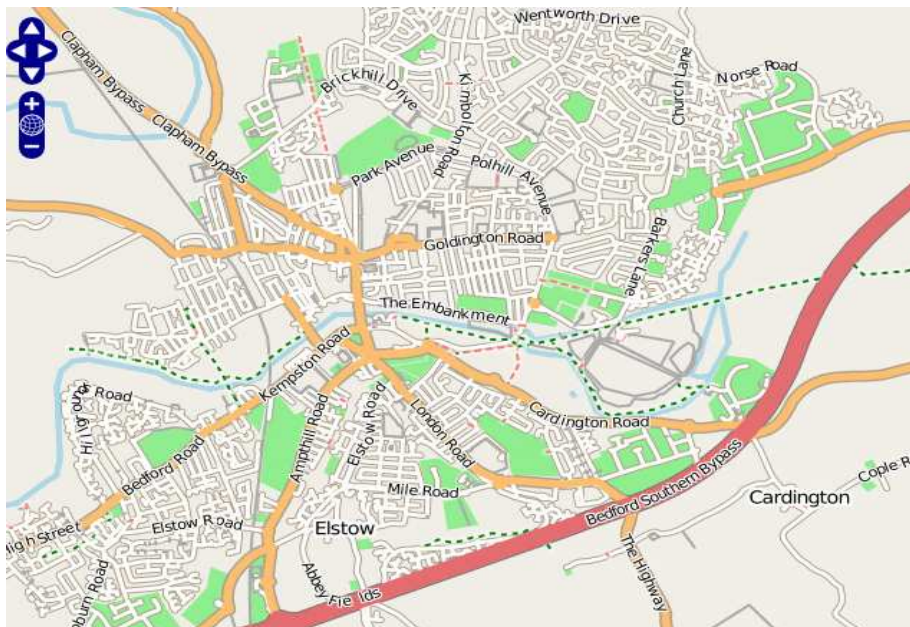


Figure 2. Example of first mapnik rendering

(Source: http://wiki.openstreetmap.org/wiki/Featured_Images/2006)

The initial map styling was designed by Artem Pavlenko, and at this time he was the only person able to make changes to the mapnik stylesheet for the map tile output. The resulting map was a significant improvement, and was so well received that it soon became the accepted default map style for the project.

4.3. OpenStreetMap “cartography day”

It was soon realised that the situation just noted would have to change if things were to move on. Steve Coast facilitated what would now be called a

hackday (but was called a cartography day by him at the time). He brought together himself, Artem Pavlenko, and two prominent cartographers engaged in the project (Richard Fairhurst and the present author). The aim of the get-together was threefold: to refine Artem's initial style with some immediately achievable changes; to produce a list of potential next level changes; and to share the knowledge of working with mapnik to enable others to work on style enhancements in the future.

The meeting took place at Richard Fairhurst's house in Oxfordshire on 7th February 2007 and was very fruitfull. Imagine the scene of a an image of the current map being displayed on the wall via a projector, Steve Coast saying "what can we do to improve this element of the map", the two cartographers discussing and agreeing a potential change in the look of that element, and of Artem Pavlenko making coding changes to the stylesheet accordingly, reloading it and displaying the revised resulting output.

The immediate result was a swathe of changes across many different scales (or zoom levels in OSM parlance) of the map output. These considerably clarified the look and feel of it, but cannot really be demonstrated in one single image. The list of extras was initially taken on board by Pavlenko, who worked on them in the weeks after the meet. Something that was to be very significant was that the author very shortly learnt the basics of mapnik style editing, and took over much of this work, and subsequently became the instigator of many future changes to the map styling.

The quality of the resulting cartography has had good feedback from both within and without the project:

*"..... the default OSM mapnik style is still the finest most widely deployed online cartography."*⁵

*"The quality of internet mapping will have to improve. There are signs that this is already happening - OpenStreetMap, for example."*⁶

⁵ Alex McKee adding a comment to: <http://vimeo.com/5609679>

⁶ Mary Spence, president of the British Cartographic Society: <http://www.cartography.org.uk/default.asp?contentID=864>

5. OpenStreetMap's influence

5.1. Mapnik

As well as being the part of the toolchain for OpenStreetMap, mapnik has penetrated substantially into the web cartography world. Examples of its use can be found in such disparate applications as The People's Map, Everyblock.com, mysociety.org, afganistanelectiondata.org, the London Organising Committee of the Olympic Games, and MapQuest Open – often combined with OpenLayers⁷.

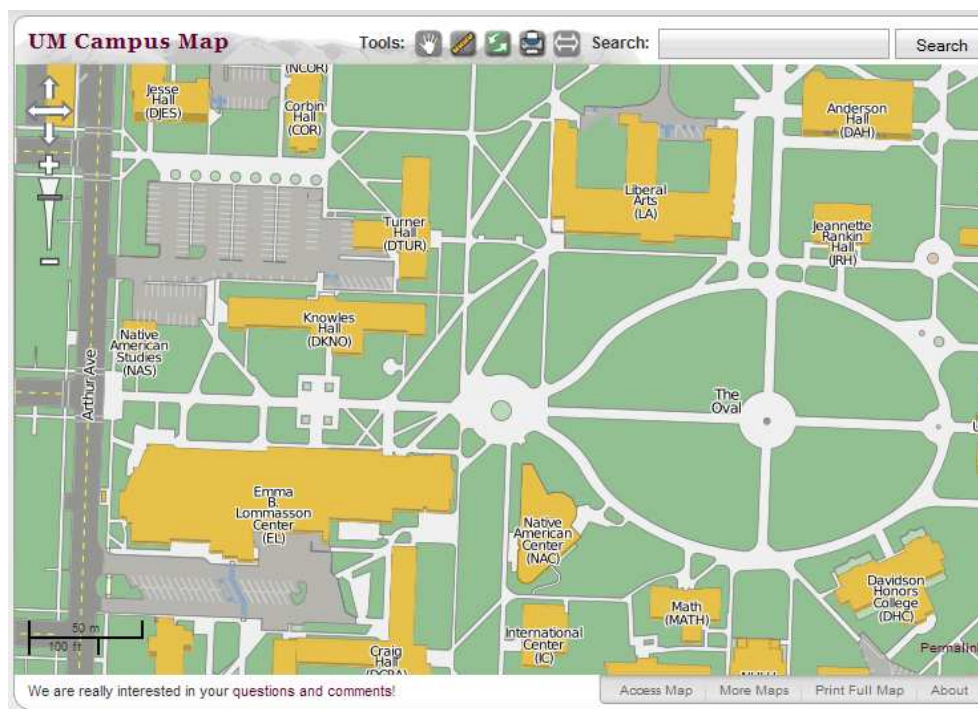


Figure 3. Example of mapnik rendering (Source: <http://map.umt.edu/>).

The website noted in Figure 3 credits its map production to: mapnik, tilecache and OpenLayers. It is not stretching reality to suggest that much of the success of mapnik is down to its usability within Openlayers and more particularly the exposure that the OpenStreetMap project has given it. Michal Migurski (2009), when talking of the work of Stamen Design

⁷ A fuller list of is available on the mapnik wiki at: <http://trac.mapnik.org/wiki/DemoGallery>

(amongst other things the designers of the LOCOG maps mentioned above) declared about mapnik - “it’s the thing that makes OSM look so good, it probably wouldn’t exist without OSM”.

5.2. OpenLayers

OpenLayers is now so prevalent in web cartography that there is no point providing a listing of its uses here. Suffice it to say that when the Ordnance Survey launched its OpenSpace initiative it was inevitably based on OpenLayers, as is their ‘getamap’ service. Was it just coincidence that when it launched the People’s Map had an almost identical toolchain to OpenStreetMap – using both mapnik and OpenLayers? It can be argued that, in a similar way to with mapnik, OpenStreetMap has been instrumental in creating the usage pattern for OpenLayers. In fact this prevalence has been the driver to Cloudmade (founded by Steve Coast and Nick Black, two prominent members of the OpenStreetMap community) deciding to create an alternative platform to OpenLayers. It is called Leaflet and it is “a modern, lightweight BSD-licensed JavaScript library for making tile-based interactive maps for *both desktop and mobile* web browsers”.⁸ So, the OpenStreetMap bandwagon has generated another tool to enrich the modern cartographer’s work.

5.3. Editable styles

The author knows only too well how the simple style file of those heady days of the adoption of mapnik in 2006 has escalated into the multiple inter-linked style files that are now in use to produce the OSM output. For some people this complexity, and the need to understand and work with XML files and mapnik’s dependencies, has stopped them from developing alternative style files and outputs. Cloudmade realised this and developed editable styles for their clients. In 2009 they released their Style Editor. This allows you to change the colour of features such as roads, landuse, etc and turn on/off the object visibility of many other features. Although it appears to be a pretty powerful tool at first glance it does suffer from limitations for serious cartographic use. For instance you cannot change the width of road types, or the size of lettering associated with particular features. Once you have made some changes you can name and save a style and even embed that style in your own webmap (using the Cloudmade Web Maps Lite API). However, you don’t have access to the actual stylefile that will have been created, which remains in the Cloudmade server space somewhere.

⁸ Aims of Leaflet directly quoted from: <http://leaflet.cloudmade.com/index.html>

5.4. Google

Google have been surprising un-innovative in pure cartography terms. In a move that seems to directly follow Cloudmade's Style Editor lead they released in May 2010 their own style editor, which was accessed through their Maps API v3. However, it seems to have very little take-up – as judged purely by the number of times sites with non-standard Google maps appearing on them that have been noted by this author.

Prior to this they had introduced: the satellite view (April 2005), hybrid view (Jul 2005), traffic info (February 2007), My Maps (April 2007), Streetview (May 2007), and Contours in Terrain View (April 2008). However, the one new feature that was a reflection on the increasing influence of OpenStreetMap was the introduction of Map Maker in June 2008. This allowed - initially it must be said in a small experimental sample of island nations only - users to add and edit most features you see on maps including roads, lakes, parks, points of interest, businesses, cities and localities. It was possible to trace many of these features using satellite images. To most people it seemed to be a direct emulation of the crowdsourcing approach of OSM, whereby people with knowledge of changes required by the map (Google map in this case) could make those changes through a web editor. Parsons (2008) blogged on what Map Maker was/wasn't and also attended the OpenStreetMap conference in Limerick shortly afterwards to give a presentation on the same topic.⁹

The fact that the data input through Map Maker becomes Google "property" under the terms of their licence, and the fact that they have over time been increasingly highlighting the "mapping parties" that they host has riled some parts of the OSM community. Maron (2011) published a blog post (over)-reacting to these two facts that caused quite a ripple at the time. Aside from the non-openness of the resulting data, it has been the thesis of this paper that many of the changes noted are a direct follow-through to OpenStreetMap's increasing influence – and this surely is a supreme example of such imitation/flattery.

5.5. Bing

Bing were quite a long time coming to the innovation table. Perhaps their most innovative addition to the field was the introduction of Streetside in February 2009. Whilst this slick stitched photo view was impressive it suffered from being tied in to Silverlight, which may well have hindered its

⁹ Video: Ed Parsons: "What Map Maker is/is not" at SOTMo8 <http://vimeo.com/6751141>

uptake. So, has OSM had any influence on this Microsoft mapping platform, and vice versa? In November 2010 OSM founder Steve Coast blogged that he had accepted a post as Principal Architect at Bing Mobile. In the same blog he announced that “Microsoft is donating access to its global orthorectified aerial imagery to help OpenStreetMappers make the map even better than it already is”. Finally the blog announced that Bing maps had an OpenStreetMap layer (again only viewable with Silverlight in place)¹⁰. Since then it seems to have gone really quiet on the Bing front, unless things have passed under the author’s radar. Then in June 2011 Magicshop Frontdoor was announced. This allows anyone on the web to help improve Bing’s address location results. The site allows you to drag a pin from where Bing thinks an address currently is to the front door of the property. You do this using their aerial imagery. You are shown random pins to not bias results and multiple people have to move the same pin for the update to happen. As each pin is corrected the data is used to improve Bing’s local search results and also it is released to the OpenStreetMap project under the Open Database License¹¹. A strange little app that doesn’t seem to have garnered much bloggage since its launch.

5.6. Mapquest

It has often been claimed that the Mapquest web mapping portal was the most visited web map site in the world. Whether this was the case or not (and the rise of Google Maps would suggest otherwise), it was not doing anything particularly dramatic, until in July 2010 it was announced that they were launching “an experimental site that uses the free open source OpenStreetMap data, as well as various open source tools, software and utilities to bring it all together”¹². The beta website was launched with an excellent routing algorithm. Randy Meech, the Head of Engineering, summed up why they were doing this:

“We believe that open source is ultimately the future for AOL’s local and mapping applications. And we’re very excited about supporting OpenStreetMap, which powers the maps behind Patch, our local news and information platform. We believe community generated maps that are of high quality and accuracy will end up ultimately being the better mapping product for users. Allowing users to improve the areas they know and care

¹⁰ Blog: <http://blog.stevecoast.com/im-working-at-microsoft-and-were-donating-ima>

¹¹ Amended text from: <http://blog.stevecoast.com/magicshop-frontdoor-is-now-live>

¹² <http://devblog.mapquest.com/2010/07/09/mapquest-opens-up-uk/>

about like streets in their neighborhood, in addition to hiking trails, parks and bike paths, we believe will lead to the best mapping experience for all users.”

The initial site was just for the UK, as coverage was so good there already. At the 2010 OpenStreetMap conference they announced a “\$1 million fund to support the growth of open-source mapping in the United States”. Things then started getting serious. Mapquest was using mapnik to generate their map tiles, but it had not got some features that they required (eg SVG Symbolizer, label offsets). So they hired Artem Pavlenko (the mapnik designer) to improve it with the features they needed, and then release them back to the community. The OpenStreetMap search tool (Nominatim) wasn’t as functional as desired so it’s designer Brian Quinion was headhunted to make improvements, and feed them back as well.

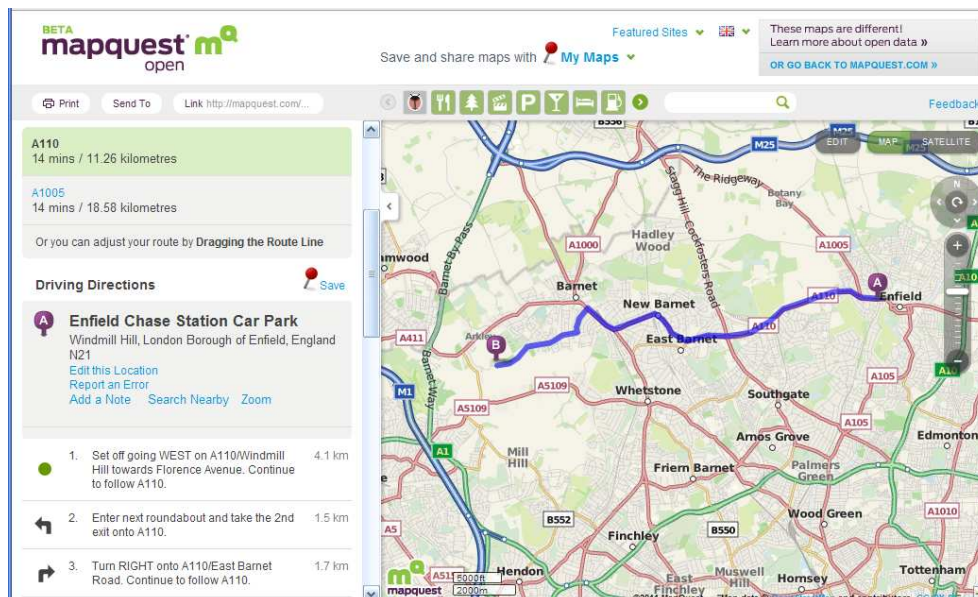


Figure 4. Mapquest open – using OSM data (<http://open.mapquest.co.uk>)

This picking up of skilled participants from the OpenStreetMap development pool (two other ex-Cloudmade staff were taken on) shows graphically how the project has influenced mainstream cartographic developments. Small incremental features began to appear, many of which have been in the OpenStreetMap oeuvre for some while, such as generating HTML code for Embedded Maps. Within months more countries became

covered by the service (with country-specific sites), as Mapquest's confidence in the data quality increased.

Then in October 2010 Mapquest made a donation of a large (unspecified) amount to the OpenStreetMap Foundation for additional hardware to support daily OSM operations. In November two Edit buttons were added to the site to make it even easier for users to contribute data to the project. The links powered up OSM's two main editors, Potlatch and JOSM, to work in the background. In February 2011 they introduced a routing service geared to walking rather than cars, bikes etc as normal. This was presumably because of the richness of the OSM data, with its footways, paths, gates and other detail. But the best part is the Open-ness of the whole initiative. Even the stylefiles are available from the Mapquest Open Initiatives website¹³.

6. Conclusion

The examples above hopefully make the case for OpenStreetMap being much more than just a database, and also for it being catalytic in some serious cartographic developments. If any further evidence is required then one only has to look at some of the maps and products that have come directly, or indirectly, out of the project. Three sources for this are the Best of OSM site¹⁴ and the wiki page for Services Based on OSM¹⁵ and the wiki page for Software (and the huge amount of Mobile Applications)¹⁶ based around OSM and its tools.

References

- Maron (2011) <http://brainoff.com/weblog/2011/04/11/1635> Accessed 8 June 2011
- Migurski M (2009) <http://mike.teczno.com/img/openpapermaps.pdf> Accessed 5 June 2011
- Parsons (2008) <http://www.edparsons.com/2008/06/what-map-maker-is-is-not/> Accessed 8 June 2011
- Pavlenko A (2007) Open Source Renders the World, Society of Cartographers Bulletin 41:13-16

¹³ <http://developer.mapquest.com/web/products/open>

¹⁴ <http://bestofosm.org/>

¹⁵ http://wiki.openstreetmap.org/wiki/List_of_OSM_based_Services

¹⁶ <http://wiki.openstreetmap.org/wiki/Software>

Usability of editors: what to improve?

Dr Patrick Weber*, Dr Catherine Emma Jones**

* Department of Civil, Environmental & Geomatic Engineering,
University College London (UCL), Gower St. London WC1E 6BT
email: p.weber@ucl.ac.uk

** Department of Geography, University of Portsmouth,
Portsmouth PO1 3HE
email: kate.jones@port.ac.uk

Abstract.

This paper presents one of the first systematic investigations into the usability of a Volunteered Geographic Information (VGI) editor front-end, using established best practice in Human Computer Interaction (HCI) research. For this research, OpenStreetMap and its default editor Potlatch 2 were evaluated, through a detailed recording of user interaction patterns with the website and editor front-end. Ten participants with no prior experience of VGI contribution were instructed to enrol and contribute data to OSM, all the while being monitored using a mobile eye tracker and video screen capture software in a computer lab environment. The resulting data was analysed to reveal how users interact and experience VGI editors, as well as highlight problems and deficiencies in the user experience. The results from this research project are a set of recommendations for the future development of these editors, specifically relating to improving the user experience and ease of use of VGI editors.

Web-GIS, Usability Evaluation, Volunteered Geographic Information, Geographic Information Systems, Crowd-Sourcing

1. Introduction

In 1995, Traynor and Williams wrote a paper acknowledging the inherent complexities present within conventional desktop Geographical Information Systems (GIS). It states that the myriad of possible interactions with the map distracts the user from their primary objective, developing spatial knowledge, instead focusing on map manipulation (Jones et al. 2009). A body of work in Human Computer Interaction and desktop GIS

(MacEachren and Kraak 2001; Haklay and Tobón 2003; Worboys, Duckham, and Kulik 2004; Jones et al. 2009) MacEachren and Kraak (2001) state that desktop GIS remain arduous to learn and remember, inefficient and idiosyncratic making them prone to errors, with unhelpful feedback and error types. These usability issues mean that users need to be domain experts, and require expensive training to be comfortable and confident in their daily usage.

Mainstream desktop GIS software is not user friendly, but can the same be said of today's Geoweb applications? Today, there are a number of interactive web tools within the domain of geovisualisation, concerned with the representation, integration, and spatial cognition to facilitate user interaction with spatial data (MacEachren 2004; Slocum 1999; Dykes, MacEachren, and Kraak 2005). These advances in web-based geovisualisation capabilities represent systematic improvements in technical functionality, but significant interface complexity issues remain, affecting usability of these tools and leading to a rather limited uptake by potential users (Slocum 1999; Robinson, Morrison, and Muehrcke 1995; Robinson, et al. 2005; Andrienko and Andrienko, 2006)

Recent technical advances have lead to the rise of web-mapping applications, with an ever-increasing number of lay users adopting online mapping applications (Bing Maps, Google Maps etc). Online mapping API's enable third party web developers to integrate web-mapping functionality in numerous web sites and services, leading to mass-market adoption and consumption of spatial data and services by lay users. Their success means that geoweb services have today moved closer to the attainment of almost universal usability – which is even more astonishing given the exceptionally complex spatial concepts they encapsulate (Shneiderman and Plaisant 2005; Marsh 2007). Their usability significantly stems from the design emphasis centred on user needs, with the result that a user can sit in front of the interface and intuitively start using them without significant prior training.

Web-mapping technology and services, together with the wide availability of GPS receivers, and ubiquitous internet access leading to improved possibilities of mass collaboration, finally enabled the rise of spatial crowd sourcing projects, most notably OpenStreetMap (OSM). OSM follows the peer production model of Wikipedia, with the aim to create a map database, that is free to use, editable and licensed under a liberal copyright scheme encouraging dissemination and usage. OSM today clearly is a successful and growing project, and is on a trajectory to mass-market adoption.

This trajectory of technology adoption can be mapped to a model of technology diffusion and adoption, representing different phases in the lifecycle

of a new technology as it spreads through a given market (Moore 1991). Significantly, in 1991 Geoffrey Moore suggested that a “Chasm” exists in this framework between the Innovators, Early Adopters and the Early Majority (see Figure 1), which represents a step change in adoption rates and signifies a difficult transition from a niche technology towards mass market adoption.

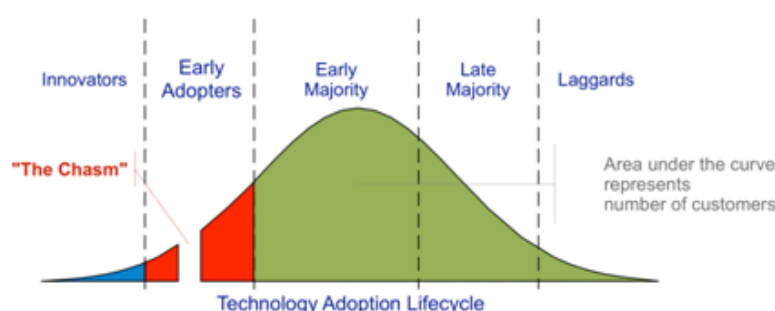


Figure 1: Technology adoption lifecycle¹.

Although OSM is successfully crossing this *Chasm* in terms of adoption of its data, by mass market value added ‘resellers’ such as Mapquest and Microsoft², the *Chasm* represents a much more formidable obstacle when looking at OSM contribution rates.

OSM has reached a core user group of highly motivated enthusiasts who act as *Innovators*. For instance, in June 2011 there were about 14.000 active contributors versus a total registered user-base of about 420.000, i.e. only 4% of registered users were active contributors in June³. User contribution patterns also highlight a further disparity, with just 31 user accounts contributing 50% of the data⁴. However, impressive the growth of OSM has been in terms of active participation in the project, it is fair to say that participation rates remain very low.

¹ Source <http://en.wikipedia.org/wiki/File:Technology-Adoption-Lifecycle.png>.

² Other notable companies are CloudMade, a startup offering OSM based webmapping API's, Nestoria property search engine and Skobbler, a german mobile application developer for road navigation.

³ Source <http://wiki.openstreetmap.org/wiki/Stats>.

⁴ Admittedly, this includes user accounts which import existing bulk datasets, eg US Tiger or the AND company dataset covering the Netherlands.

The challenge remains as to how to widen participation towards mass-market active contributions, and specifically why this has not happened yet at the same pace as the adoption of the product by passive consumers. One more statistic highlighting this participation gap is that of all users registering an OSM account; only 35% go on to make at least one edit to the database⁵! Given that no account is required to consume OSM data or services, the only motivation for registration should be that these users want to contribute data, yet, somehow lose interest or give up before making or saving an edit. Nielsen (2003), a leader in the usability analysis of websites, states that if a website is difficult to use people leave it. Is this the case of the editing process and interface in OSM?

To answer this question, this paper presents the results of a qualitative study investigating the user experience of a group of new users to OSM, by testing the usability of the online editing tool Potlatch 2. The motivation for the experiment is to contribute to the development community of OSM and in some small way contribute to the collective efforts of crossing the “Chasm”.

2. Methodology

There are a number of usability evaluation methods including usability testing, heuristic evaluation, cognitive or pluralistic walkthroughs and predictive models (Nielsen 2004; Haklay and Zafiri 2008; Skarlatidou 2010; Rogers, Preece, and Sharp 2007). A number of techniques have been used to evaluate usability of mapping applications including: (1) observation studies of users (Davies and Medyckyj-Scott 1996; Traynor and Williams 1995) (2) screen shot studies (Haklay and Zafiri 2008); (3) Video recording of end-user input (Robinson et al. 2005; Jones et al. 2009). These approaches represent best practice in usability research, and a convenient development guide for the experiment methodology, which combines participant observation, eye tracking, participant’s *Think Aloud* and screen video capture. A discussion with one of the *Potlatch 2*⁶ developers revealed some of the key tasks for new users. The tasks build up complexity starting with a very simple activity of adding a pub and providing the correct name

⁵ Source <http://neis-one.org/2010/08/%E2%80%99Enominal-members%E2%80%99C-of-osm/>

⁶ Potlatch 2 is the default OpenStreetMap editor which most users encounter when they first click on the 'Edit' tab on the website. Written in Adobe Flash, it enables users to edit the OpenStreetmap inside their browser.

to complex tasks such as ensuring that roads are connected at junctions (for a list of all the tasks see Figure 2). The initial task (not shown in the diagram) requested that the user sign up and activate an OSM account.

The OSM development server environment enabled a consistent test-bed for evaluation, with the data to be edited reset after each participant. There was no time limit given to users, and they could move on from a task even if

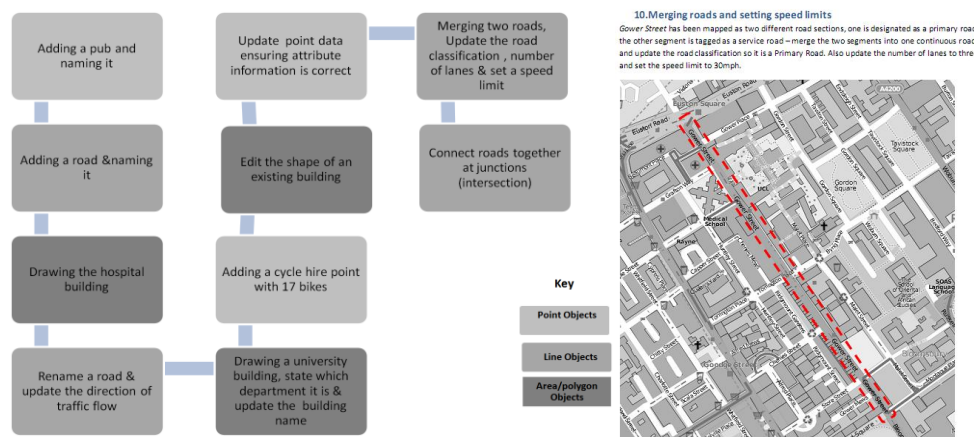


Figure 2: Flow Diagram of tasks, and example step as detailed in instructions. Users were provided with a description of what they had to complete and a map of the local area with the object to be edited highlighted.

not completed. At the start of the experiment each participant completed a general IT questionnaire to indicate the extent of their computer literacy.

Data was collected using a non-intrusive eye tracker recording the participant's gaze, capturing user's interest in interface elements as the eye fixates on them (Poole and Ball 2005). The gaze tracks are recorded on top of a screen recording of the user's interaction with the website (showing mouse movements and clicks etc). In parallel, each participant was asked to *Think Aloud*, (Voudouris and Marsh 2008; Marsh, Dykes, and Attilakou 2006) i.e. users were asked to say whatever they are looking at, thinking, doing, and feeling, as they went about their tasks. After the completion of the experiment, each participant completed a post experiment questionnaire.

The results in this paper represent a first step qualitative analysis based on the observations of ten participants, with data collected through eye tracking and screen video/audio recording, as well as pre and post-test questionnaires.

3. Analysis of Results

Usability can be defined as defines usability as "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use." Usability can be assessed according to numerous measures and concepts, but a commonly accepted framework for usability evaluation relies on five evaluation concepts (Nielsen 2003):

1. Learnability - ability of users to understand the purpose of the interactive tools and learn how to use them; how easy is it for users to accomplish basic tasks,
2. Memorability - ability of users to retain the acquired skills and utilize the tools after a time lapse - how easy is it to re-establish proficiency,
3. Satisfaction - whether people like to use the tools and are not afraid of them,
4. Efficiency – once users have learned a design, how quickly can they perform tasks,
5. Accuracy - how many errors do users make, how severe are the errors and how can they recover from the errors.

Given that this paper presents the first preliminary analysis results, the analysis will primarily focus on the issues of learnability, to identify how new users learn to use the Potlatch Editor and what types of issues hinder the effectiveness of a user learning the system.

3.1. Evaluating learnability

Learnability is one of the main tenants of usability, yet there tends to be discussion around how it should be appraised (Grossman, Fitzmaurice, and Attar 2009). The most common definition of learnability goes back to 1980 and suggests that, "*the system should be easy to learn by the class of users for whom it is intended*" (Michelsen, Dominick, and Urban 1980). Therefore, the essence of learnability equates to the ability of users to understand the purpose of a tool(s) and considers how quickly and efficiently they are able to learn how to use them. It also considers how easy it is for users to accomplish the basic tasks.

For this assessment we shall consider Shneiderman's definition that "it is the time it takes a member of the community to learn how to use the basic commands relevant to a set of tasks" (2002) as a basic measure of learnability, followed by some detailed analysis of specific basic functionalities such as zoom, pan and save, to highlight some common problems and sources of errors.

3.1.1 Task durations and completion rates

Figure 3 shows the average durations for each task across the 10 participants (regardless of if they complete the task successfully – it represents the average time spent on each task). On average it took participants 03:12 minutes to sign up and register for OSM (task 1) but observation of the users revealed that many users tried to log in without realising they needed to wait for the required the activation email.

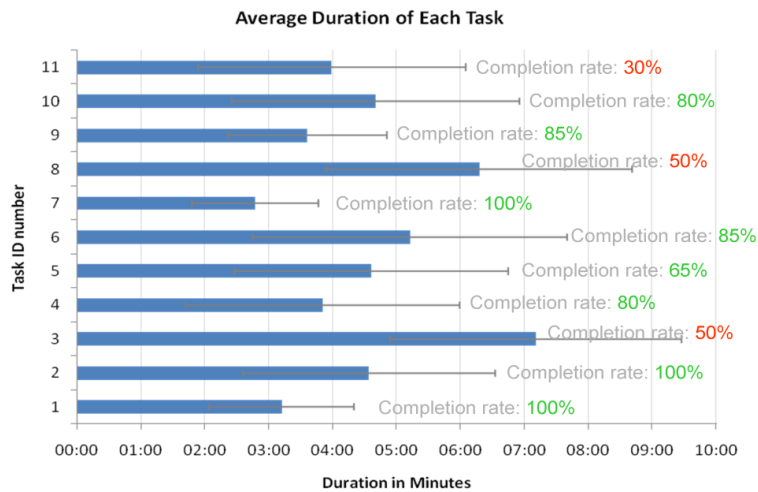


Figure 3: average durations of time spent on each task with confidence intervals and completion rates.

The success rate of a task is defined as the, “percentage of users that completed a task correctly” (Skarlatidou 2010; Nielsen 2003). A score of 1 was given for a completed task, 0.5 for a partially completed task, and 0 for a incorrect task. The total task success rate was 75% with 95% confidence interval of the success rate falling between (0.65 and 0.81)⁷. All participants successfully completed the first task (2) adding a pub and naming it. The average duration time of 4 minutes 34 seconds (with a CI of +-01:13 minutes) for this task can be explained by three factors we observed: (1) needing to set Potlatch 2 as default editor (a setting not activated on the development server); (2) not realising /reading the text at the top right that you drag and drop a pub by first selecting the correct icon from the tag set and (3) the time taken to identify the relevant tag.

⁷ A success rate for the experiment and each task was calculated according to the formula:

$$\frac{\sum s + \sum ps}{N}$$

where s is successful attempts, ps the number of partially successful attempts and N is the number of total attempts.

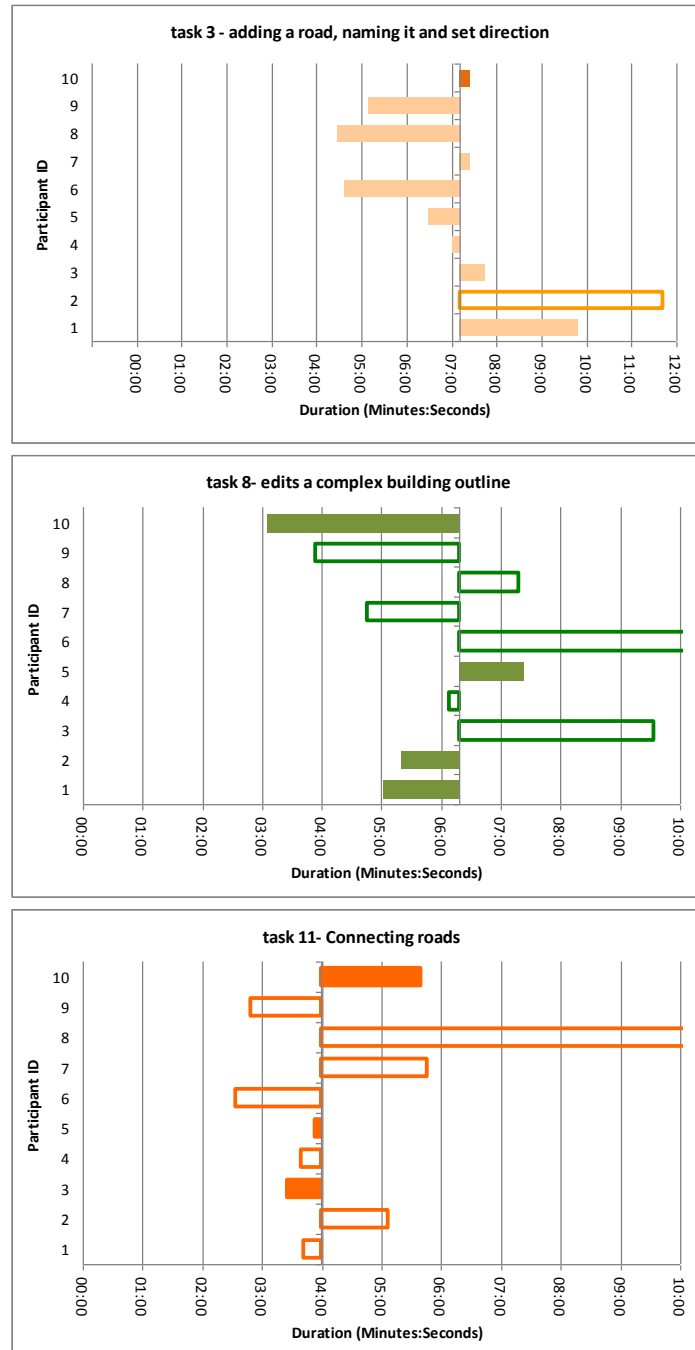


Figure 4: Tasks (t3) adding a road (t8) editing a complex building (t11) connect a road at 2 junctions (Hollow bars = incomplete, Pale bars = partially completed, bold = successful, y axis centred on average duration).

Three tasks appeared particularly problematic with completion rates of 50% or less (see Figure 4).

- a) Task 3: Adding a road, naming it and setting it as one way (a partial completion was awarded if the road failed to connect to other roads)
- b) Task 8: Editing the outline of a complex building foot print.
- c) Task 11: Connecting a road at both ends where there is an over shoot and an undershoot.

Task 3 had a much longer average duration of over 7 minutes than most tasks. The task described above was not completed successfully by participant 2, shown with a hollow duration box in Figure 4 (a) and was only successfully completed by one user. All others failed to connect the road that they drew. Observation of the video revealed that users did not realise you could start creating ways by simply clicking on the map in the edit view:

“...[I] was not aware it was going to start drawing when I clicked on the map ... A bit of trial and error there to get the road to draw” [P06: 10:00], and

“..I can’t quite work out how to make a road.” [P07 09:05].

Only 5 of the 10 participants managed to edit a building outline to match the footprint (task 8, completing the “Cruciform” university building) from the underlying areal photograph. Three users even came back to the building outline at the end of the experiment and tried to complete the task. The frustration of users was evident:

“..let’s go back to the Cruciform, my nemesis!....OK so I give up!” [P06 50:07].

The main issue was that there is no visible user interface element which enables users to add nodes to an existing line or area⁸. The functionality was hidden. Users mostly ended up either dragging other existing nodes of the geometry over to finish the footprint, resulting in an erroneous footprint, or created a separate footprint for the missing building wing⁹. This

⁸ There are keyboard shortcuts provided by Potlatch, but these are only documented in the online help. The specific keyboard shortcut for adding nodes to an existing way were almost universally not discovered by the participants.

⁹ Some participants then found the *Merge* functionality in the edit toolbar in the bottom-right, and believed they could merge the two building polygons. Unfortunately, the merge functionality only works on two lines connected at their end-points (there was no UI feedback to indicate why merging two polygons would not work).

points to issues in regards to the discoverability of some key functionalities in the system, detailed in the next section.

3.2. Locating functionality

Another key learnability factor is the ability of users to locate key functionality within the interface – functionality should not be hidden away from the user. Lazar et al. (2006) found that users lose up to 40% of their time due to frustrating experiences with computers, caused by missing, hard to find and unusable features of the software. For mapping interfaces the map controls are essential - they enable users to interact with the map.

As part of the tasks users invariably had to use the search, pan, zoom the map window and because we asked participants to think aloud as they were completing the activities were able to code up the videos to identify when they were trying to locate this functionality within the interface. We could also use the eye tracking to reveal elements on the page that they were fixating on as they set about looking for this functionality.

3.2.1 Searching for search

For conventional web pages it is essential that the search box is easy and effortless to find (Nielsen 2001). In practice, this means the top of the main body of the page, below any banner area – in order to provide the most likely finding place. Eye tracking analysis of users looking for the search box during Task 2 validates this statement.

Figure 5 (a) highlights the path of the gaze for 6 participants, a set of fixations ordered by time. The search location was in the last place a user looked for it. Four of the participants fixated on the zoom, then scanned across the top menu looking from left to right, before scanning up and down the left side, they were fixating on each interface element including the logo (the looking glass in the OSM logo is a common symbol for search!), help, wiki project and conference links and banners and only then did they locate the search bar. Since posting a video of the gaze tracks on our research blog OSM have moved the search box up on the left side to sit just below the main logo.

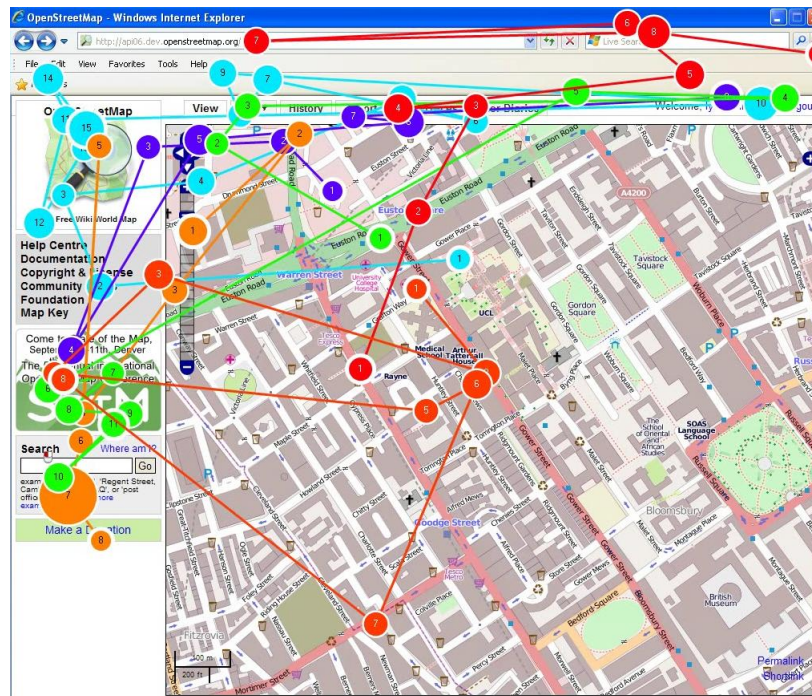


Figure 5: Gaze Plots of 6 participants looking for the search functionality.

3.2.2 Looking for the Zoom

An essential for all mapping applications be they desktop GIS or web-based API's are the common map elements that facilitate exploration of the digital map, chiefly pan and zoom. According to Mark Harrower there are no web standards for the placement, design and functionality of these types of map elements (Harrower and Sheesley 2005), but usability principles suggest there should be consistency in their placement.

There were two types of behaviours among the participants looking to zoom: (1) participants with GIS experience instinctively used the mouse scrolling wheel. (2) Participants without a GIS background looked for specific UI elements to activate zoom in-out, relying on previous experience of the View mode of OSM and its user interface elements.

The gaze heat plots (see Figure 6) of two participants searching the interface for the zoom functionality highlights the most fixated areas where users expected the zoom functionality.



Figure 6: Gaze heat map of 2 participants looking for the zoom functionality.

Consistency is a critical aspect of user friendly design and suggests that related elements should be presented in a similar way. Users had trouble locating the zoom bar in the Edit window because: (a) the edit mode zoom element is completely inconsistent with the view window zoom element, with participants expecting similar placement in the top left of the map window (as evidenced by the top-left hot spot in the gaze heat map); (b) and the discreet + - symbols (which again are not consistent with expected interface elements from the view window) are drowned out by the *noise* of a cartographically busy map, at least this is the case in densely mapped urban areas such as Central London. A comment by one participant summarises this inconsistency quite clearly:

“..not sure if I am missing the obvious Zoom and Pan functions...Aaaghhh there they are, small and hidden” [P10, 31:40].

Here the participant is blaming *themselves* for the trouble they are experiencing locating the zoom functionality in the edit window. The first time this participant searched for the zoom bar they failed to find it and the user switched back to the view window (cancelling all the edits made) located themselves and then switched back to the edit window. This is neither efficient nor effective. It would be relatively simple to either improve the visibility of the zoom on the right side (same look and functionality as the one on the view window) or move it to the top left corner.

3.2.3 Using the pan & selecting objects

Most web-mapping interfaces and desktop GIS traditionally implement map panning functionality by clicking on a location on the map and dragging the map. This is the same in Potlatch, where panning is initiated by clicking and dragging the mouse on the map. Unfortunately, if the first mouse dragging action happens to start above an existing object, instead the object will be selected and moved by the user. When trying to pan, we observed users unintentionally selecting objects and dragging them across the map as they were panning – some users noticed - others did not. Given the density of mapped features in Central London, the test area, this selection versus pan issue became very apparent, as expressed by participants when trying to pan to a new area:

“there is a lot of clutter there....” [P03 11:44].

“Even though I know this area, this visualisation is making it difficult for me to find things“ [P06 30:28]

“oh my gosh it’s hard to know where to click so you don’t impact what is there already!” [P08 37:50].

In Figure 7, the user clicks on the map – does not realise this action has selected an object - exclaims *“Woow, woow, woow!”* Then selects undo and then says *“did not mean to move that”* [P03: 12:06]. Clearly the user did not understand why the object moved, as in a matter of seconds they repeat the exact same error. The issue of panning and moving objects is succinctly summarised by participant 6,

“oops I just moved the anatomy building, did not mean to do that!” [P06: 31:42]. This participant then stated

“...Soon there will be so many objects in the map I will not find an area that I can use to pan” [P06: conversation post experiment].

This issue clearly is of concern specifically in densely mapped urban areas, as opposed to less densely mapped rural areas, with novice users inadvertently introducing errors to existing objects. It could be easily rectified by adding a clearly visible pan map control enabling users to pan without the danger of selection.

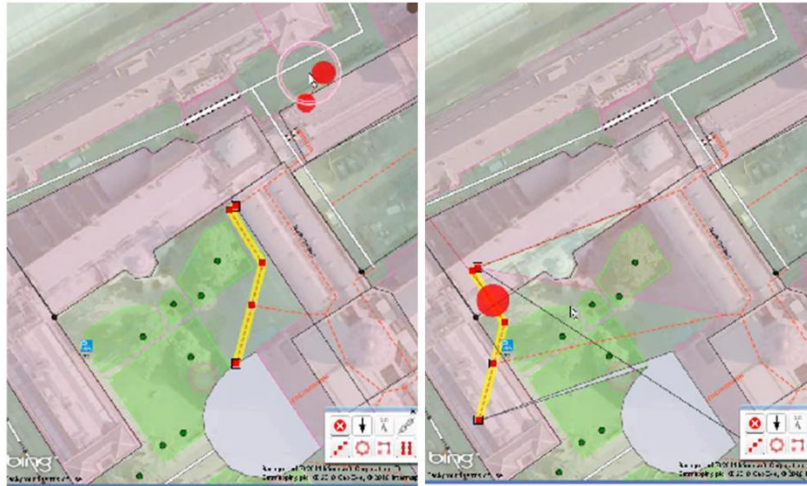


Figure 7: Example screenshots of participant inadvertently moving an existing object (in yellow) while trying to pan.

3.2.4 Save functionality

Given the aforementioned statistic that only about 35% of all registered users go on to make at least one upload to OSM, we were also interested in the usability of the save functionality. Potlatch does not automatically save edits, and users were prompted half-way through the experiment to save their edits, eg. upload their changes to the main OSM database. We observed three issues our participants encountered with this process.

A. Location of the save button

Participants had difficulty finding the save button. This could be due to conventions setup for example by Microsoft Office that save functionality can normally be found at the top-left corner of the UI, whereas in Potlatch the save button is located in the top-right corner. This presumption was also apparent in the eye tracking heat map in Figure 8. One user's comment when looking for save is:

"...Am I supposed to click somewhere to save this. Ummm it should have a save right....! Aghh top right ...weeeee!" [P03: began searching at 10:51 found it at 11:03 seconds]

and *"...[I was] not expecting it to be in top right-hand corner... I suppose"* [P06 27:14]

Participants fixated on the top left corner or the bottom right to locate the save button and generally scanned in an anti-clockwise direction fixating on the obvious most visible elements on the map/page.

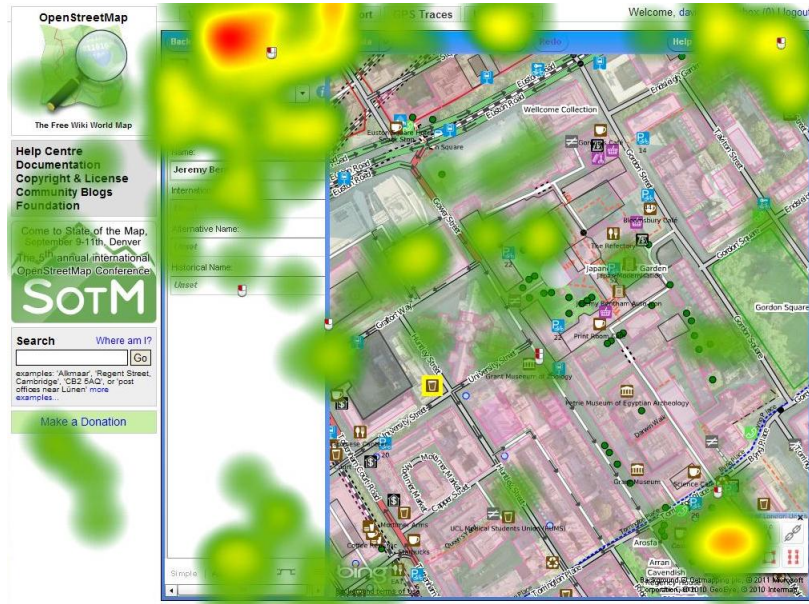


Figure 8: Looking for save button gaze heat map of 4 participants.

B. Exiting Edit mode without saving

There was a knock-on effect for users who had difficulty zooming and panning. As an alternative for moving/zooming the map, they resorted to switch back to the *View* mode to navigate to the next task location, and then switch back to the *Edit* window. If they had unsaved changes, a modal message box would appear before exiting the Edit mode, giving the participant two options: *OK* to continue to the View mode, or *Cancel* to remain in the editor. The participants generally clicked *OK* without reading the message text or stated that, “*they had not made that many edits*” [P03 21:50] so they selected *OK* regardless and thus lost all of their edits.

In usability engineering the *OK/Cancel* order represents the natural reading order, while *Cancel/OK* improves the information flow as the action ends with its conclusion. From this perspective the order of the buttons is not consequential but an improvement would be to set the default to the outcome where no edits are lost, ie. *Cancel* and highlight the button. Users rely upon defaults as guidance and a default would guide a novice user to the expected response.

C. Users did not save frequently

One user did not save any edits at all and only 3 users saved edits 11 times (at the end of every task, see Figure 9). On average, participants saved edits 6 times. This could of course be an artefact of the experiment, as the

changes were not saved to the real OSM database. It does point however to the fact that the save functionality is important for users to not loose contributions, and that it is not always a given that users remember to save edits.

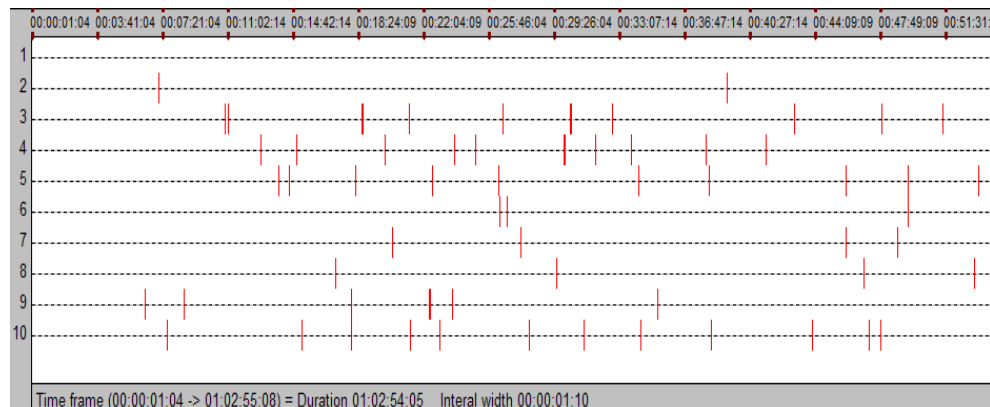


Figure 9: Timeline of when users clicked Save.

3.3. Post-test questionnaire

Apart from the screen, gaze, and audio recordings, a post-test questionnaire also presented the opportunity to collect more data on user satisfaction. A preliminary analysis of this questionnaire revealed that participants found it difficult to use the Potlatch editor, but at the same time, were overall satisfied with the editing experience. For a number of reasons this outcome cannot be taken too literally. The number of participants was relatively small ($n=10$) and there was no clear consensus from the responses for both satisfaction and ease of learning. However, there was genuine consensus between participants that the system was not easy to use. A more indepth analysis will be conducted in the future using conjointly a transcript of *Think Aloud* comments recorded during the whole experiment, and the post-experiment questionnaire. This will help us identify common themes across the individual participants that will guide use towards a more in-depth experiential analysis of their encounter with OSM.

4. Discussion

Given the scope of the study, as well as the amount of data that was collected from the 10 participants, the issues presented before are only the first tentative results from the data analysis. However, these first results already provide some clear insights into initial user behaviours, expectations and interaction patterns and which issues negatively affect learnability in the Potlatch 2 editor. One framework to structure the learnability evaluation is Nielsen's usability heuristics, which enable a classification of the aforementioned issues into different classes of problems. This classification then marks a useful synthesis of learnability issues and some tentative remarks towards their improvement. Please note that only the most relevant categories of this framework are discussed:

(A) *Visibility of system status*: The system should always keep users informed about what is going on, through appropriate feedback within reasonable time. We observed the need for more system feedback, particularly when transitioning between different edits, and selecting objects. Making users better aware of the selection status might prevent or lessen the amount of accidental edits to unintentionally selected objects. Furthermore, notification of the initial loading data phase of the editor is only made known to a user through a small notification in the top left corner, leaving the user to wonder why the map is so empty. Improving the visibility of this notification should guide users. Similarly, many users did not seem to be aware that their edits were not saved – better feedback of the unsaved status of edits may help improve the visibility of system status and reduce lost edits.

(B) *Match between system and the real world & Consistency and Standards*: The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Clear, concise language or concepts should also be consistently expressed through user interface elements. We observed some areas where such guidelines could improve and clarify functionality to novice users. For example, a more consistent placement and grouping of interaction elements as user's preconceptions would dictate could be applied to search, zoom, pan and save functionality discussed beforehand. Users also stated that the term *Way* meant nothing to them and they did not understand what is meant by *Designation*.

(C) *User control & freedom, and Error Prevention*: Users often choose system functions by mistake and will need a clearly marked *emergency exit* which enables them to leave the unwanted state without having to go through an extended dialogue. Alternatively, a careful design acknowledging, minimising and mitigating errors, also contributes to reduce error-

prone conditions. In our experiments, one basic re-occurring problem many users had was that there was no clear exit path to un-select an object. As users moved on and forgot about selection status, subsequent data entry attempts could corrupt previously edited objects, a significant issue where errors are introduced to existing objects. The editing interface should consider a more obvious path for how users un-select a road or building without deleting it or pressing undo. A clear and obvious improvement would be the provision of a select/deselect button within the editing tool bar, which would in turn minimise impacts of editing errors.

(D) Recognition rather than recall, flexibility and efficiency of use & keyboard accelerators: Minimize the user's memory load by making objects, actions, and options visible. For example, we observed a lengthy delay in the time it took a user to start drawing a road, as opposed to adding a Point of Interest. Adding a POI starts explicitly through selection of a POI category and drag & drop of the icon, versus drawing a line, which is implicit in a click on an empty area of the map. Although a quick, flexible way to draw new lines, the lack of an explicit button to draw a line confused new users. Also affecting learnability was the fact that some actions are only accessible through keyboard shortcuts, for example the functionality to create new nodes in an existing line (see building footprint task). Although keyboard shortcuts are very effective accelerators for experienced users, their functionality needs to be accessible through user interface elements such as icons to be discoverable by new users.

(E) Help documentation should be easy to find and useful to the user: Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large. During the experiment users had to be prompted to explore the help documentation but after reading it users generally were able to complete the task. Limitations of the existing help could be improved by larger font sizes, a clearer order of topics from beginner to advanced and editorial restraint to make the instructions as succinct as possible. More attention could be drawn for example also to some excellent introduction tutorial videos for first time users of Potlatch.

5. Conclusion & Future Work

To draw the paper to a conclusion the barriers to learnability observed during the experiments included the limitations of hidden functionality, lack of user feedback between interactions/ tasks and the inefficient location of existing functionality and map controls. Some of the interface issues are

relatively straightforward to improve, for example the location of user interface elements. One example is the location of the *Search* bar, which since the first publication of these results on a blog post, has been moved by OSM developers.

Other issues touched upon here require more effort, and whilst this research provides detailed analysis and evidence to support and highlight problems, quick and easy solutions to these issues are not always possible. Some usability issues are connected to, and intrinsically linked for example to the OSM data-model: the freedom for users to create and use any key-value tags for example brings significant UI challenges to guide new users tagging choices; the lack of a polygon object class in the OSM data model presents not only conceptual data modelling ambiguities, but also interface and cartography challenges as evidenced here.

From the perspective of a practical contribution to the OSM project, this paper presents the beginning of a dialogue with the Potlatch developers, with the ambition to develop an ongoing process of *A/B* interface evaluations. This very much lies in the spirit of continued usability evaluation and scientific rigor to explore the impact and effectiveness changes may have on new user's experiences with OSM.

In terms of scientific research development, this work represents the first type of usability experiment of VGI and its editing tools. We hope to contribute further to this developing body of research, first through the further development of research results from the existing datasets presented here, ie. including further usability components apart from learnability; secondly, to compare results obtained for Potlatch with other OSM editors, as well as other more traditional GIS type interfaces; and thirdly the expansion of this research to other VGI projects, and interaction mediums.

6. References

- Andrienko, N., and G. Andrienko. 2006. Exploratory analysis of spatial and temporal data: a systematic approach. Birkhäuser.
- Davies, C., and D Medyckyj-Scott. 1996. "GIS users observed." *Int. J. Geographical Information Systems* 10 (4): 363–384.
- Dykes, Jason, Alan M. MacEachren, and M. J. Kraak. 2005. Exploring geovisualization. Elsevier.
- Grossman, Tovi, George Fitzmaurice, and Ramtin Attar. 2009. A survey of software learnability: metrics, methodologies and guidelines. In *Proceedings of the 27th international*

- conference on Human factors in computing systems, 649–658. CHI '09. Boston, MA, USA: ACM. doi:10.1145/1518701.1518803.
- Haklay, M., and C. Tobón. 2003. "Usability evaluation and PPGIS: towards a user-centred design approach." *International Journal of Geographical Information Science* 17 (6): 577–592.
- Haklay, M., and A. Zafiri. 2008. "Usability engineering for GIS: learning from a screenshot." *The Cartographic Journal* 45 (2): 87–97.
- Harrower, Mark, and Benjamin Sheesley. 2005. "Designing Better Map Interfaces: A Framework for Panning and Zooming." *Transactions in GIS* 9 (2) (March 1): 77–89. doi:10.1111/j.1467-9671.2005.00207.x.
- Jones, C. E, M. Haklay, S. Griffiths, and L. Vaughan. 2009. "A less-is-more approach to geovisualization—enhancing knowledge construction across multidisciplinary teams." *International Journal of Geographical Information Science* 23 (8): 1077–1093.
- Lazar, Jonathan, Adam Jones, Mary Hackley, and Ben Shneiderman. 2006. "Severity and impact of computer user frustration: A comparison of student and workplace users." *Interacting with Computers* 18 (2) (March): 187–207. doi:16/j.intcom.2005.06.001.
- MacEachren, A. M. 2004. *How maps work: representation, visualization, and design*. The Guilford Press.
- MacEachren, A. M, and M. J Kraak. 2001. "Research challenges in geovisualization." *Cartography and Geographic Information Science* 28 (1): 3–12.
- Marsh, Stephanie Larissa. 2007. "Using and Evaluating HCI Techniques in Geovisualization: Applying Standard and Adapted Methods in Research and Education." <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.112.6786>.
- Marsh, Stephanie Larissa, Jason Dykes, and Fenia Attilakou. 2006. Evaluating a Geovisualization Prototype with Two Approaches: Remote Instructional vs. Face-to-Face Exploratory. In *Information Visualisation, International Conference on*, 0:310–315. Los Alamitos, CA, USA: IEEE Computer Society. doi:http://doi.ieeecomputersociety.org/10.1109/IV.2006.51.
- Michelsen, Christie D, Wayne D Dominick, and Joseph E Urban. 1980. A methodology for the objective evaluation of the user/system interfaces of the MADAM system using software engineering principles. In *Proceedings of the 18th annual Southeast regional conference*, 103–109. ACM-SE 18. Tallahassee, Florida: ACM. doi:10.1145/503838.503847.
- Moore, G. A. 1991. "Crossing the chasm." New York.
- Nielsen, Jacob. 2003. Usability 101: Introduction to Usability. useit. <http://www.useit.com/alertbox/20030825.html>.

- Nielsen, Jakob. 2001. 113 Design Guidelines for Homepage Usability. Useit. <http://www.useit.com/homepageusability/guidelines.html>.
- Poole, Alex, and Linden J Ball. 2005. Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future. In *Encyclopedia of Human-Computer Interaction*, ed. C Ghaoui. Vol. 780. Pennsylvania: Idea Group. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.95.5691>.
- Robinson, A. H, J. L Morrison, and P. C Muehrcke. 1995. *Elements of Cartography*. John Wiley & Sons Inc.
- Robinson, Anthony C., Jin Chen, Eugene J. Lengerich, Hans G. Meyer, and Alan M. MacEachren. 2005. "Combining Usability Techniques to Design Geovisualization Tools for Epidemiology." *Cartography and geographic information science* 32 (4) (October 1): 243-255. doi:10.1559/152304005775194700.
- Rogers, Y., J. Preece, and Sharp. 2007. *Interaction design : beyond human-computer interaction*. Chichester: Wiley.
- Shneiderman, Ben. 2002. "Promoting universal usability with multi-layer interface design." *SIGCAPH Comput. Phys. Handicap.* (73-74): 1-8. doi:10.1145/960201.957206.
- Shneiderman, S. B, and C. Plaisant. 2005. *Designing the user interface 4 th edition*. Pearson Addison Wesley, USA.
- Skarlatidou, Artemis. 2010. Web Mapping Applications and HCI Considerations for their Design. In *Interacting with Geospatial Technologies Interacting with Geospatial Technologies*, ed. Muki Haklay, 245-264. Chichester: Wiley. <http://onlinelibrary.wiley.com/doi/10.1002/9780470689813.ch12/summary>.
- Slocum, T. A. 1999. *Thematic cartography and visualization*. Prentice Hall Upper Saddle River, NJ.
- Traynor, Carol, and Marian G Williams. 1995. Why are geographic information systems hard to use? In *Conference companion on Human factors in computing systems*, 288-289. CHI '95. Denver, Colorado, United States: ACM. doi:10.1145/223355.223678.
- Voudouris, Vlasios, and Stephanie Marsh. 2008. Geovisualisation and GIS: A Human Centered Approach. In *Visual languages for interactive computing: definitions and formalizations*, ed. Fernando Ferri. London: Idea Group Inc (IGI).
- Worboys, M., M. Duckham, and L. Kulik. 2004. "Commonsense notions of proximity and direction in environmental space." *Spatial Cognition & Computation* 4 (4): 285-312.

GEMMA – Making Maps Even Easier

Oliver O'Brien, Steven Gray, Andrew Hudson-Smith

Centre for Advanced Spatial Analysis, University College London,
90 Tottenham Court Road, London W1T 4TJ, United Kingdom

Abstract. GEMMA – a Geospatial Engine for Mass Mapping Applications - is a JISC (UK) funded project which aims to open up the power of custom maps to everyone, by making it as simple as possible to create a map. One of the key workflows will be harnessing free and open data sources to create simple maps. We present an application of GEMMA – the OpenStreetMap Feature Highlighter, which takes data from OpenStreetMap and allows the user to select certain features and styles, position the map on an area of interest, and produce a PDF for a report or an image for online use. The website is built with a carefully selected subset of possible features, to make the experience as straightforward as possible, as it is aimed primarily at people who are unfamiliar with creating and using maps.

Keywords. OpenStreetMap, features, reports, mobile

1 Introduction

The GEMMA project has been designed to tie together a number of diverse map-based web applications created by various authors at CASA. These include *MapTube*, an online archive of user-generated maps of spatially-referenced data; *SurveyMapper*, a survey website that regenerates a map showing the spatial distribution of votes as they happen; *CensusProfiler*, which displays results of the 2001 UK Census as choropleth maps generated dynamically from the data tables, categorizations and colour schemes chosen by the user; and *OpenOrienteeringMap* (OOM), which creates high quality vector-based PDFs of simple orienteering maps, directly from OpenStreetMap data, by applying custom stylesheets to a local, near-live updating copy of the dataset.

As well as its use as the primary data source for OOM, OpenStreetMap (OSM) is used to provide a transparent “contextual” overlay, showing town and city names above the choropleth map in *CensusProfiler*.

As the OSM dataset of spatially referenced data continues to increase rapidly (OpenStreetMap Wiki 2011), the value of the dataset for highlighting geographical features increases. As part of GEMMA, it is realised that utilising the technology developed for *CensusProfiler*, with OSM as the data

source for feature highlighting, as well as the background map, could potentially be useful for report writing and illustration. Combining this with the PDF creation mechanism in OOM would allow attractive PDFs to be easily created.

Ordnance Survey's *Vector Map District* product, part of its *OpenData* release, has the potential to provide an alternative source dataset to OSM, for Great Britain, so its inclusion will be considered. It may be particularly useful for areas where OSM coverage is still significantly incomplete, as evidenced by Haklay (2011). Other data sources used for GEMMA will include user-supplied data in the form of Google Fusion Tables, government and city "data stores" such as the London Data Store, and related institutional spatial datasets, such as EDINA's *GoGeo*.

A mobile application will also allow for user collection of simple (point-based) data, from observations and survey responses, which can then be mapped with GEMMA as an additional data source along with OpenStreetMap.

2 Marketplace Review

A number of web projects and services offer functionality that is similar to GEMMA in some aspects. Most services are fuller featured than GEMMA and have considerably greater functionality.

2.1 GeoCommons

The GeoCommons platform, built by GeoIQ, is an extremely powerful yet user-friendly web application for visualising complex spatial datasets on maps. It occupies the middle-ground between simple data display on online maps, and professional spatial analysis with GIS software.

2.2 Ushahidi

Ushahidi started as a project to allow spatial information to be quickly crowdsourced from disaster areas. It has developed into a more general map crowdsourcing framework. It allows collection of data from a wide variety of platforms, including via SMS messages.

2.3 EpiCollect

EpiCollect is a platform for collecting point-based data, and visualising it. It has both iPhone and Android applications, for collecting and viewing the data, as well as a website.

2.4 Google MyMaps

Google MyMaps allows users to place and annotate markers on a Google Maps background, and save them for viewing by others.

2.5 StaticMap

StaticMap is a simple engine for creating a static image of OpenStreetMap data, superimposing custom user-specified features on top. The entire image can be constructed using URL parameters or through the Static Map website. The resulting image can be recreated by reusing the URL.

2.6 StaticMapLite

StaticMapLite is unrelated to StaticMap, although has the same functionality – creating a static image from OpenStreetMap data, complete with custom markers. It comes with a simple and attractive “wizard” for building up the URL string.

2.7 ITO Map

The ITO Map product is probably the most similar to GEMMA’s OpenStreetMap Feature Highlighter – it overlays and highlights certain OSM features on a minimal background map. The features available are carefully curated by ITO and updated daily. The benefit of ITO themselves choosing the features, ensures correct use of the necessary OSM tags, at a cost of flexibility. The intended audience is mainly OSM contributors looking to improve the quality of OpenStreetMap. Current examples include highlighting roads with/without speed limit indicators, and navigability tagging for waterways.

3 Design

3.1 Key Principals

One of the overriding principles of the GEMMA project has been to make it as easy as possible to use by anyone. It is straightforward to create an application that you yourself, or your peers, can use. Creating an application that is useable by all is a challenge in itself but some important design principles have been taken to ensure ease of use. This includes deliberately limiting the functionality and range of functions, aiming for a simple, streamlined approach.

The main interfaces are a web portal and mobile application. These have a common grid based layout of icons, which trigger the different workflows within GEMMA. GEMMA should therefore be thought as a collection of mini applications with a framework build around it to bring together separate workflows. The individual icons are common to both interfaces and and trigger the same functionary. The framework is highly extensible in design allowing more applications/workflows to be added to GEMMA in the future to improve the overall system.

Authentication is managed through the Facebook, Twitter and Google APIs, increasing security and meaning that users do not need to register separate-

ly with GEMMA. Using Google authentication also unlocks a potential further data source – Google Docs, Spreadsheets and Fusion Tables.

3.2 Diverse Data Sources and Outputs

GEMMA is a consolidation project – it is designed to pull together various projects and present a unified, simplified interface to them. One such data source is SurveyMapper, which records and presents maps of survey results. Both the imagery, and the underlying data, would be accessible. An example can be seen in Figure 1.

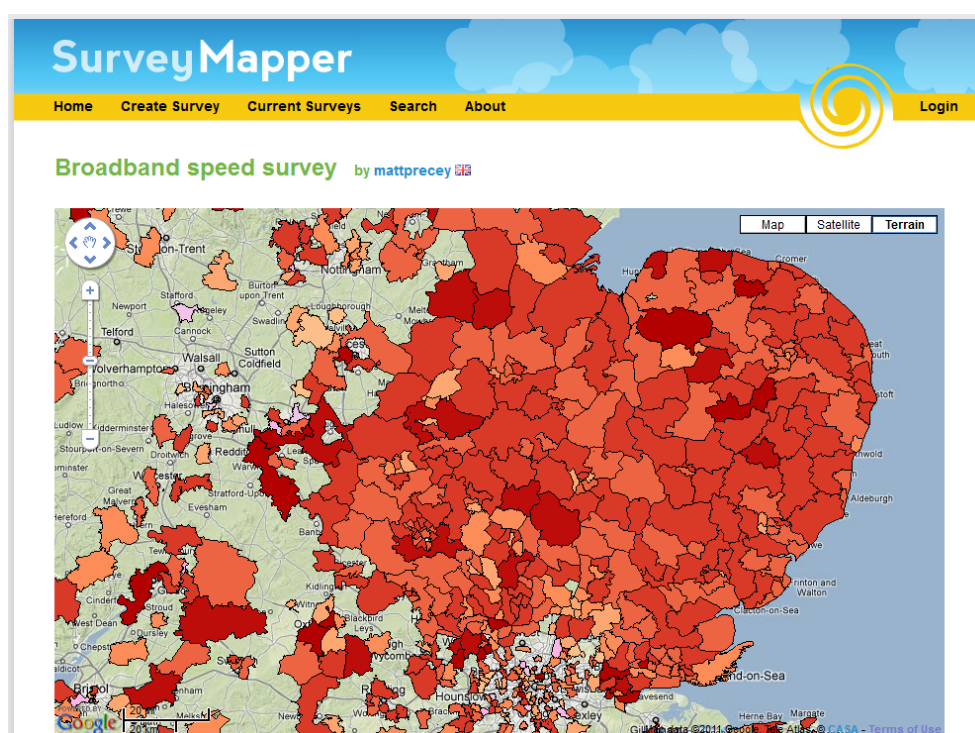


Figure 1. Example SurveyMapper map, showing results from a Broadband speed survey in East Anglia, UK. (SurveyMapper website. Contains Google Map imagery which is copyright Google and TeleAtlas.)

Another key datasource is MapTube. As with SurveyMapper the pre-made choropleth maps, and the underlying source data, would be available. Inexperienced users may prefer using the pre-made map images, while more experienced users might want to “remake” them with, for example, different colour schemes, for GEMMA. An example map in MapTube is given in Figure 2.

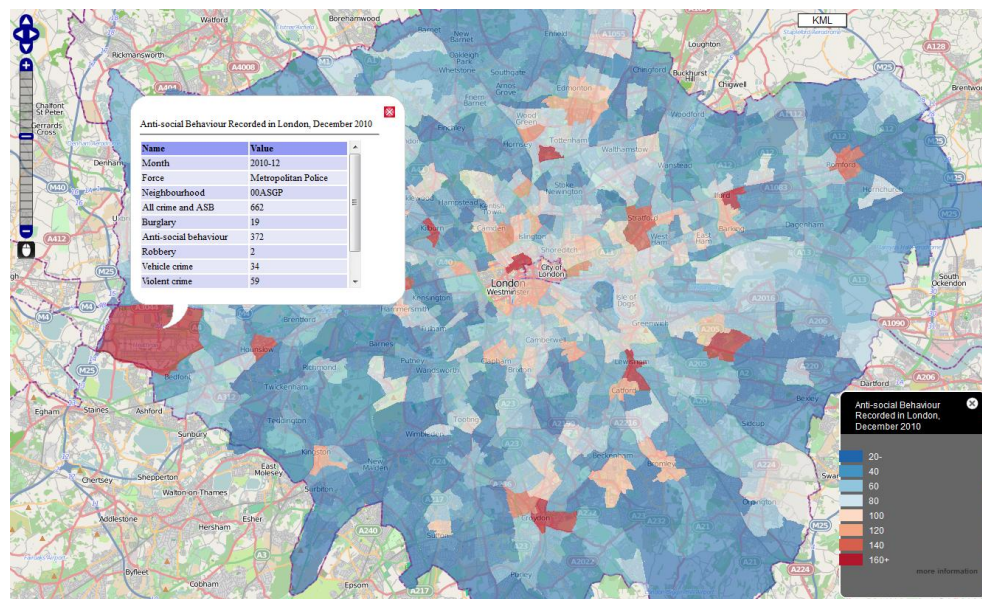


Figure 2. Example MapTube map, showing police figures for anti-social behavior hot-spots in London in December 2010. Heathrow is selected. OpenStreetMap’s default map is used as a backdrop to provide context to the choropleth map layer. (MapTube website. Contains map data which is CC-BY-SA OpenStreet-Map and contributors 2001, and crime data which is licenced under the Open Government Licence.)

It is envisaged that the “image tiles” from SurveyMapper and/or MapTube, both of which are generated by the *MapTubeD* service, would be pulled into a GEMMA map, along with information from other sources, potentially including OpenStreetMap, which would be generated by Mapnik.

GEMMA would take these diverse sources and produce the final map, as a PDF of layered tiles and vector artwork, or as a PNG image. It would make use of Mapnik’s ability to produce PDFs of both the (vector) OpenStreet-Map data, and composited PNG images from SurveyMapper, MapTube and other data sources. This is summarised in the figure below.

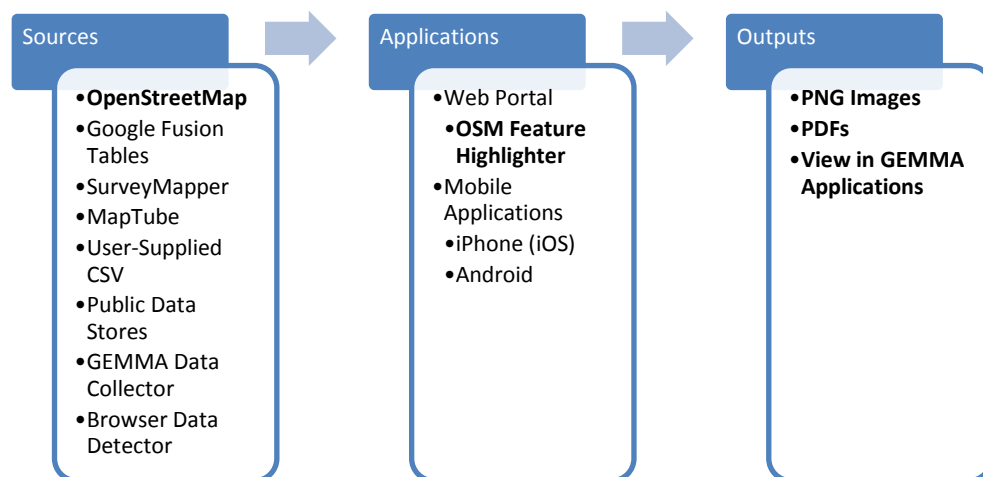


Figure 3. Sources, Applications and Outputs of GEMMA. The OpenStreetMap functionality in GEMMA is bolded.

4 Prototype Applications

4.1 Web Portal

An early prototype of the main web portal for GEMMA. This acts as a single interface to all of GEMMA's functionality, including the GEMMA OpenStreetMap Feature Highlighter that is explored in this paper.



Figure 4. A prototype of the GEMMA web portal. (GEMMA website.)

4.2 Mobile Portal and Data Collector

An iPhone and Android-platform application has been designed, as a mobile portal to view and collect data through the GEMMA system. One of the modules in the application is Data Collector, which allows easy collection of point-based data, for later mapping with GEMMA. Google Fusion Tables are used to store the data.



Figure 5. A prototype of the GEMMA mobile portal (left) and data collector (right). (GEMMA mobile client.)

4.3 OpenStreetMap Feature Highlighter

This is the key OpenStreetMap-related functionality of GEMMA. It presents a simple user interface to selecting specified features from a copy of the OpenStreetMap database, held for the project. It then superimposes such features on the top of a general OpenStreetMap map – currently a custom rendering is being used, which has minimal colour to avoid distraction with the highlighted features.

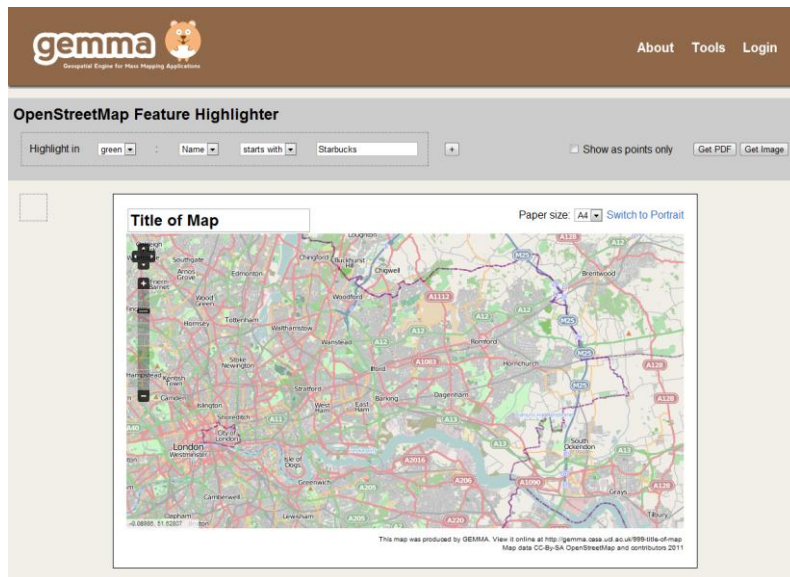


Figure 6. An early prototype of the OpenStreetMap Feature Highlighter in GEMMA. (GEMMA website. Contains map data which is CC-BY-SA OpenStreetMap and contributors 2001.)

5 Example Outputs

The following figures show outputs from the OpenStreetMap Feature Highlighter functionality of GEMMA.

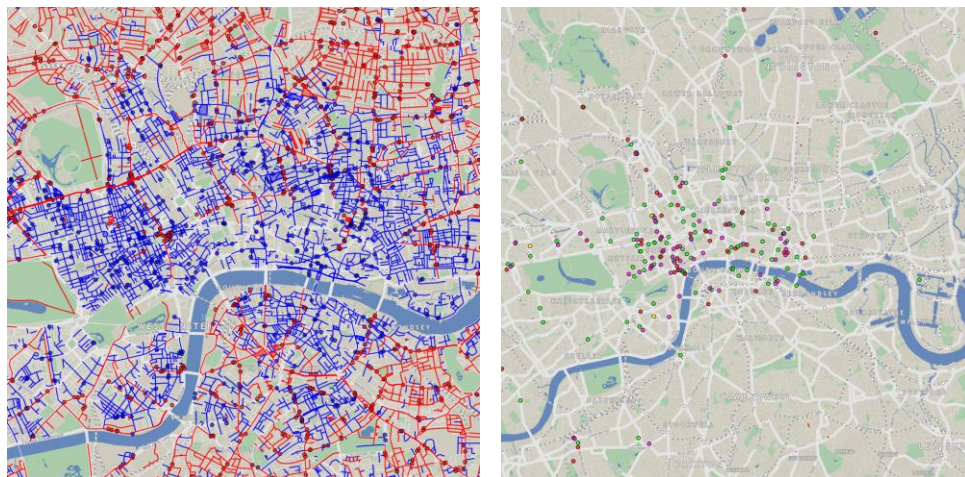


Figure 7. Two example outputs of the GEMMA OpenStreetMap Feature Highlighter. Left: showing the occurrences of “Road” versus “Street” as a name suffix, in central London. Right: Point map highlighting the locations of coffee shops of the four major chains in central and inner-city London. (GEMMA website. Contains map data which is CC-BY-SA OpenStreetMap and contributors 2001.)



Figure 8. Extract showing the high-resolution of the PDF output from GEMMA OpenStreetMap Feature Highlighter, due to the use of vector graphics. (GEMMA website. Contains map data which is CC-BY-SA OpenStreetMap and contributors 2001.)

6 Conclusion

GEMMA is a useful project to link together some of CASA's earlier work. The OpenStreetMap Feature Highlighter will allow increased use of specific parts of the OpenStreetMap dataset, particularly in reports, where high-quality output is required, keeping the spatial data in the OpenStreetMap database itself but storing the specific report metadata in the GEMMA framework. As such, it is hoped that the OpenStreetMap Feature Highlighter will be a useful addition to the OpenStreetMap application ecosphere. More advanced uses of GEMMA – combining OSM with the other data sources available – have the potential to further create new and unique maps.

References

- Haklay, M (2011) OpenStreetMap and Ordnance Survey Meridian 2 Comparison 2008 – 2011. <http://povesham.wordpress.com/2011/05/20/openstreetmap-and-ordnance-survey-meridian-2-comparison-2008-2011/> Accessed 20 June 2011
- OpenStreetMap Wiki (2011) Stats. <http://wiki.openstreetmap.org/wiki/Stats> Accessed 20 June 2011

OSMatrix – Grid-based Analysis and Visualization of OpenStreetMap

Oliver Roick, Julian Hagenauer, Alexander Zipf

Chair of GIScience, Institute of Geography, Heidelberg University, Berliner Str. 48, 69120 Heidelberg

Abstract. Since data in OpenStreetMap is mostly surveyed by untrained private citizens rigorous quality control is required in order to produce a reliable data set. “Hidden” information, such as user activities, topicality of data, which is typically not visualized in maps might indicate areas with increased need for quality assessments. OSMatrix is a visual analytics approach that provides such information. The paper presents a short introduction to the OSMatrix application and some preliminary findings drawn from the resulting maps.

Keywords. Visual analysis, data quality, Volunteered Geographic Information

1. Introduction

Web 2.0 (O'Reilly 2005) technologies enable users to contribute to web-based data-stores, accessible through the World Wide Web. This change of perspective on how the World Wide Web is used resulted in a variety of projects, which are based on participatory collaboration (e.g. social networks such as Facebook or content communities like Flickr and Youtube). This development has also changed the way geographic information is acquired, managed, and distributed through the World Wide Web. Michael Goodchild coined the term Volunteered Geographic Information to subsume this development and to describe „the widespread engagement of large numbers of private citizens, often with little in the way of formal qualifications, in the creation of geographic information, a function that for centuries has been reserved to official agencies“ (Goodchild 2007).

The most prominent example for the phenomenon of Volunteered Geographic Information is OpenStreetMap (OSM), a project with the aim to

create a free world map solely through voluntary effort. OSM was founded in 2004 by Steve Coast and has since become a serious free alternative to data sets provided by commercial vendors and also gained the attention of the scientific community. There is a wide range of applications, which demonstrate the applicability of OSM as a data basis, e.g. the routing service OpenRouteService¹ (Neis & Zipf 2008), the virtual globe OSM3D (Neubauer et al. 2009) or disaster logistics applications (Neis et al. 2010). Furthermore, several studies on the quality of OSM data (Hakley 2010, Neis et al. 2010, Helbich et al. 2010, Ludwig et al. 2011) have shown that – at least in urban areas – the data quality is mostly comparable to commercial data sets.

However, since the data of OSM is mainly captured by untrained amateurs, it is helpful to learn about user behavior, frequency of updates, the topicality of the data, or the number of objects mapped for a certain feature type and to identify areas that stand out from their surroundings and thus may be subject for further investigations. This information provides interesting insights on the spatial diversity of OSM data, e.g. different activities and mapping habits of users in different countries or quality issues, and therefore is a valuable source to improve the overall data quality of OSM.

Because of the large size and complex structures of geospatial data sets, visual analytics have been proven as helpful methods to provide an overview and easily reveal patterns or anomalies within the data set (DiBiase 1990, Dykes et al. 2005). In the context of Volunteered Geographic Information visual exploration has already been applied successfully to provide insights on the given data set: Trame & Keßler (2011) created heat maps to identify objects in the OSM data set that are subject to a large number of edits. MacEachren et al. (2011) introduced a web-based geovisual analysis approach to allow for sense making of Twitter tweets in order to support crisis management agencies.

OSMatrix² is an attempt to supply the above-mentioned information using a visual analytics approach. Essentially, OSMatrix is a web-based application which portrays the spatial distribution of several key attributes, that where derived from the OSM dataset.

The paper is structured as follows: First an overview of data procession and the application itself is given. Second, some findings in OSM applying the OSMatrix and preliminary results are presented. The paper concludes with

¹ <http://openrouteservice.org>

² <http://osmatrix.uni-hd.de>

some final remarks and presents an overview of planed activities in order to improve the application.

2. Application Design and Implementation

In order to illustrate the spatial distribution of OSM data for Europe the area is divided into hexagonal cells using the open source library GeoTools. For each cell a given number of attributes is then calculated. The results are tagged with a timestamp and stored into a PostGIS database for further analysis and visualization.

The calculated attributes include information on user behavior (number of objects modified per user, number of contributing users), the topicality of data (version numbers and timestamps of the mapped objects), the relative completeness (number of features and number of attributes) or simply aggregated numbers of several feature types (area covered by buildings, farmland, commercial, residential and industrial areas) in OSM.

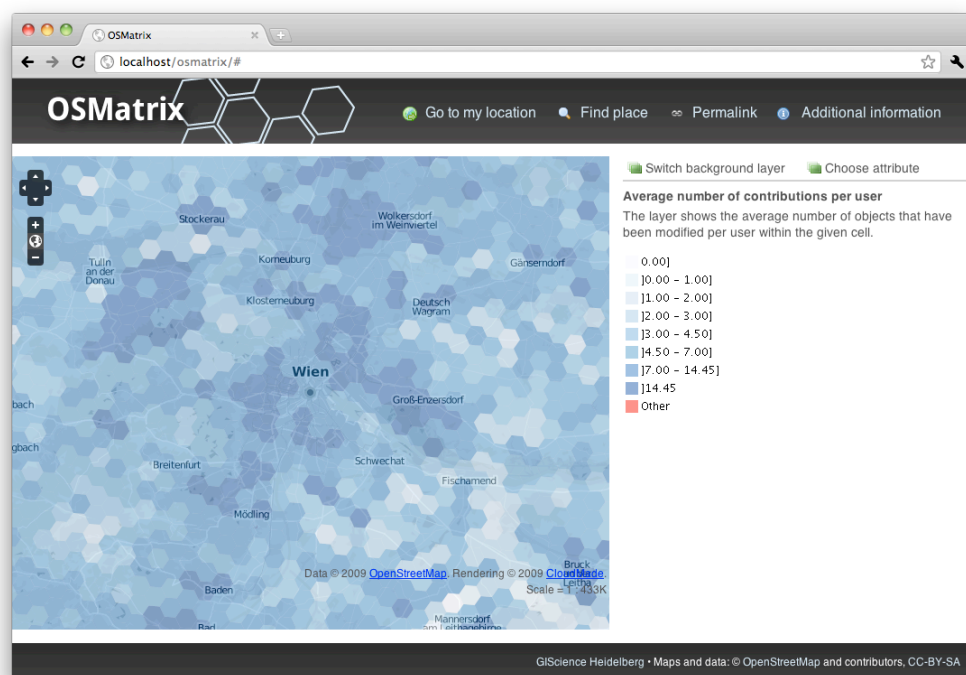


Figure 1: Screenshot of the web-based OSMatrix client.

In order to provide a geovisual analysis tool, a web-based application has been implemented. Therefore, the aforementioned PostGIS database is lin-

linked to a Geoserver based map service, which provides access to maps of the attributes through an OGC compliant Web Map Service interface. The client application is based on OpenLayers and allows for the map-based portrayal of the calculated attributes (Figure 1). Furthermore, additional information on the values for a given cell can be queried using the GetFeatureInfo tool.

3. Results

In the following section some preliminary insights that were derived from the visualization of the attributes mentioned in the previous section will be presented. The focus of this study concentrates on aspects of user activity and overall number of features and attributes. Single feature types have not been considered yet, except for buildings, because these are subject to increased mapping activities since Bing areal image were conferred right to use for mapping.

3.1. Germany

For Germany, the images of the number of features and the sum of all attributes within a cell are similar. For both, in the southern part the numbers are very high except in mountainous areas, whereas in the northern part those high numbers are restricted to urban areas. In rural parts the sum of attributes is mostly below 10 and the number of features is below 2.

Compared to other countries, the average version number is larger for Germany. Again, there are huge differences between urban and rural areas with greater values in urban areas. Same goes for the average number of contributions per user as well as the number of contributing users per cell: In urban areas the number of contributions is mostly above 7, in rural areas mostly below 3. The number of contributions is above 10 in cities and below 2 in rural areas.

In conclusion there is a very active community in Germany, which is reflected by the great number of features and attributes and the number of contributions and contributing users. Nevertheless, there are differences between urban and rural areas, with more active users in urban areas.

As the area covered by buildings implies, houses are mapped quite well in Germany's urban areas. However, the images are still somewhat patchy, indicating that buildings are still not mapped completely. In the countryside the numbers are mostly 0, so there are no buildings mapped at all.

3.2. The Netherlands

The Netherlands seem to be mapped almost completely. The sum of features and the number of attributes show the largest values compared to all other countries in Europe and the numbers are constant across the whole country.

The average version number is between 1 and 2 throughout the whole country, except for some smaller areas in the west, which is very low compared to other countries, which happen to have an active user community.

The average number of objects modified per user is constantly above 14.75 in the whole country and is thus amongst the highest in Europe.

Across the whole country the number of contributing users per cell is larger than in other European countries, but comparable to the numbers in Germany. But still, the numbers in rural areas are larger than in Germany.

Also, the area covered by buildings evolves constantly across the country. As expected, urban areas show larger numbers, but also rural areas show values between 30.98 and 76.29. Therefore, we conclude, that buildings are mapped almost completely throughout the country.

The large numbers of attributes and features throughout the whole country along with the low version numbers and large numbers of contributions per user may indicate that there has been a data import for the Netherlands. The large area covered by buildings reflects that as well.

3.3. France

The number of contributing users shows the expected situation for France: Numbers are significantly larger in urban areas than in rural areas. Hot-spots are Paris, Marseille, Montpellier, Toulouse, Bordeaux and Lyon.

Contrary to the number of contributing users, the number of features mapped per cell and the sum of attributes behaves differently. One can find hot spots around the larger cities as well, but there are also areas with high numbers identifiable, where –based on the number of contributing users– lower values were expected. These areas include parts in western France around the cities of Tours, Poitiers, Nantes, Rennes and Brest as well as Perpignan and close to the Spanish border near Saint-Jean-de-Luz.

The aforementioned regions also show a larger area covered by buildings within a given cell. That's interesting since these areas do not match urban areas in France, where we expect a higher coverage of buildings. Furthermore, the boundaries of the patches match the boundaries of several administrative boundaries, which may indicate that the data originates from data imports.

The average version number also does not shape up as expected. In general the number is lower compared to Germany. But there are some patches, which stand out: Large average version numbers can be found around Amiens in north, between Paris and Reims (Figure 2), around Dijon in central France as well as around Bordeaux in the west.

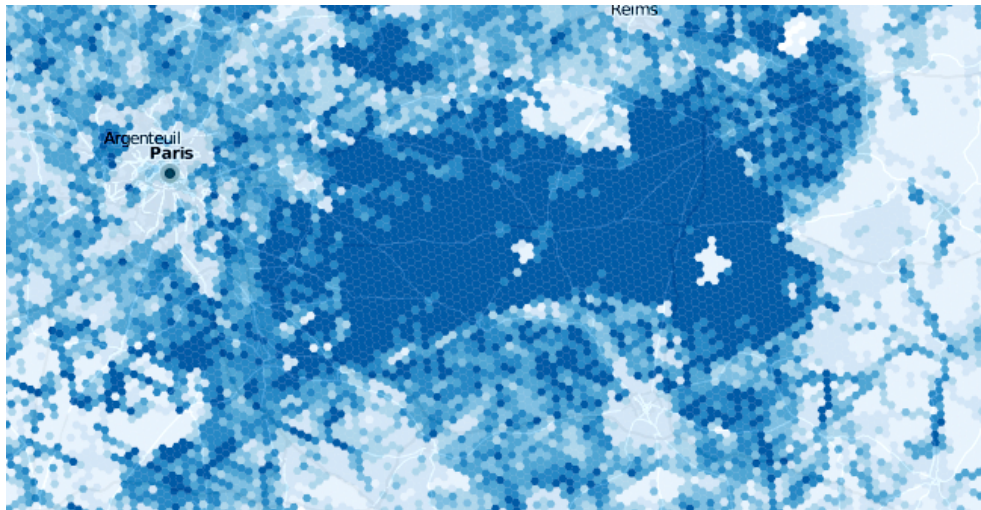


Figure 2: Increased version numbers in the area west of Paris.

In France, active user groups can be found mostly in and around large cities. Some areas might be subject to data imports, indicated by the number of features along with the number of contributing users.

3.4. Spain

The number of features and the sum of all attributes per cell show similar images in Spain. In cities, such as Madrid, Barcelona or Valencia the sum of all attributes is mostly above 2000 and the number of mapped features is at least above 100, which is amongst the largest values amongst Europe. In rural areas the values are mostly below 4.

The average version number shows some interesting patterns. Again, the urban areas around the larger cities show significantly larger values (between 2 and 4) than their surrounding rural parts (below 1). Furthermore, in rural areas linear patterns with values between 4 and 10 can be identified, which mostly matches OSM's tertiary highways (Figure 3).

In general, the number of contributing users and the number of objects modified per user and cell is notably lower than in Germany, the Netherlands and even France. Exceptions are Madrid, the urban areas at coastlines

and some cities in the northern part of Spain. In rural areas the values are mostly around 0.

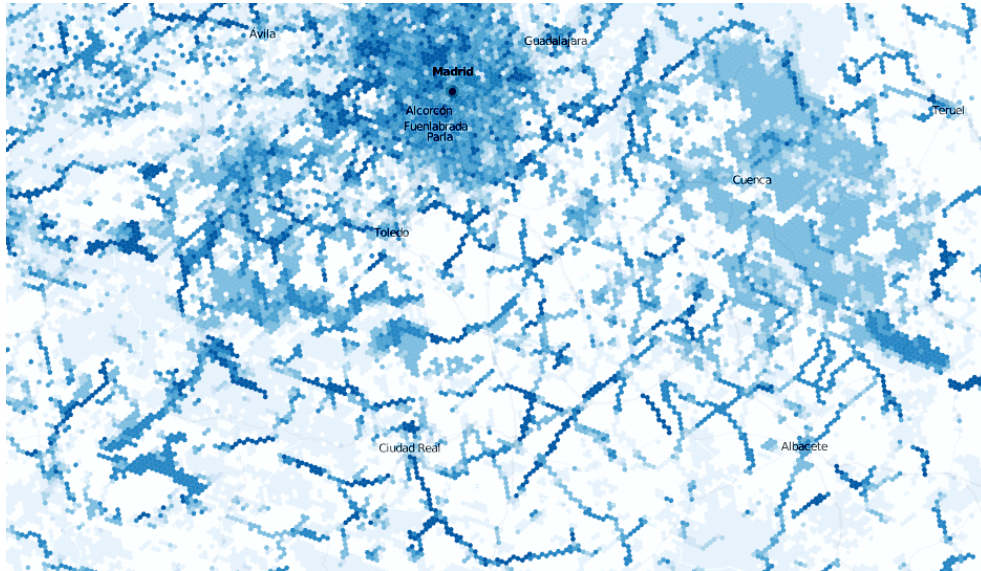


Figure 3: Linear patterns of increased average version numbers in rural Spain.

Also, the number of mapped buildings is lower than in countries we discussed before. Except for Madrid, Barcelona and cities at the northern coast the area covered by buildings is mostly 0. But also the areas of those patterns showing larger numbers do not completely match the areas of the corresponding cities. Thus, it can be expected that even urban areas buildings are largely unmapped. Therefore, it can be concluded, that mapping of buildings has started partly in larger cities but is still far from complete.

In conclusion, most user activities are present in urban areas. Exceptions are few highways in rural areas, which were subject to an increased number of edits. In rural parts of the country almost no activities were visible. This might be related to sparse population densities and a lack of features that can be mapped.

3.5. Portugal

In general, the situation of Portugal is very similar to Spain's. Cities have a more active user community and are thus mapped more completely. On the contrary in rural areas features besides the road network are mostly not mapped at all.

But areas with a significantly larger average number of attributes and a larger average version number can be identified. These patterns match ar-

areas of national parks or preservation areas (Figure 4). A reason for this might be data imports based on data sets provided by the management of the preservation areas.

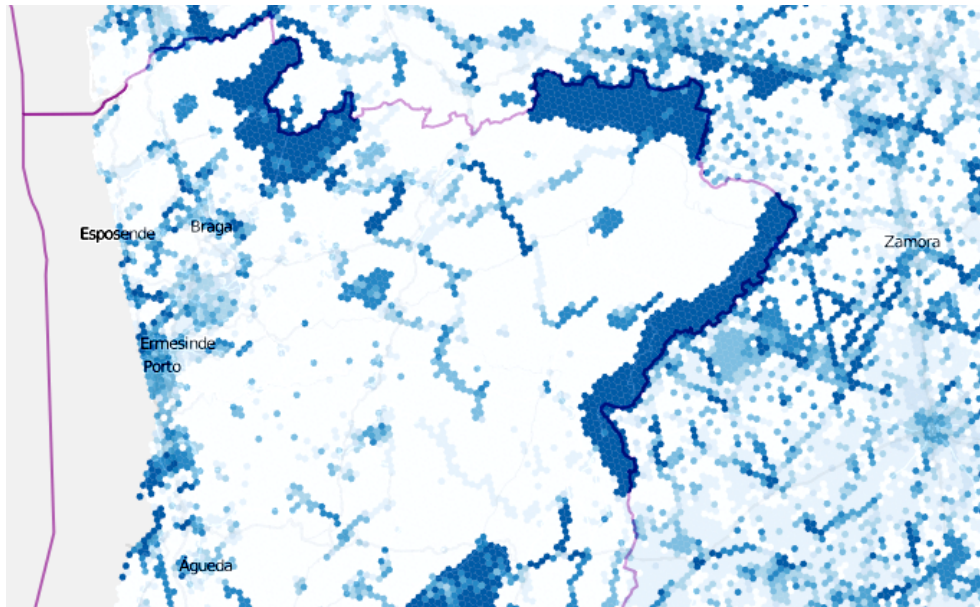


Figure 4: Natural preservation areas show increased numbers of average version number in Portugal.

3.6. Poland

Poland is in many terms different to the previously examined countries. Many attributes show significant rectangular patterns, which unlikely reflect natural activities of OSM users.

First of all, the number of mapped features per cell shows the expected increased values around urban areas such as Warsaw and Poznan. Additionally there are several rectangular patterns in the north and in the south (Figure 5) which may indicate regional data imports. The average number of contributions per user as well as the number of contributing users reflects these patterns as well.

The average number of attributes per object is generally comparable with the numbers in other countries. Northeast of Warsaw and in the south close to Kielce, there are rectangular patterns with values around 9 to 10 in places. In the western part of the country close to Gorzow Wielkopolski one can find an outstanding area with values larger than 10.

Again, identified rectangular patterns might be subject to data imports. This assumption has been approved by comparison with the Mapnik rendering

of OSM data where the same patterns were also visible at natural features or the density of the street network. Thus, quality control and mapping activities are necessary in the surrounding areas.

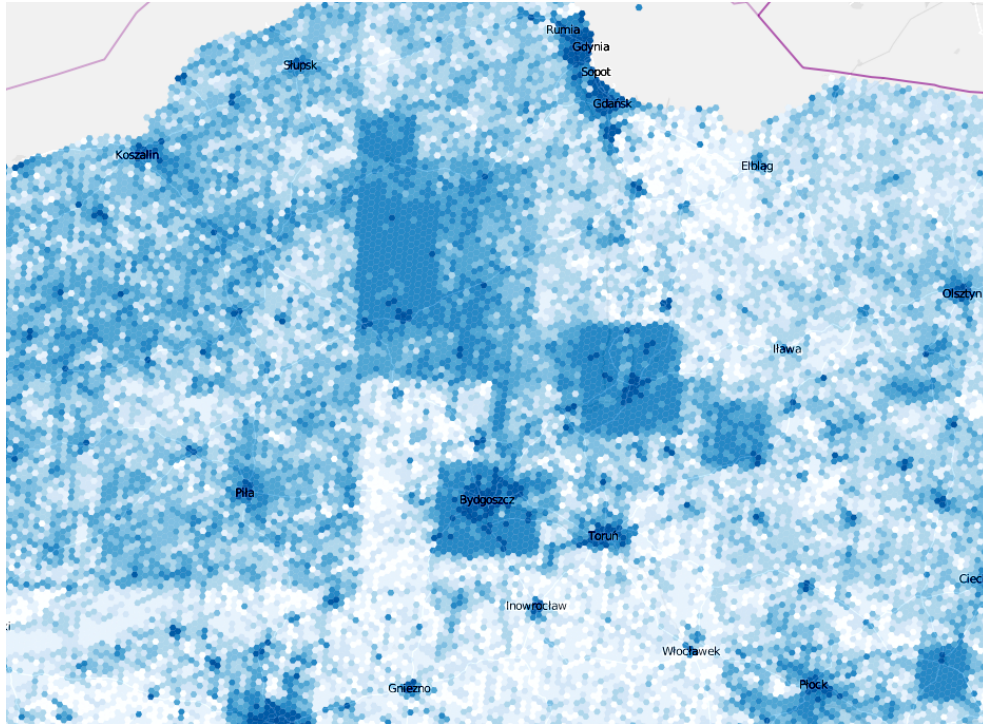


Figure 5: Rectangular patterns of the number of features mapped per cell in Poland.

4. Conclusion

The analysis of the results of the OSMatrix in the previous section revealed, that there are huge differences in terms of user activity and numbers of mapped features between selected countries across Europe.

In Germany there is a very active community and thus a lot of features got mapped, whereas in the Netherlands the relative completeness of the map can be explained with data imports covering the whole country.

For countries of southern Europe it was shown that most activities are restricted to urban areas. This may be related to lower population densities or lesser objects available to be mapped.

In Poland we identified several areas where the values revealed distinct different patterns. These areas might be subject to data imports and thus

we assume that the surrounding areas are still not close to be mapped completely.

5. Future Work

A first goal in future is to improve the overall computational performance of the application. Because of the large number of features that is displayed simultaneously both client and server are temporarily overloaded especially at lower zoom levels. Solutions might be the use of a tile cache service or the use of a grid coverage based data structure at large scales instead of a feature-based structure.

Furthermore, we also aim at providing information on the temporal evolution of the attributes. Therefore, the attributes will be calculated every two months. The results may be portrayed in a single map for each timestamp or as histograms showing the evolution for a given cell. The client's functionality has to be extended accordingly.

Concerning the interpretation of the maps it may be reasonable to relate the found insights to the population density or compare the findings to those of other data sets, for instance those of commercial vendors. Furthermore, it is necessary to investigate how single feature types are mapped in order to find out what gets mapped where and why.

Further, it is important to research, how size and shape of the cells influences the results of the map: Larger or even rectangular cells may alter the results significantly and thus may be taken into account by future implementations of OSMatrix.

References

- Dykes, J., MacEachren, A.M. & Kraak, M.-J. (2005): Exploring geovisualization. Elsevier.
- Goodchild, M. (2007): Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69, 211-221.
- Haklay, M. (2010): How good is volunteered geographical information? A comparative study of OpenStreetMap and Ordnance Survey datasets. *Environment and Planning B: Planning and Design*, 37, 682 - 703.
- Helbich, M., Amelunxen, C., Neis, P. & Zipf, A. (2010): Investigations on Locational Accuracy of Volunteered Geographic Information Using OpenStreetMap Data. *GIScience 2010 Workshop*. Zurich, Switzerland.

- Ludwig, I., Voss, A. & Krause-Traudes, M. (2011): A Comparison of the Street Networks of Navteq and OSM in Germany. In: Geertman, S., Reinhardt, W. & Toppen, F. (Eds.): *Advancing Geoinformation Science for a Changing World*. Springer.
- MacEachren, A. M., Robinson, A. C., Jaiswal, A., Pezanowski, S., Savelyev, A., Blanford, J. & Mitra, P. (2011): *Geo-Twitter Analytics: Applications in Crisis Management*. 25th International Cartographic Conference. Paris, France.
- Neis, P. & Zipf, A. (2008): OpenRouteService.org is three times „Open“: Combining Open-Source, OpenLS and OpenStreetMaps. GIS Research UK (GISRUK 08). Manchester.
- Neis, P., Singler, P. & Zipf, A. (2010): Collaborative mapping and Emergency Routing for Disaster Logistics - Case studies from the Haiti earthquake and the UN portal for Afrika. *Geospatial Crossroads @ GI_Forum '10*. Proceedings of the Geoinformatics Forum Salzburg.
- Neis, P., Zielstra, D. Zipf, A. & Strunck, A. (2010): Empirische Untersuchungen zur Datenqualität von OpenStreetMap - Erfahrungen aus zwei Jahren Betrieb mehrerer OSM-Online-Dienste. AGIT 2010. Symposium für Angewandte Geoinformatik. Salzburg. Austria.
- Neubauer, S., Over, M., Schilling, A. & Zipf, A. (2009): Virtual Cities 2.0: Generating web-based 3D city models and landscapes based on free and user generated data (OpenStreetMap). *GeoViz 2009*. Contribution of Geovisualization to the concept of the Digital City. Workshop. Hamburg, Germany.
- O'Reilly, T. (2005): *What Is Web 2.0? Design Patterns and Business Models for the Next Generation of Software*. URL: <http://oreilly.com/web2/archive/what-is-web-20.html>. Accessed on: May 12th, 2011.
- Trame, J. & Keßler, C. (2011): *Exploring the Lineage of Volunteered Geographic Information with Heat Maps*. GeoViz, Hamburg, Germany.

OSMonto - An Ontology of OpenStreetMap Tags

Mihai Codescu*, Gregor Horsinka*, Oliver Kutz**
Till Mossakowski***, Rafaela Rau*

* DFKI GmbH Bremen, Germany

** Research Center on Spatial Cognition,
SFB/TR 8, University of Bremen, Germany

Abstract. We provide an ontology for OpenStreetMap tags, called OSMonto, which can be used in different ways: (1) browsing of the hierarchy of tags is eased, (2) tags can be related with other ontologies, let it be tool-specific ontologies or more general ontologies like the recent Schema.org ontology supported by the dominant search engines, and (3) concepts can be unified, even if they are tagged in various ways. This will foster the integration of OpenStreetMap into the semantic web.

Keywords. OpenStreetMap tags, ontologies, Semantic Web

1. Introduction and Motivation

OpenStreetMap has evolved into a rich source of geodata. When searching and navigating through a map portal like www.openstreetmap.org, semantic metadata can greatly help with providing an intention and activity-based access to the data. In the case of OpenStreetMap, the metadata is provided in the form of tags that are entered into the database in a Social Web and wiki-like manner. Metadata obtained through such Social Web, collaborative and community based efforts have specific characteristics, namely they evolve in a bottom-up way, contain a lot of noise (typos, redundancies, etc.) and are subject to constant change.

Currently, OpenStreetMap's tags are organised and maintained through a collection of Wiki pages¹ that list the popular tags and specify their intended

¹See <http://wiki.openstreetmap.org/wiki/Tags> and <http://wiki.openstreetmap.org/wiki/Features>

use. In this paper, we propose to complement this by organising the tags into an *ontology*, which we call **OSMonto**².

This will bring the following advantages:

- an ontology provides an easier overview of the tags and their hierarchical structure than a Wiki does. Browsing the tag ontology can be done using ontology editors like Protégé;
- ontology mappings can provide different *views* on the tag ontology:
 - tags can be enriched with an ontological semantics by mapping existing ontologies to the tag ontology;
 - different tags that are used for the same concept (due to local differences or the evolving nature of tags) can be united to one ontological concept through a mapping;
 - search and navigation tools can use their own, purpose-driven ontologies (e.g. an ontology of activities that is shown to the user) and map them to the tag ontology;

The ontological perspective opens the door to *ontology-based data access* that can provide an enriched query language for the OpenStreetMap database. The ontology mappings that are necessary for obtaining the views can be generated semi-automatically or even automatically with the help of ontology matching tools. This approach provides a relatively simple, yet effective solution to the generally rather hard problem of how to relate data to ontologies.

2. Ontologies

Ontologies are formal descriptions of the concepts in a certain domain of discourse and can be informally understood as fixing a meaning for the terms of a particular field. Ontologies are used in artificial intelligence, the semantic web, systems engineering, software engineering, biomedical informatics, library science, enterprise bookmarking, and information architecture as a form of knowledge representation about the world or some part of it. Domain ontologies are typically formulated in the web ontology language OWL³.

Formally, an OWL ontology signature consists of sets of *atomic concepts*, *roles* and *individuals*, which fix the vocabulary. Sentences that can be expressed are of two types: TBox sentences are subsumption relations

²See the project's homepage at <http://osmonto.do-roam.org/>

³<http://www.w3.org/TR/owl2-overview/>

between concepts which are defined inductively from atomic concepts using the universal concept, the empty concept, unions, disjunctions, negations and universal and existential quantification over roles. ABox sentences contain assertions saying that certain individuals belong to certain complex concepts expressible in the vocabulary. Since the ontologies we use here do not contain individuals, we will concentrate on presenting TBox sentences.

Several syntaxes have been designed for ontology languages; in this paper we prefer to use Manchester OWL syntax (Horridge et al, 2006) which provides, for the fragment corresponding to the description logic ALC, the following grammar for concepts:

$$C ::= A \mid \text{Thing} \mid \text{Nothing} \mid C \text{ and } C \mid C \text{ or } C \mid \text{not } C \mid R \text{ some } C \mid R \text{ all } C$$

where R is a role and A is an atomic concept.

The semantics is set-theoretical: an *interpretation* I consists of a non-empty set W (the universe) and an interpretation function $\cdot^I$ which assigns a subset of the universe to each atomic concept, a binary relation to each role and an element of the universe to each individual. The interpretation extends from atomic concepts to complex concepts in the expected set-theoretic way following the grammar, more precisely: the top concept Thing is interpreted as the universe W , Nothing as the empty set (bottom concept), a *conjunction* C and D by the intersection of the interpretations for C and D , a *disjunction* C or D by the union of the interpretations, a *complement* not C by set-theoretic complement, and finally *universal* (R all C) and *existential* (R some C) *role restrictions* as follows:

$$(R \text{ all } C)^I = \{x \text{ in } W \mid \text{forall } y \text{ in } W . R^I(x,y) \text{ implies } y \text{ in } C^I\}, \text{ and}$$

$$(R \text{ some } C)^I = \{x \text{ in } W \mid \text{exists } y \text{ in } W . R^I(x,y) \text{ and } y \text{ in } C^I\}$$

Two ontologies can be related by an *ontology mapping*, sending atomic concepts, roles and individuals of the source ontology to (not necessarily atomic) concepts, roles and individuals of the target ontology. Among many other applications, ontology mappings are important for extracting modules from large ontologies.

3. OSMonto: An Ontology of OSM Tags

OpenStreetMap's internal files are lists of nodes, ways and relations, which can be tagged with information about the respective map element. The convention is that any user is free to introduce his own tags, but it is recommended to use existing tags and only have new ones if they are not already covered by the existing ones. The tags of the map elements are represented as (key, value) pairs. An element of the map may have multiple tags (see be-

low for an example of an OSM node with its tags in an XML representation. This format has been developed by the OSM community. The listed tags vary from node to node).

```
<node id="834034642"
lat="53.0871310" lon="8.8091071"
version="7" changeset="6027662"
user="Kerridge" uid="324245"
timestamp="2010-10-13T09:51:39Z">
  <tag k="addr:city" v="Bremen" />
  <tag k="addr:country" v="DE" />
  <tag k="addr:housenumber" v="20" />
  <tag k="addr:postcode" v="28215" />
  <tag k="addr:street" v="Theodor-Heuss-Allee" />
  <tag k="amenity" v="charging_station" />
  <tag k="name" v="Elektrotankstelle swb" />
  <tag k="note" v="telephone reservation necessary" />
  <tag k="opening_hours" v="Mo-Fr 6:00-18:00; Sa off; Su off" />
  <tag k="operator" v="swb" />
  <tag k="phone" v="+49 421 3593186" />
</node>
```

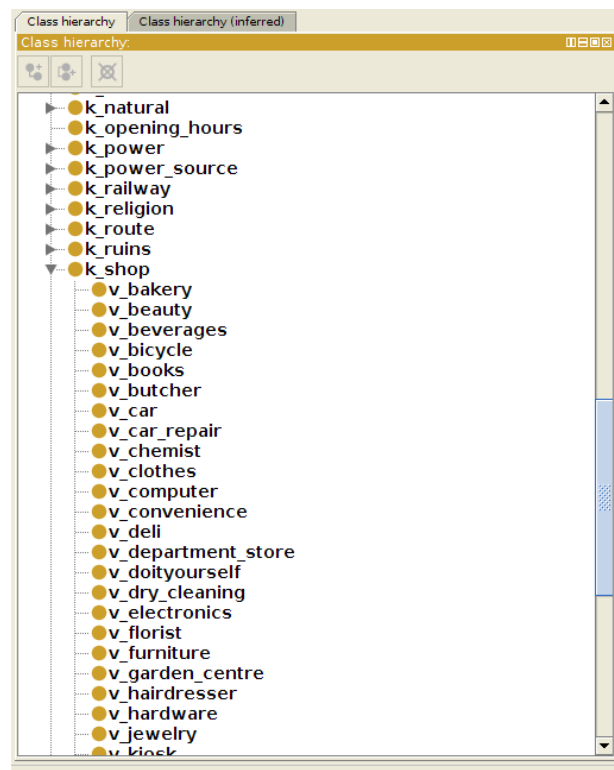


Figure 1: OSMonto: the ontology of OSM tags viewed with Protégé

The ontology of tags, OSMonto, shall stay as close as possible to the structure of the OSM files in order to facilitate ontology-based database querying. This means that we do not try to correct any possible conceptual mistakes in the taxonomy of OSM tags, but rather have it reflected faithfully in the structure of the ontology. Indeed, desired adaptations of the ontology concepts can be achieved by ontology mappings.

When designing OSMonto, it makes sense to decompose the tags into a hierarchy according to the keys: the key becomes a superconcept of its values. We have followed this approach whenever the value was an OSM constant rather than a string/numeral. Since it is possible that a key and a value have the same name whilst the names of the concepts are required to be unique in OWL (OSM has "station" as value of the key "railway" but also a key named "station"), we decided to prefix all keys with "k_" and all values with "v_". E.g.: k = "amenity" and v = "charging_station" would introduce a concept "k_amenity" with a subconcept "v_charging_station". Another problem is that some values are subclasses of more than one key.

E.g., "v_no" is a subclass of "k_smoking" but also of "k_smoking_outside". (We maintain our design uniform, so "v_no" must be a concept; other choices would also be available.) In this case, we extended the value to "v_smoking_k_no". Another design decision was to take into account tag dependencies. For example, when a node is tagged with

k = "amenity" v = "restaurant"

it is possible (but not mandatory) that the cuisine is also tagged:

k = "cuisine" v = "seafood".

In such cases, we introduce a role `hasCuisine` with "v_restaurant" in domain (it is also possible that "v_fast_food" is tagged with "k_cuisine") and range "k_cuisine" in order to be able to select only those restaurants with a certain cuisine.

Recalling the example of French restaurants of the previous section, notice that nation specific cuisines are added directly as subconcepts of "k_cuisine". This is conceptually a mistake in the design of OpenStreetMap's tags, which we here reflect in the ontology which is meant to be very close to the OSM tags.

In order to create a realistic ontology of OSM tags, one faces the problem of an open project where everyone is allowed to contribute - which is also an OSM strength. This has the effect that the data source can be regarded as dynamic, not only at the level of entries, but also at the level of tags. In the OSM wiki page⁴, there exists a list of tags, but this list does not reflect the status quo of the actual OSM databases.

⁴http://wiki.openstreetmap.org/wiki/Map_features

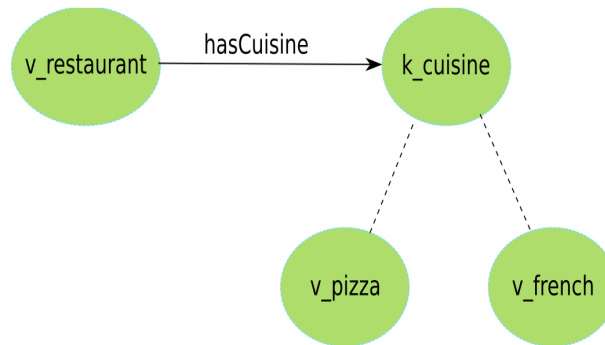


Figure 2: Restaurants with cuisine

Some tags which are in the wiki page are not yet tagged by the community, some tags which were abolished through discussion in the wiki or the mailing lists are still used by the mappers. Therefore, the wiki provides only an overview of the available tags; to have a more realistic estimation, one should use websites like Taginfo⁵, where the OSM data of the whole world is searched and a list of tags in use, sorted by the number of occurrences, is provided as a result. Of course, this list will also contain spelling errors or falsely used tags.

The most straightforward solution here is to consider relevant those tags that have a certain, high occurrence in the database, using the list provided by Taginfo. This strategy could result in a limitation using a certain percentage (e.g., all values with, say, more than 0.3% occurrence rate for the respective key are included), but this approach fails to capture all interesting values in the cases where some keys appear with a far higher occurrence and thus the percentage of important values is low. Also, some keys have far more values (e.g., amenity with 7714 values in use according to Taginfo) than others (e.g., smoking with 22 values), so that the percentage of each value naturally is quite low, which is another point against a certain percentage as a limit for inclusion.

This is why a limitation based on the absolute occurrence of a value makes more sense. In our case, we decided to select all values which occur more than 100 times in the database. Spelling errors are thus excluded as well (there is never 100 times the same mistake), and still all relevant values will be in the database. Theoretically, this threshold could be exceeded by mis-

⁵<http://taginfo.openstreetmap.org>

takes created during automatic tagging procedures. In reality, there is no evidence in Taginfo that this is the case. It is either prevented by the professionalism of those using automatic tagging, or mistakes of such quantity are quickly noticed by the community and repaired.

After this procedure, we added the tags that are in the wiki but not covered through our search of Taginfo. This guarantees that we also include tags which are not yet used by the mappers, but in the future shall be implemented or will replace other tags. To keep this ontology of tags up-to-date, one option is to promote it within the OSM community, as we do through this paper. People creating new tags could include them themselves into the ontology as well. Another option is automation, e.g., programs searching regularly through Taginfo for new tags.

4. Applications

OSMonto offers an easy and compressed *overview* of the keys and their values which are used in OSM. It resembles the page “Map Features” in the OSM wiki, but does not include descriptions of the tags and thus delivers a quicker overview of keys and especially their values. Also, other than the wiki page, it orientates more on the tags which are really in use at the moment (through the Taginfo website). So it is more an interesting device for users who want to make use of the existing database rather than for users interested in information how to tag something. Moreover, since all tags are in English, the ontology provides a high-level, natural-language-agnostic way of browsing the information.

Another possible use of the ontology is the *unification of concepts*. E.g. when searching for a place to swim, OSM offers a wide range of tags. Sometimes this occurs because of changes in the tagging system, which are not immediately taken over by the users in the data. One can quite easily introduce a new concept in the ontology which contains the tags `amenity=swimming_pool`, `leisure=swimming_pool` as well as `sport=swimming`. In OWL notation, such a concept would be defined as follows:

```
Class: Swimming
EquivalentTo: k_amenity_v_swimming_pool or
k_leisure_v_swimming_pool or
k_sport_v_swimming
```

In the DO-ROAM project, we created a unified concept for charging stations for electric cars. It combines the tags `fuel:electricity=yes` (fuel stations with a possibility to charge electric cars) and `amenity=charging_station` (a device solely for charging electric vehicles).

Tags can be enriched with an ontological semantics by mapping existing ontologies to the tag ontology; interesting such ontologies would for instance be OpenCyc⁶ (general knowledge base and commonsense), GUM-Space⁷ (a linguistic ontology of space), or Dolce⁸ (a foundational ontology for cognitive engineering). Of particular interest is the linking together of different such aspects within a network of ontologies in order to combine the different viewpoints into one coherent view (see Kutz et al. 2010).

Also, Schema.org is a recent ontology that shall be used to annotate web content, with the goal of allowing to search the Web semantically (launched jointly by Bing, Google, and Yahoo on 2nd of June 2011). Since this effort is supported by the dominant search engines, this ontology is expected to get high impact in the future. The vocabulary used in Schema.org was inspired by several earlier efforts, in particular Microformats, FOAF, GoodRelations, and OpenCyc. Via an ontology mapping, OSM can be linked to Schema.org.

In the DO-ROAM project⁹, we have developed an *activity-oriented navigation tool* for OSM. This means that the user specifies a starting and an ending point for her trip, then chooses at each time an activity she wants to perform and the system provides markers on the map for the locations where the activity takes place for the user to select. Once all desired activities have been selected, the system generates a route from the starting point to the ending point which includes the selected locations for the user's activities. We have also integrated the energy-efficient route generation of the green-nav.org, using external (not OSM based) data. An older prototype builds on top of the existing Rails portal for OpenStreetMap and complements the free text search of engines like Nominatim with activity search, possibly taking into account opening hours, via both the free text search and in a more guided fashion, by browsing a tree of activities.

The central element in both tools is an ontology of spatially located activities. On one side, it is presented to the user as a interface element for guiding the selection of the desired activity. Moreover, we allow a certain degree of flexibility by performing a lexical analysis on the free text queries of the user and trying to match synonyms of the used words with concepts of the ontology. On the other side, the ontology of activities is connected with the ontology of tags via a mapping.

⁶See <http://www.cyc.com/opencyc>

⁷<http://ontologydesignpatterns.org/wiki/Ontology:GUM-Space>
and <http://www.ontospace.uni-bremen.de/ontology/gum.html>

⁸See <http://www.loa-cnr.it/DOLCE.html>

⁹<http://do-roam.org>

Since the number of concepts and roles is quite large, providing such a mapping manually would be a very tedious process. We can, however, use an ontology matching tool to obtain a list of pairs of concepts that are in correspondence. This approach is very effective - e.g. with the ontology matcher Falcon, the degree of automation reaches 80%.

This means that the user is still required to verify and confirm the matches produced with the tool, and possibly introduce new matchings between concepts that were not identified by the tool's analysis. For example, in the case of the charging stations, the matching tool is able to match the concept with the tag "charging_station", but the semantic analysis is not powerful enough to match it with "fuel:electricity" as well and this must be provided by the user. On the other hand, since the tags for swimming places are syntactically similar, the tool would estimate all of them as possible variants with a high probability and the user can decide to create a concept for the union of the 3 variants.

Ontology-based data access is a data integration methodology that separates the 'knowledge' about data from reasoning about it. This is achieved by providing an abstract representation of the application domain with the help of an ontology, a schema of the sources where the real data is stored, together with a mapping between the elements of the ontology and those of the data schema. Typically, the schema of the data is assumed to be a relational database schema, and the mapping provides a query in the database for each concept and each role of the ontology. The advantage of this approach is that we can use the knowledge base constituted by the TBox and the ABox sentences of the ontology to derive information about the data which is not present in the database, using query rewriting.

The data integration management component of DO-ROAM follows the principles of OBDA: the domain of interest - spatially located activities - is modelled as an ontology, the OpenStreetMap data is stored in a database, and the concepts of the ontology are related to queries in the database.

For the representation of and access to ontologies within the Ruby on Rails framework, we have developed a new library, Rails-OWL. Since OWL is represented in XML, our library is based on the existing library REXML¹⁰ for reading in XML documents. Rails-OWL represents an OWL ontology in the Rails database. This allows programmers to easily and flexibly access ontologies in a way similar to the access of the geodata.

In ontology-based data access, usually, one SQL query per ontology class is designed manually, and this is used for the database interpretation of ontology terms, implemented by query rewriting. In case of OpenStreetMap, we

¹⁰<http://www.germane-software.com/software/rexml/>

would need to design dozens of such SQL queries, which is a tedious process. Instead, we use the OSM tag ontology, which is tailored towards the OSM database in such a way that the relation between classes in the OSM tag ontology and the OSM database is generic: since the basic classes directly correspond to keys and values of OSM tags, the corresponding SQL queries are simple, and this is then used for query rewriting of more complex class terms. This query rewriting is implemented in Rails-OWL easily, because classes, roles and such are first-class citizens.

5. Conclusions

The potential uses of OSMonto, our ontology of OpenStreetMap tags, that we have pointed out can be realised only if the ontology is kept up to date with the current development of OpenStreetMap tags. To ensure this, further research about manual and automatic update facilities is needed, including incorporating related work done for instance in the Web 2.0 context (see e.g. Bindelli et al. 2008). Probably an automatic update via TagInfo and the tagging wiki pages can do most of the work, keeping the number of necessary manual corrections at a minimum. On the other hand, links to existing ontologies might suggest useful ontological structuring principles that need to be implemented manually. Eventually, the ontology may also give some fruitful insights into how to extend and structure the realm of OpenStreetMap tags.

References

- Horridge M, Drummond N, Goodwin J, Rector A, Stevens R, and Wang H (2006) The Manchester OWL Syntax. OWL: Experiences and Directions (OWLED-06), CEUR-WS Volume 216
- Kutz O, Normann I, Mossakowski T, and Walther D (2010) Chinese Whispers and Connected Alignments. Proc. of the 5th International Workshop on Ontology Matching, OM-2010 (ISWC-2010)
- Bindelli S, Criscione C, Curino C A, Drago M L, Eynard D, and Orsi G (2008) Improving Search and Navigation by Combining Ontologies and Social Tags. On the Move to Meaningful Internet Systems (OTM-08), p. 76-86, LNCS 5333, Springer
- Codescu M, Horsinka G, Kutz O, Mossakowski T, and Rau R (2011) DO-ROAM: Activity-Oriented Search and Navigation with OpenStreetMap. Christophe Claramunt, Michela Bertolotto, Sergei Levashkin (Eds.), Fourth International Conference on GeoSpatial Semantics (GeoS-11), p. 88-107, LNCS 6631, Springer

Projected Hill Map – Between Virtual-Reality and Physical Models

Gregory Marler

School of Engineering and Computing Sciences, Durham University

Abstract. Geographic information is an important part of our lives, the research sought to investigate a new alternative system to the traditional flat way of viewing maps. The aim was to investigate if geographic understanding of an area is easier to both gain and share through use of a system that uses a data projector to combine a physical terrain model and a computer-controlled map display. Evaluation participants still drew largely on local knowledge rather than the map, however the physical model helped them to explain and discuss their view with other participants. There was strong evidence of the value added by both the physical model and the changing multiple layers of a computer controlled map. Further work could be made with regards to an input method closer for the participants.

Keywords. Education, GIS, TUIs, HCI, Group Work, Projection, Display

1. Introduction

It is well known that in understanding the geography of a location and connecting places on a map to their counter parts in the real world, the landscape of the area can be critical yet hard to represent in a way that is easy to understand. For many centuries detailed 3D relief models have been hand crafted to scale showing the lie of the land so military leaders, rural communities, building development planners, tourists, and others, can gather to discuss an area of interest. The information available on these models is limited to what is painted on the model, with no option of temporary overlays or loading of additional sources. This leads to it being more common to view flat 2D maps on a computer screen or projected onto a wall. While the wall projection allows many people to view it, it leads back to an issue of understanding areas where hills and valleys have a strong presence. Various cross-disciplinary work between Computer Science and Geography has

come up with systems such as 3D graphics and even fully immersive virtual reality (VR) (Fairburn & Parsley, 1997) where geographic data can be displayed and controlled.

These however either still continue to use 2D displays, or (in the case of VR) are limited to one person viewing the data. Even with the latest technology in virtual reality, the user is still taken out of a natural situation where they would be comfortable. The emerging field of Tangible User Interfaces (TUIs) puts understanding in humans associating with physical objects that they can touch and/or hold. TUIs respond by users interacting with all their fingers (Merrill et al. 2007), and they are free to communicate with voice and body language to other people in the room while operating a system. A solution to the problem would be a physical model which users can physically stand around, point to and touch during discussions, while the data displayed on the model can be instantly changed via a computer. The primary aim of the project is to investigate if geographic understanding of an area is easier to gain and share through use of a system that combines a physical terrain model and a computer-controlled map display, in comparison to traditional flat maps. Focus will be made on how groups of people use the system, as this supports the system being more beneficial than virtual reality solutions.

2. Related Work

2.1. Participatory Geographic Information Science

Participatory Geographic Information Science (PGIS) works on allowing rural communities and tribes to create a map together so that they can document information about their surroundings. When this first developed from creating a flat map to creating the first 3D PGIS model in Africa, it engaged all ages of the 7 tribes/clans involved and “proved to be a catalyst in stimulating memory and in creating visible and tangible representations” (Mutchemi et al. 2009, p37). This modeling exercise involved about 150 people over 11 days.

2.2. Tangible User Interface

Human-Computer Interaction (HCI) looks at ways of interacting with computers, with the output usually being through GUIs (Graphical User Interfaces) and rectangular 2D screens. Ishii & Ullmer (1997) notice that these environments are “separated away from the ordinary physical environment within which we live and interact”, and look at a number of Tangible User Interfaces using metaphors to translate bits of information and actions into physical objects. Of interest is the “Tangible Geospace” application devel-

oped in conjunction with metaDesk platform. A map of the MIT campus is displayed on the metaDesk table, on which a small physical model of the lab building can be placed. The user can rotate and pan the map by physically moving the model of the lab. Other objects are provided with interaction, such as 'activeLens', a smaller screen attached to an arm so the user can move it in three dimensions and use it to explore a 3D model of the campus buildings. The activeLens is used as a secondary view into the map, similar to using a magnifying glass but allowing the user to explore in all directions.

2.3. Video Projection

Some innovation has been made by pointing projectors vertically down onto the floor rather than onto a wall, with the addition of a camera to detect movement over the projected image. Such systems have been installed in classrooms, visitor centres, and even hospitals, and are effective in engaging children using interactive learning (OM Interactive 2010). Several games can have content adapted to suit the subject being taught, and other games/applications have been helpful in working with special education needs (SEN) students.

Artists known as 'video jockeys' control and project animations onto the front of buildings. They perform a process called 'projection mapping' to align parts of their animation with the non-flat shapes of the building facade and graphics can dance around windows, archways, and columns (Ant 2009).

3. Solution

A system was built comprising of multiple components to provide a new user interface on the Virtual Reality Spectrum (Bodum 1997) that could be evaluated. A physical model was made out of white card, onto which a computer projector was pointed, displaying images controlled by software on a computer. *Figure 1* shows how the components connected to each other.

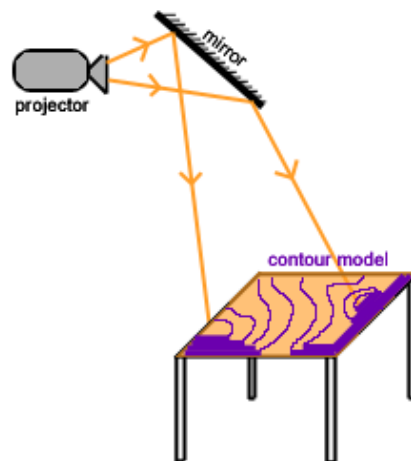


Figure 1. System overview of a projector used to display computer output on a contour model.

3.1. Physical Model

Each contour line was cut out of a piece of 5mm-thick white foam board. These were glued together to create a physical 'hilly model' of the study location, as shown in *Figure 2*. While the model is in 3D, the value of the vertical dimension is in intervals rather than continuous values (the hills are steps, not smooth), this is often referred to as 2.5D. The contour lines were at 5 metre height intervals, to provide enough ranging information about the landscape without over-complicating model interpretation and creation.

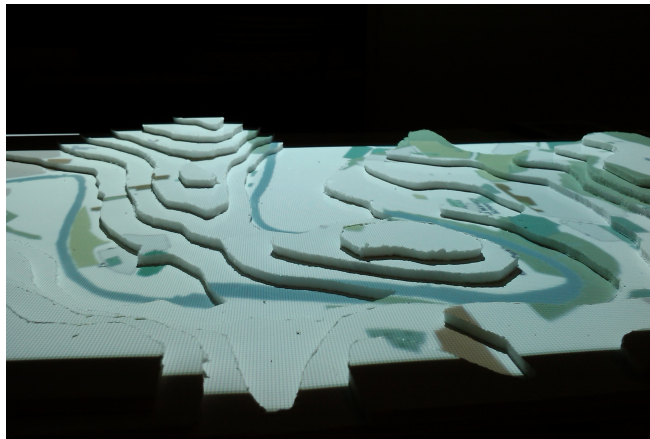


Figure 2. Side-on photo of the model, with rivers and other features projected on.

With the other components of the system created, the contour lines could be projected onto the foam board sheets and traced. This greatly aided construction of the model. With a hand coping saw, and allowing some time for glue to dry on each layer, it took one person 9 hours to build the model. The time consumption was mainly in the cutting of the material. It is thought that with experience and with good cutting tools, the construction time could be even quicker. The time taken is considerably shorter than the construction of a 3D model with smooth terrain and painted surfaces, that can take teams multiple weeks to complete being unaided by a computer system. (Muchemi et al. 2009)

3.2. Computer Projector

To connect the physical side of the system to the computer-controlled side, a computer data projector was placed to the side of the model's table but at a higher position. A frame was constructed around the projector to hold a mirror that would direct the projector output down onto the white contour model. Various aspects had to be accounted for that dealt with issues arising from the use of a mirror that would distort the image. The keystone fea-

ture on the projector firmware was used to remove the trapezium-shape and the operating system settings mirrored the image back to normal. The high brightness, 2500 lumen, projector allowed the system to be used in normal ambient light so users can communicate with each other.

3.3. Map Display Software

To control what is displayed on the white contour model, via the projector, a program was built using HTML and javascript. This was run in a standard browser extended over two screens (the second screen being the data projector). The choice of HTML/javascript was made to allow quick development of an interface to control the display output. It also allowed the use of OpenLayers, “An opensource javascript library to load, display and render maps from multiple sources on web pages” (OpenLayers 2011). Controls for the position where the map is displayed, and at what size, are provided, so that the model or the projector does not need to be physically shifted or the zoom adjusted.

3.4. GIS System

The OpenLayers library is used to display geographic features, that are then projected onto the contour model. While a full advanced GIS system was beyond the scope of this project, through the use of a customised control of the OpenLayers instance a basic GIS has been made. This system could be developed further to make use of other geospatial features and tools. The map instance can display a number of map features ‘layers’, using either vector tiles or tiled map images. The display of each layer can be individually shown or hidden and the ordering of each layer changed. Features can be shown so that more information is available to the viewer(s), or features/layers hidden so that the display on the model is clearer and less cluttered.

The vector layers, loaded from OpenStreetMap .osm files, describe the geometry of a certain feature. This allows our system to fully control how the features are displayed on the contour model. The width/size and colour of lines and points can be changed by the user. Combined with a feature-full GIS, vector layers could contain geographic information that was specific to the current use of the system. For example, a vector layer could be proposed areas for a new business site, identified dangers, or monitored locations. The downside of vector layers is that they require more processing power, and this was extremely prevalent for city-wide features(e.g. roads) in our system that used a web browser.

Map tile images can also be used in the system. This is the defacto standard for web maps. Standard image files that have been rendered and saved are

named in a convention so that the system can use the appropriate images for the location and tile the images next to each other. Browsers are optimized to displaying image files and so the processing power is minimal, overcoming the issues encountered with vector layers. As a reference between geographic co-ordinates and screen pixels is maintained, vector layers can still be used at the same time and shown above the image tiles.

4. Results

4.1. Participant Sessions

To evaluate the map display system that had been created, sessions were run with focus groups consisting of 2-4 members. Each participant answered a pre-demonstration questionnaire that gathered information about their demographic, computer/technology level and map experience. All the participants currently lived in the area displayed, or within 4 miles. However the depth of local understanding varied between participants, with some visiting only a few specific places.

The computer interface was manned by a facilitator, the participants were provided with a list of the layers that they could request to be displayed or hidden. Five group sessions took place, comprising of a total of 15 participants. 11 of the participants were undergraduate or postgraduate students. All participants were aged between 18 and 40.

4.2. Session Timings

The geographic questions used for the project involved hill/landscape knowledge and tried to be comparable in difficulty, however it is inevitable that there will be differences between answers so timings are only compared when the same questions are used. *Table 1* details how many groups used each question pack. The question packs A and C will not be looked at, as they can not be used to compare timings for both situations.

Question B1 “What buildings are at risk of flooding” was open-ended on how many answers could be given, and thus many were considered and a longer time taken than with other questions. Almost half the number of answers were considered when the model was not in place.

Participants took twice as long with the model when answering question B2, “Where would you watch the sunset”, but they came up with the same

Question Pack	Groups used with model	Groups used without model
A	2	0
B	2	1
C	0	3
D	2	1

Table 1. Number of groups using each pack(set) of questions.

number of answers. Many theories could be considered, but it is thought the question requires more information than was available through the map so even with the model the participants could not see any other possible answers irrespective of how long they looked at it.

Question B3 “Plan a route from point A to point B avoiding going uphill.” and question D3 “Plan a route to walk from the cathedral to the railway sta-

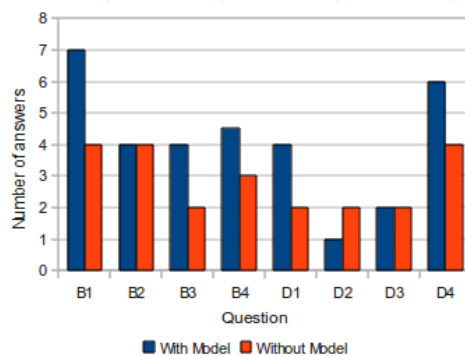


Figure 3. Average number of answers discussed.

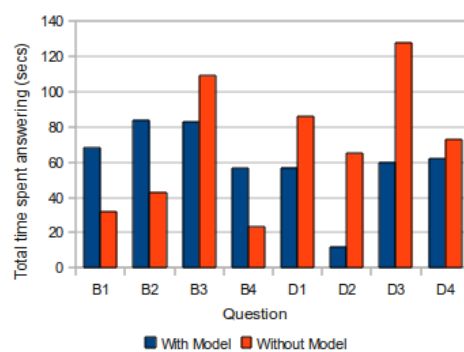


Figure 4. Average time between reading the question and the group agreeing on a final answer(s).

tion” were both about a number of choices rather than choosing one point. The participants had usually gone to the cathedral or the railway station, so not many answers were considered for D3. They both have similarities though as the longest time was spent on these questions when the model was in place, and without the model participants took twice as long. Question B3 supports the Projected Hill Map system as an average of 4 answers/routes were considered, whereas only an average of 2 were considered without the model, yet a longer time was taken. This may have the result of the best answer not even being considered.

Question B4 “If a large event is on, where might traffic form?” supports the system with the model, but doesn’t stand out with any particular differences. The question does not rely on knowledge of the 3D landscape.

Question D1 “What college is the highest” had twice as many answers considered with the model. Without the model groups used local knowledge to think of two colleges and then attempted to work out which of them was higher. Whereas with the model some colleges were considered and even chosen that weren’t normally. This is because participants who went to the university knew routes between colleges and so they thought which one was further ‘up the hill’ when going further along the main road and not considering the actual college site or when the road levels out.

Question D2 asked “What route did you take to get here”. Fewer answers were considered with the model, which relates to the question having a clear correct answer and the thought that the model makes it easier for participants to understand the map.

Question D4 “With unlimited money, where would you live?” was similar to question B1 in being open-ended and so many answers were considered, even more with the model. It took longer for a first answer to be reached when the model was in place, however without the model the group took longer overall despite not considering as many answers. From watching the group, it seems the participants who had the model thought about the question more before sharing it with the rest of their group.

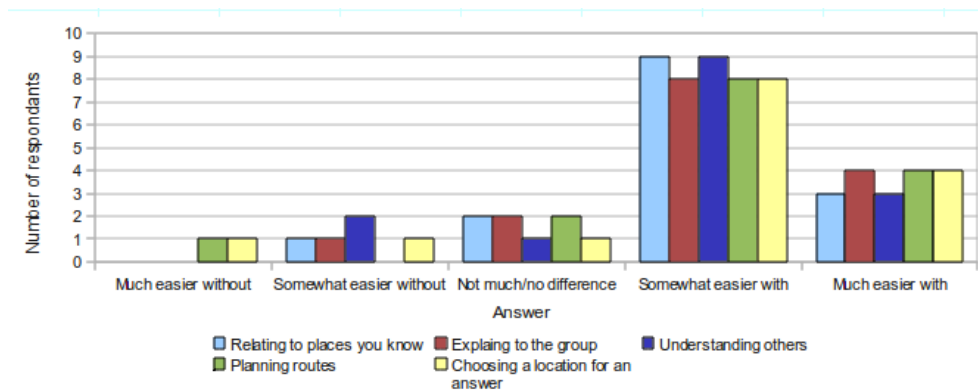


Figure 5. What aspects participants found easier with or without the model.

4.3. Post-session Questionnaire

After the participants had used the map with and without the model, they answered some questions about how it went. *Figure 5* shows the majority of participants found the sessions to be somewhat or much easier with the model for various aspects. The model made less impact on planning routes and choosing locations for an answer. This may be due to people believing they already know the best answer from their local knowledge and the model really aided them when they had to tell others or understand where on the map their answer was.

The majority of participants found answering questions with the model to be easier or more enjoyable, some participants said it was interesting or helpful. They commented that the model aided with imagining views, recognising places, and finding routes. Without the model comments had negative feelings that extra effort was required, it wasn't as fun, and it was boring, or “like using a normal map”. This supports the suggestion that not us-

ing the model was comparable to current map uses and that users are often uninspired by such maps. One admission of “got a couple of locations wrong” was made, perhaps explained by another participant's comment the map was “more open to imagination”. None of the participants seemed to have past experience with GIS systems so all found the ability to show and hide features “very helpful”, “really good” and agreed it “improved clarity”.

When participants were asked what they thought the model would be good for, most of the suggestions related closely to the questions they had been given to answer using the map. Hence planning walks/cycles and tourism use were the most common suggestions. A unique suggestion was “photography (identifying the best places for shots)” which highlights that the system allows users to discover new places without having to leave the room. It also draws upon the aspect that maps give an overview of the area so you can picture what could be seen from one point. This is especially clear when you can take hills into account that block views or provide focal points. Three participants noticed the application for use in the classroom as a good situation, “I reckon children would really like it”, with a teacher suggesting that it would be useful for “teaching children about where they live”.

5. Conclusions

The project investigated a new way of presenting geographic terrain and information with the aid and adaptability of a computer system. It successfully created a prototype to demonstrate the method and allow participants and researchers to investigate how it worked and how it might be helpful in a wide range of situations. The research should be seen as a pilot study, since time constraints limited the sample size and the range of available participants (a more diverse selection of knowledge about the area used would be desired).

However, the results strongly suggest that a 'Projected Hill Map' was beneficial, compared to a projected flat map on a table. The results of the evaluations support that group work is better with a physical model, and that the system could be a viable way to incorporate computer controlled display. The benefit of the computer was not initially apparent to the participants, as they were not familiar with having layer features or changeable information available. However, this work will allow those already familiar with GIS, to consider the potential applications of a system with a physical model for their own work, particularly given the improvements it can make to group discussions.

The model provided hands-on interaction with the map as more pointing took place and height differences were worked out by touching and count-

ing contour levels, where as with no model the users tried to read the values of the contour lines which may be blocked by other features.

More answers were considered with the model and this sometimes meant more time spent on discussions. However without the model, participants spent more time before finding their first possible answer, even having trouble understanding places they had been (e.g. Question D2). Overall discussions often took longer despite less answers being considered. Participants missed answers that got considered with the model, admitting that without the model the map was “more open to imagination”.

The model was particularly good for planning and discussing routes (e.g. questions B3, D3). Supported by how participants felt about using the model. Question D1, along with the suggestion of using the model for finding the best places for photography shots, promotes the idea that the system enables users to see the overall picture and think beyond their focused knowledge.

5.1. Further Work

The construction of the model, although considerably faster than traditional 3D model creation, poses a limitation to how, when, and where the system could be used. Even some of the participants for the evaluation commented that the system was limited to one geographic place (although if other models had been made they could easily have been swapped). Studies could focus on how the software could be extended to produce files that could be used by computer-driven machine cutters. It may be possible to explore a dynamic contour model through use of an array of individual white blocks of which the vertical position can be changed using computer controlled mechanics from below the board.

The system provides no interaction/control other than through the computer terminal situated aside from the model. User control could be developed in a number of ways, aiming to preserve the physical concept of the model and natural interactions. Systems where projectors point to the floor, have used cameras to detect movement across the projected image and interpreted this as user feedback to select or control various items. The use of a tablet PC (e.g. iPad) to host the user interface/control panel may provide a technically-simpler means of interaction with the computer while around the model. A tablet PC can be passed to other people in the group, it can be placed down or picked up as required, so hands are free to interact with the model and with the group.

The project had a Computer Science focus, but it's findings suggest that other disciplines may wish to investigate use of the system further. Geogra-

phers are used to exploring and understanding maps but the Projected Hill Map system might represent a powerful tool to explain their work of a geographic area to other parties. In education this system could aid classroom discussions across subjects, providing a way in which children can understand what they are looking at and engage with the subject matter in a geographic context. An English student in the participant evaluation commented that they didn't like reading maps on trips but had enjoyed answering questions and found it fun discussing answers with the model. They said the model would be good "especially [with] younger people".

It would be insightful for another study to be performed with more participant groups and an improved pre-observation questionnaire. Including questions such as what transport methods participants used in the area, to gauge the granularity of their locality-understanding. A second evaluation into the effectiveness of the system should look at participants who had varying knowledge of the area, particularly people who had never been to the city, had lived there a few weeks, and those who had lived there for much longer.

References

- Ant (2010) Antivj – audiovisual installation on the Hotel de Ville, enghien-les-bains, France.
<http://antivj.com/desherece/index.htm>. Accessed 11 September 2010.
- Bodum L. (1997) Future Directions for Hypermedia in Urban Planning, Spatial Media and Virtual Reality.
- Fairburn & Parsley (1997) The Use of VRML for Cartographic Presentation, Computers and Geoscience 23 (4) . pp 475-81.
- Ishii H & Ullmer B (1997). Tangible bits: towards seamless interfaces between people, bits and atoms, in 'Proceedings of the SIGCHI conference on Human factors in computing systems', CHI '97, ACM, New York, NY, USA, pp. 234–241.
- Merrill D, Kalanithi J & Maes P (2007). Siftables: Towards sensor network user interfaces.- Proceedings of the First International Conference on Tangible and Embedded Interaction.
- Muchemi J, Crawhall N, Pedone G, Koinante J, Kiptum Y & Kuria K (2009). Kenya case study: Participatory 3-dimensional modelling. African Indigenous Peoples' Workshop on effective use of Information Communication Technology (ICTs) in environmental advocacy .

OpenWebGlobe SDK - An Open Source High-Performance Virtual Globe SDK for Open Maps

Martin Christen, Stephan Nebiker

University of Applied Sciences Northwestern Switzerland FHNW
Institute of Geomatics Engineering

Abstract. Virtual globes, sometimes also referred to as 3D geobrowsers, consist of virtual 3D environments capable of streaming and interactively displaying large amounts of geo-referenced spatial contents over the internet. This paper introduces the OpenWebGlobe project and virtual globe technology whose development started at the University of Applied Sciences Northwestern Switzerland. The virtual globe has been in development for many years and has been successfully implemented in several research and industry projects. The new version of OpenWebGlobe encompasses the viewer and the processing functionality. It features a completely new multi-language architecture, and has recently been released as open source. In this paper we introduce the OpenWebGlobe SDK and the OpenWebGlobe processing functionality. We then present first results from processing large geospatial data sets on a High-Performance Compute Cluster and from integrating custom-rendered OSM data with a special focus on improving the 3d effects with the generation and inclusion of an additional hillshade layer.

Keywords. Virtual Globes, WebGL, Open Source, High Performance Computing, Cloud Computing

1. Introduction

Virtual globes (VG), sometimes also referred to as 3D geobrowsers, consist of virtual 3D environments capable of streaming and interactively displaying large amounts of geo-referenced spatial contents over the internet (Nebiker et al., 2010a). In their few years of existence, commercial virtual globes such as Google Earth or Microsoft Bing Maps 3D have received enormous attention from both mass media and the geospatial community itself. In a recent survey (Nebiker et al., 2010b), carried out on behalf of the European Spatial Data Research network (EuroSDR), the authors collected

facts and opinions about prevailing VG technology and platforms, their application, their economic and political impact, geospatial base data and contents of VGs, quality issues as well as questions of standardization and extensibility. In short, it can be stated that the dominant commercial VGs offer a global coverage with an almost incomprehensible amount of geospatial base data and a very tight integration with the search technologies of the respective companies. In addition commercial VGs offer relatively simple but effective mechanisms, such as the KML markup language, for integrating user-generated contents, namely points of interest or 3D objects. However, the dominant VGs also have a number of shortcomings such as a lack of extensibility in terms of large or complex user-generated and/or third-party geospatial and spatio-temporal contents (e.g. own high-resolution DTMs or large to very large customized map or orthoimage data sets). Other potentially more serious shortcomings are the lacking extensibility in terms of functionality and the lacking ease of integration into third party applications and into operational environments which might not be compatible with data and IP right policies of large commercial VG operators.

The strive for totally new applications of VGs together with the above mentioned limitations of the main commercial VGs have been the motivation for developing our own VG technology at the University of Applied Sciences Northwestern Switzerland (Christen & Nebiker, 2010). Our virtual globe, which has recently been made open source and renamed to "OpenWebGlobe" has been in development for many years and has successfully been implemented in several research and industry projects (Eugster et al., 2008).

2. The OpenWebGlobe Technology

OpenWebGlobe consists of a high-performance 3d geobrowser and it also encompasses the software for processing very large volumes of geospatial data in highly parallel and scalable computing environments. OpenWebGlobe supports several data categories such as image data, elevation data, points of interest, vector data, and 3D objects. Before streaming over the internet, this data must be pre-processed. Such pre-processing usually comprises a transformation from a local to a global reference system, creation of pyramid layers or levels of detail (LOD), tiling of the data, and optionally compression and encryption. Because data sets are usually large, containing several terabytes, advanced out-of-core rendering mechanisms with a level of detail approach are used for visualization (Christen & Nebiker, 2011). Recently, these algorithms were ported to high performance compute clusters using OpenMP and MPI and were also released as Open Source.

Our OpenWebGlobe SDK is an open source SDK (Software Development Kit) consisting of more than 300 functions for virtual globe rendering. With this SDK it is possible to create new and innovative applications containing a virtual globe. The SDK should not depend on a computer language. Therefore, it is planned to either use C, C++, C#, Visual Basic, Python or JavaScript (WebGL) to create new applications

The OpenWebGlobe SDK doesn't include any geospatial data; therefore it relies on free or open image, map, and elevation data. With the advent of OpenStreetMap being available on a global scale, this data provides an ideal basis for OpenWebGlobe based applications. We process OpenStreetMap data on a high-performance computing cluster. By adding analytical hill-shading to the OpenStreetMap tiles, interactive virtual globe rendering of amazing quality can be obtained.

3. The OpenWebGlobe SDK

3.1. Overview

The OpenWebGlobe SDK (www.openwebglobe.org) is available in two versions: a JavaScript/WebGL implementation for plugin-free web applications and a C++ implementation with multi-platform support and multi-language bindings for the integration into a wide range of application frameworks. The two branches are shown in figure 1.

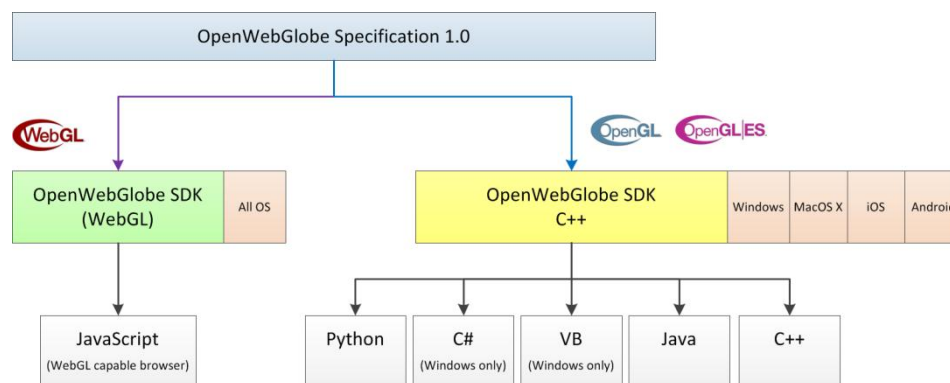


Figure 1. Two branches of the OpenWebGlobe SDK: WebGL version (left) and native code branches (right).

At the moment we focus on the WebGL branch. The C++ branch will be released at a later time. The source code of the WebGL version is available at: <https://github.com/OpenWebGlobe/WebViewer>.

The source code is licensed under MIT license.

3.2. Basic Objects

The OpenWebGlobe SDK uses an object oriented approach, but the programming is done by calling functions only. This allows us to create bindings for many different languages.

Among the basic objects (see Fig. 2) is the context object which is the combination of a render window and a graphics engine. A special mode allows windowless applications, which is useful for creating server-side applications or 3D query engines.

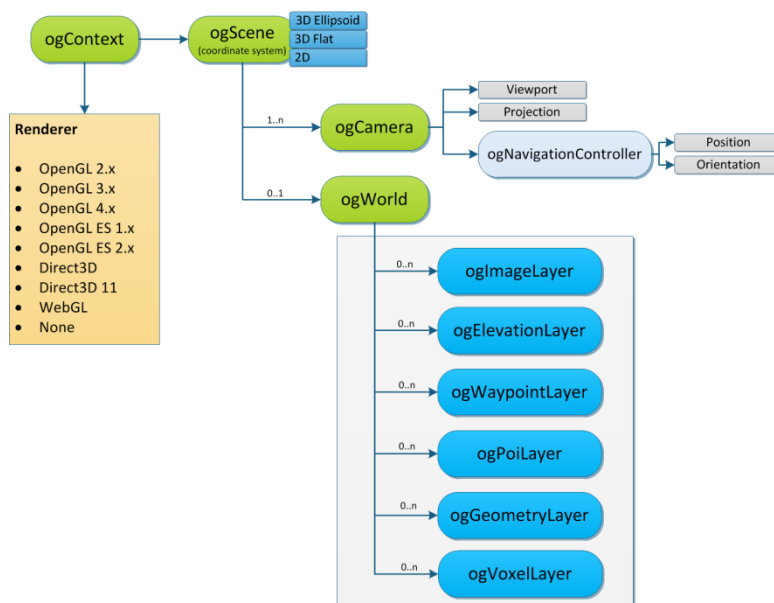


Figure 2. OpenWebGlobe basic object hierarchy

The scene object describes the type of virtual globe visualization. A virtual globe is usually represented as 3D-ellipsoid (WGS84), but it is also planned to create flat earth representations especially for local scenes or to create a 2D visualization which is usually a 2D-map-type application.

In the scene there is a camera object which controls what is visible. There can be several cameras in a scene, but only one is active at the same time. The camera has a navigation controller which allows navigation through the scene – for example with the mouse and keyboard as input. There is a de-

fault navigation controller available and a custom navigation can also be implemented.

The world object contains the globe in the format specified in the scene before (ellipsoid, flat, 2d). The world consists of different layers which can be added or removed during runtime. The most important layers are the image layer consisting of previously tiled orthophotos or maps and the elevation layer consisting of previously processed (tiled geometry) elevation data (DEM). Both layers allow streaming of nearly unlimited data. The waypoint layer contains waypoints, for example previously tracked points with a GPS receiver. Furthermore the POI layer contains points of interest (POI), the geometry layer contains 3D-objects and the voxel layer contains preprocessed (tiled) 3D point cloud data. All layers are streamable over the internet or also from a local area network or local hard drive.

Figure 2 shows the basic hierarchy of the basic OpenWebGlobe objects.

3.3. A Simple Application

```
#include "og.h"

int main(void)
{
    int context = ogCreateRenderWindow("Hello Globe", 640, 480);
    int globe = ogCreateGlobe(context);
    ogAddImageLayer(globe, "url=http://bla.com/ layer=world500");
    ogAddElevationLayer(globe, "url=bla.com/elv/ layer=srtm");
    return ogExec();
}
```

Listing 1. A simple OpenWebGlobe application written in C++.

Listing 1 shows a minimal OpenWebGlobe application. The function `ogCreateRenderWindow` is used to create a render window. There would also be a more complex function `ogCreateContext` which provides more options. The return values of these functions are an id to a context object. This id is required for other functions. It is also possible to create several render windows at the same time.

The second function `ogCreateGlobe` directly creates a WGS84 scene containing a camera and a world object. This is a convenience function: there are also functions available to create scene objects, camera objects and world objects manually.

The next functions add image (`ogAddImageLayer`) and elevation (`ogAddElevationLayer`) layers to the scene. These functions also return an id to the objects but they are not used in this mini application. Different tile layouts are supported, so it is possible to add OpenStreetMap tile servers and therefore to use OpenStreetMap Tiles in the 3D world.

With ogExec the virtual globe application starts until a terminate command is used or the window is closed. You can add custom events for example to control the mouse input, or the keyboard input, or the rendering mechanism, or the loading of asynchronous objects. Starting from here you can add code to display 3D geometry, points of interest and so on. Figure 3 shows a screenshot of such an application using high resolution data.

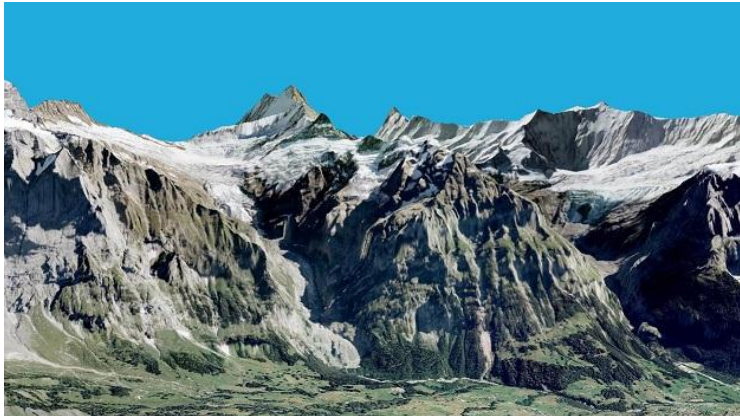


Figure 3. A hello world application streaming data with 50cm² pixel resolution. (Geo-data: SWISSIMAGE, DHM25 © swisstopo JD100033)

Listing 2 shows a simple OpenWebGlobe application written in HTML/JavaScript for WebGL rendering. This listing uses data from the OpenStreetMap Server.

```
<script type="text/javascript" src="openwebglobe.js"></script>
<script type="text/javascript">

function main()
{
  var ctx = ogCreateContextFromCanvas("canvas", true);
  var globe = ogCreateGlobe(ctx);
  var imgOpenStreetMap =
  {
    url      : ["http://a.tile.openstreetmap.org",
                "http://b.tile.openstreetmap.org",
                "http://c.tile.openstreetmap.org"],
    service : "osm"
  };
  ogAddImageLayer(globe, imgOpenStreetMap);
}
</script>
<body onload="main()">
<div style="text-align: center">
<canvas id="canvas"></canvas></div></body>
```

Listing 2. An OpenWebGlobe application written in JavaScript (WebGL) which uses OpenStreetMap data.

3.4. Advanced Objects

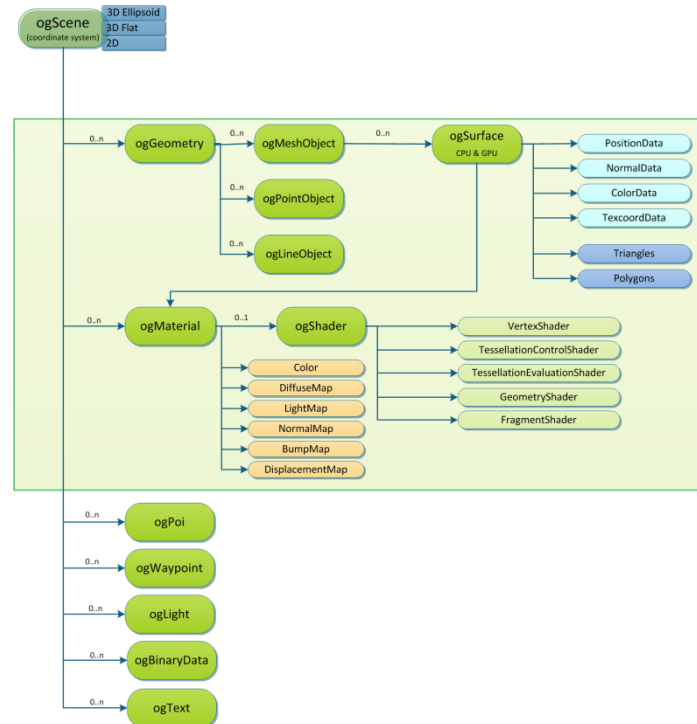


Figure 4. Advanced OpenWebGlobe Objects

As shown in figure 4, there are also some more advanced objects for 3D-Geometry, POI description, binary or multi-language text data. In summary:

- The geometry object describes a 3D-object which can be either a mesh containing 3D surfaces, or a point object consisting of points only, or line objects containing lines only.
- Surface objects should have a material which optionally has a GPU based hardware shader.
- POI objects describe a single point of interest.
- Waypoint objects describe a single waypoint (with references to its neighbors)
- Light objects describe a light in the scene.
- Binary objects describe a chunk of binary data.
- Text objects describe a multi-language utf-8 encoded text.

4. OpenWebGlobe Data Processing for High Performance Computing Environments

4.1. OpenWebGlobe Processing Overview

The OpenWebGlobe data processing algorithms (Christen et al., 2011) have been developed with a focus on scalability to very large data volumes – for all supported data types, including imagery, map and terrain data – and have been optimized for parallelism. We adapted the algorithms to support as many cores as possible and came up with a set of OpenWebGlobe processing commands. All commands run on normal computers (regular laptops and work stations) and on HPC systems, including cloud services. We use the Message Passing Interface (MPI) to communicate and distribute the workload on our cluster and OpenMP for multiprocessing.

Our development and test platform (see following section) runs on the Microsoft Windows HPC Server 2008 R2. However, our code is cross platform and runs on Linux, too. Commercial clouds, for example Amazon Elastic Compute Cloud (Amazon EC2) are also supported by the OpenWebGlobe data processing code.

One important aspect is compatibility to other geospatial data services. Therefore, we chose to use the OpenStreetMap tile layout for our tile storage. This makes it very easy to also use our processed data in 2D applications like OpenLayers.

4.2. OpenWebGlobe Processing Commands

Configuring the Processing Tools

Configuration is done in an XML file (setup.xml) where you specify the directory where data will be processed and a directory for logging. The XML configuration file is quite simple and looks like this:

```
<ProcessingSettings>
  <processpath>/home/mdc/process/imagedata</processpath>
  <logpath>/home/mdc/logs</logpath>
</ProcessingSettings>
```

Creating a Layer

Layers are used to display data on the virtual globe. Among layers there are image layers for image data, elevation layers for 3D terrain, POI layers for points of interests.

With the **ogCreateLayer** command it is possible to define a new layer. New layers must have an extent. We use tile coordinates to define the extent (see figure 5). If tile coordinates are unknown we provide the **ogCalcEx-**

tent tool to calculate extents of datasets in various ways, for example from a WGS84 boundary.

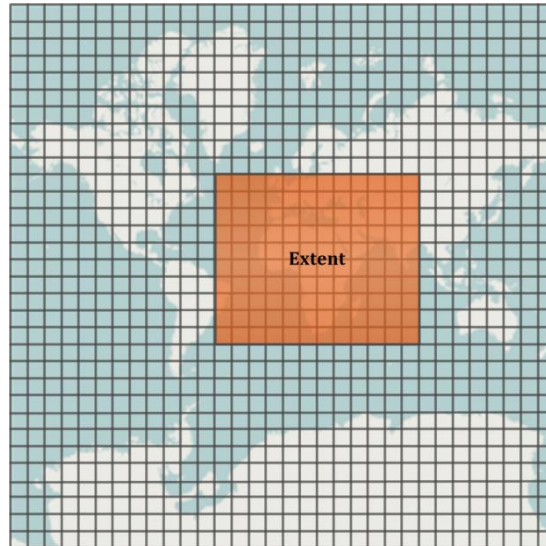


Figure 5. An extent for a layer is defined in quadtree tile coordinates.

Example:

```
ogCreateLayer --type image --name OpenStreetMap -lod 18 --extent  
12341 4321 12455 4830
```

This creates a new layer called "OpenStreetMap". It will have a maximal level of detail of 18 and the extent provided in tile coordinates.

Adding Data to the Layer

Adding data to a previously created layer can be done with the **ogAddData** command. This only adds data in the extent of the dataset. If data is added outside that boundary, it is ignored or cropped. If datasets are overlapping, the overlapping area will be filled with data from the first loaded dataset, which is not in a defined order. Therefore it is recommended the different datasets fit together.

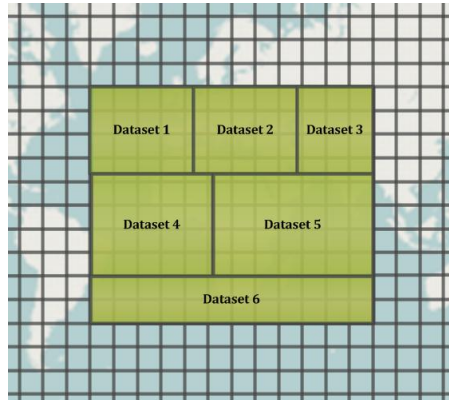


Figure 6. Different data sets are added to the previously defined extent.

The **ogAddData** command can be called several times at the same time. It makes sense to use one **ogAddData** command per compute node. The **ogAddData** command itself uses all cores on the current machine.

Example:

```
ogAddData --image myImage.tif --srs EPSG:4326 --layer OpenStreetMap
-fill
```

Triangulating (Elevation Data)

If you added elevation data you have to triangulate it. There are two modes: delaunay triangulation and regular grid (Christen et al., 2011). This will triangulate the dataset with specified options. It is also possible to provide parameters for thinning out the dataset.

Example:

```
ogTriangulate --layer SRTM --TIN --maxpointspertile 500
```

Resampling

Resampling the dataset is one of the most important tasks: it calculates all level of detail tiles for the dataset. If the dataset is an image dataset the lod is created by reducing 4 images to 1 (along the quadtree).

If the dataset is an elevation dataset, the level of detail is calculated by thinning out the points with lowest weight and by creating a new triangulation.

Updating Data

If a region changes, it may be updated by calling **ogAddData** with the "overwrite" option. After adding data the area must be triangulated (elevation only) and resampled again.

Resizing a Layer

Sometimes it may be necessary to increase or decrease the extent of a layer. This can be done with the `ogResize` command. After resizing you should add new data and resample the whole dataset again.

Failure

During processing a power failure may happen or a disk crashes or whatever. It is very important that work is not lost on a failure. After a failure you simply call the `ogCleanup` command and restart the processing.

Data Deployment

Image Data can be deployed for usage on a server. Deployment means that data can be compressed (for example compressing image data or writing elevation data in a certain format). Deployment also allows to write all dataset into a .tar archive so that it can be transferred to another place, for example in a datacenter or to the cloud.

5. Experiments and Results

In this section we introduce our high-performance computing test environment and present first results from processing different geospatial data sets for the OpenWebGlobe virtual globe.

5.1. Our Hardware System

Our Institute recently acquired a small high performance compute cluster (HPCC) from DELL. It consists of a head node and 4 compute nodes, has a storage capacity of 120 TB, and runs on the Microsoft Windows HPC Server 2008 R2 operating system.

The head node is a DELL PowerEdge R710 with two quad-core processors and 16 GB RAM. The compute nodes are integrated into a M1000e blade chassis. Each compute node consists of two 6-core processors with 12 GB RAM and a NVIDIA Tesla M2050 GPU. Therefore we have a total of 48 CPU cores and 4 GPUs for computing.

The storage is a SAS array which consists of a Dell PowerVault MD3200 and 5 MD1200. We have a RAID 6 configuration with 72 disks each with 2TB capacity, which leads to an approximate net capacity of 120 TB. The storage is directly attached to the head node and we reach 1.4 GB/s sequential read and 850 MB/s sequential write. The compute nodes are connected with a 10 GbE application network and a 1 GbE private network.

5.2. Processing Orthoimagery Data on the HPCC

In a first experiment, a national high-resolution orthoimage data set was processed on the HPCC using the OpenWebGlobe processing software.

The data set consisted of the Swissimage orthoimage mosaic provided by Swisstopo. The Swissimage data set covers the entire country of Switzerland and has a GSD of 0.25 m in most parts of the country and of 0.5 m in mountainous areas. The data set is shown in figure 7.

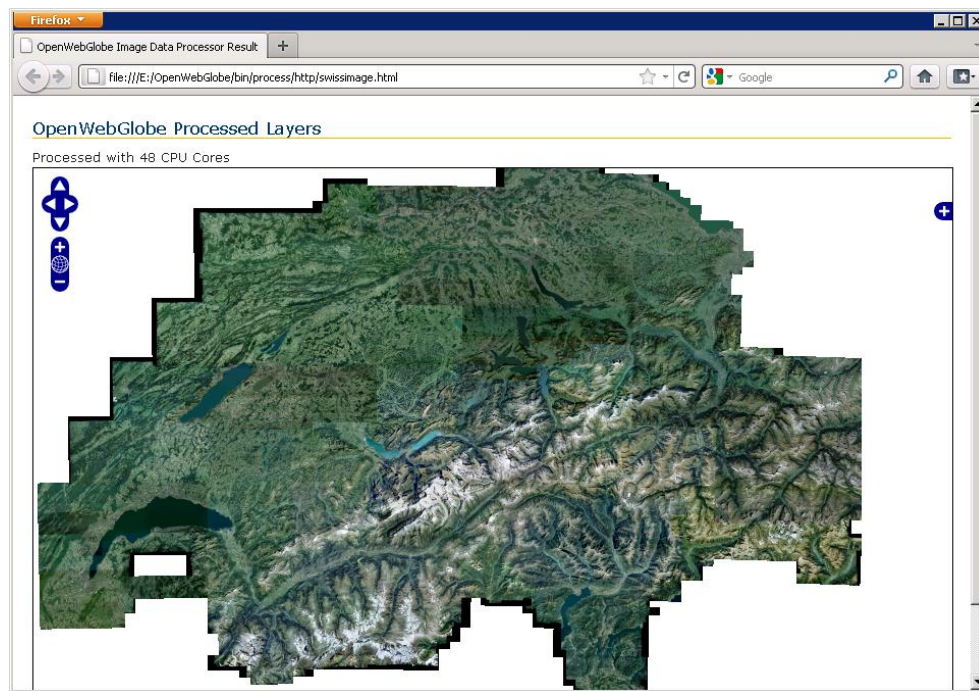


Figure 7. The Swissimage data set displayed in OpenLayers. (Geodata © swisstopo JD100033)

The raw data is around 1.2 TB (uncompressed), with 0.79 TB having a GSD of 0.25m and 0.33 TB having a GSD of 0.5m. The raw data is available in 4128 files. The dataset is stored in around 29 Million tiles. The total processing time, including resampling was around 48 hours.

Process	Cores / Nodes	Tiles/s (write)
ogAddData	48 / 4	342
ogAddData	24 / 2	230
ogAddData	12 / 1	127
ogAddData	1 / 1	14

Table 1. Generating PNG Tiles on our HPCC using different numbers of cores and nodes.

Our first results are shown in table 1. We use tile operations (reading/writing tiles) as benchmark. When using all 4 nodes and 48 cores, we reach 342 tile writes per second.

When reducing the number of compute nodes and cores, the tile operations decrease in a significant manner.

Process	Cores / Nodes	Tiles / second (write)	Tiles / second (read)
ogResample	48 / 4	107	223

Table 2. Resampling dataset

Resampling requires reading and writing large numbers of tiles. The number of tile operations (read and write) remains approximately the same as when adding the lowest level of detail. We reach 330 tile operations per second. However, resampling could be further improved by adding a better caching strategy and by preventing reading previously stored tiles again.

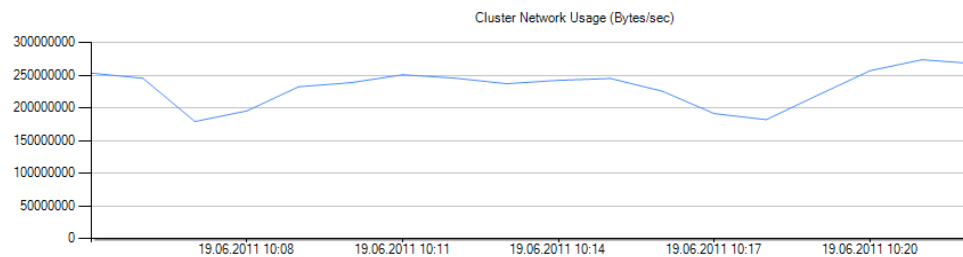


Figure 8. Cluster Network Usage when creating tiles (over 20 minutes)

Figure 8 shows the cluster network usage when creating tiles in a 20 minute window. Compressed tiles are sent over the network to the storage. The average network usage is around 215 MB/s, which is in the 2 Gigabit range.

5.3. Adding OpenStreetMap Data to the OpenWebGlobe

With the HPCC System we can generate our own OpenStreetMap tiles for customized visualizations on the globe, as shown by Widmann (2010). The data can be preprocessed as different layers and then combined again on the globe, as shown in figures 10 and 11.

The main problem of using map data and in particular OpenStreetMap data in a 3D virtual globe are the large map areas with uniform color which require some sort of shading in order to provide depth queues and to make the elevation model visible. A hillshading layer can be preprocessed and stored in different level of details, as shown in Figure 9. In figure 12 we

combine the hillshading layer with the OpenStreetMap data, this way the observer recognizes the elevation model much easier.

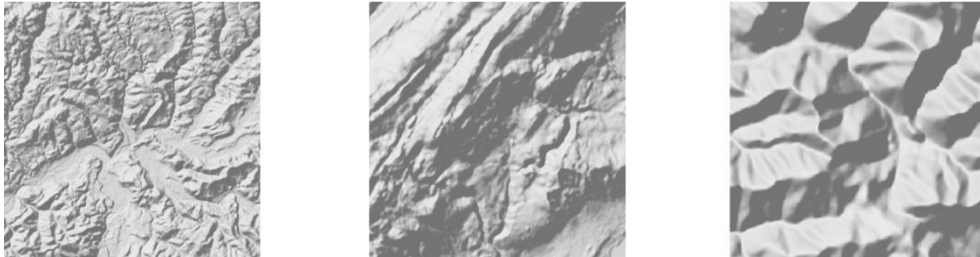


Figure 9. Different hillshading zoom levels



Figure 10. Two OpenStreetMap layers with transparency and their combination

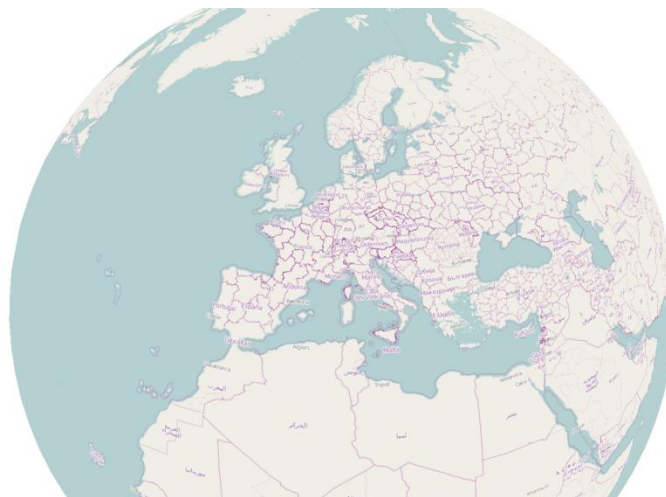


Figure 11. Displaying the OpenWebGlobe globe with OpenStreetMap data.

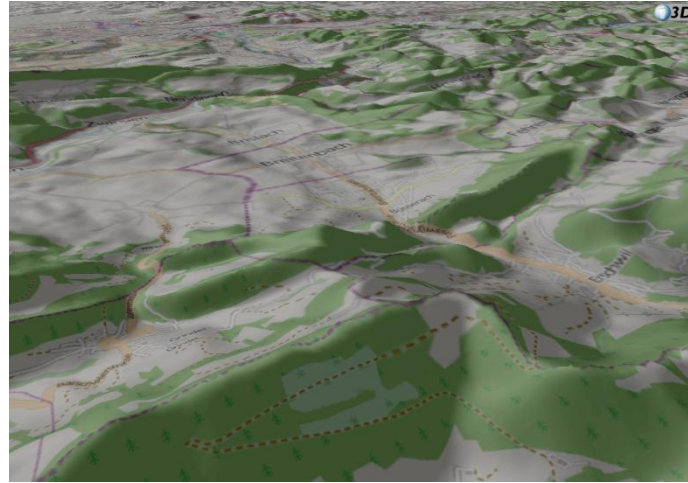


Figure 12. Combining terrain data with hillshading and OpenStreetMap tiles

6. Conclusion and Outlook

In this paper we introduced the OpenWebGlobe project and presented the OpenWebGlobe SDK for globe visualization and a toolkit for processing very large volumes of geospatial data for virtual globes in parallel and distributed computing environments. The OpenWebGlobe project was initiated to address some of the major shortcomings of the dominant commercial technologies such as Google Earth: namely, the creation of totally new or the extension of existing functionality and the creation and integration of potentially very large volumes of own and/or new geospatial contents.

With the presented OpenWebGlobe SDK it is fairly easy to create new 3D applications incorporating a powerful virtual globe – both for Web environments and for embedding in existing application frameworks, e.g. GIS. With the OpenWebGlobe processing toolkit you can integrate your own geospatial data or customize open data such as OpenStreetMap.

The presented tests and results demonstrate that OpenWebGlobe is suitable for rapidly generating and interactively exploiting high-resolution data sets of entire countries and very likely of data sets covering the entire globe. Ongoing and future work includes the exploitation of GPU processing through the support of OpenCL in the OpenWebGlobe processing toolkit or the support for very large streamable 3d city models and national topographical 3d databases.

References

- Christen, M., Nebiker, S. (2010) Neue Entwicklungen im Bereich Virtueller Globen am Beispiel der i3D-Technologie. Dreiländertagung der OVG, DGPF und SGPF. Wien
- Christen, M. and Nebiker, S. (2011). Large Scale Constraint Delaunay Triangulation. In: T. Kolbe, G. König and C. Nagel (Editors), Lecture Notes in Geoinformation & Cartography (LNGC). Lecture Notes in Geoinformation and Cartography. Springer, Heidelberg, pp. 57-72. [<http://www.springer.com/978-3-642-12669-7>]
- Eugster, H., Nebiker, S. (2008). UAV-based Augmented Monitoring - Real-time Georeferencing and Integration of Video Imagery with Virtual Globes. XXI ISPRS Congress. Beijing, China.
- Nebiker, S., Bleisch, S. and Christen, M. (2010a) Rich point clouds in virtual globes - A new paradigm in city modeling?. Computers, Environment and Urban Systems, Springer
- Nebiker, S., Bleisch, S. and Gülch, E. (2010b). Virtual Globes. GIM International 24(7), [http://www.gim-international.com/issues/articles/id1560-Virtual_Globes.html].
- Widmann, S., (2010) Aufbereitung und Nutzung offener Geodaten im Virtuellen Globus i3D, Bachelor Thesis.
- Wiesmann, S., Hurni, L. Christen M. (2011) A concept for highly interactive temporal and non-temporal 3d animation of glacier evolution scenarios on the internet. In 5th Canadian Conference on Geotechnique and Natural Hazards, 2011.

Mobile Augmented Geo Data Visualization Based on Open Street Map

Alexander Erifu¹, Harald Rieser², and Gerald Ostermayer¹

¹ University of Applied Sciences Hagenberg,
Softwarepark 12,
A-4232 Hagenberg, Austria
aerifu@gmail.com

² Salzburg Research,
Jakob Haringer Straße 5/3,
A-5020 Salzburg, Austria

Abstract. This paper describes an approach to a mobile (smart phone) augmented reality navigation system which is based on Open Street Map data. It analyzes the challenges which arise from creating such a system, such as missing topographic information, incorrect sensor data and other technological limitations. Where possible, different solutions are discussed and evaluated. It as well assesses the usefulness and accuracy of the whole approach on the basis of a prototype (on a Android powered device). The paper concludes with the insight that the accuracy, which can be achieved by such a system, is sufficient but not overall satisfactory. In certain scenarios the system might fail due to bad GPS reception or incorrect geo-data. These scenarios and sources of error are discussed separately. It as well shows that the visualization and the processing of large amounts of data are very challenging topics on a mobile platform, but yet solvable.

Keywords: augmented reality, open-gl, mobile, sensors, compass, gps, geo-data

1 Introduction

With increasingly powerful hardware and the miniaturization of sensor components, Augmented Reality[3] has become a more and more interesting topic in the field of mobile computing. Since a 3rd generation smart phone, like the Android powered devices or the Apple iPhone, does already hold the necessary sensor components and CPU power to realize such a system, a variety of software has been developed around Augmented Reality. The possibilities from the “enrichment of the reality” with further information are diverse. However the technically most challenging systems might be pedestrian navigation applications that make heavy use of hardware sensors and geo-data in order to render the virtual objects and paths in real-time over(laying) the camera view of the

phone. In the context of this paper a Augmented Reality hiking application should be realized³. As platform a G1/Dream running Android 1.6 was used. One approach to such a system was proposed by Narzt et al. in 2004, where nowadays smart phone hardware was still unavailable [1] [2]. Bringing this approach to a nowadays smart phone, the internal GPS receiver could be used to position the user, and the electronic compass and accelerometer to determine a line of vision. This means that the user is able to explore the Augmented Reality (3D-model) world by looking around freely. If now the geographic map content is rendered into the live view of the camera the resulting question is, if the virtual objects will “match” with the ones in the real world. So to say, how accurate is an Augmented Reality navigation system that solely relies on the built in sensors of the smart phone and free map data?

2 Map Data

There are several map data sources which could be used, with the most interesting ones in the context of our application being Google Maps and Open Street Map. On the first sight it might be very compelling to use Google Maps, however on closer examination Open Street Map has more than one significant advantage. As much of the content is user generated, small roads and hiking tracks only known to locals are tracked, which are not available in Google Maps. The geo-data is also accurate and provides particularly in rural areas excellent granularity.

In addition, the category meta data of the Open Street Map is especially interesting. Paths can be classified as for example highways, forest roads or hiking routes. Even the SAC scale is stored, which is used to measure how demanding a hiking route is.⁴ Another major advantage is, that more points of interest are available in Open Street Maps than in Google Maps and they are categorized as well. On using these categories it is fairly easy to filter a specific point of interest group.

In Figure 1 two screenshots of the same mountain region⁵ in the Austrian Alps are depicted. The left one is taken from Google Maps and the right one from Open Street Map. It can be clearly seen that the Open Street Map contains more features, cable cars and hiking routes, which are very interesting for the application as hiking is the intended use.

What however completely disqualifies Google Maps is, that it does not provide access to the geo-data. This means that only an image of the map is available but not the geo-data itself. Open Street Map whereas allows to extract all the raw geo-objects. As this data is needed to generate the 3D models, Google Maps

³ Based on the Salzburg Research Peak AR Software.

⁴ OSM Map Features: http://wiki.openstreetmap.org/wiki/map_features

⁵ The Spielberg mountain at +47° 43' 19.04", +13° 13' 59.84"

is therefore not an option.

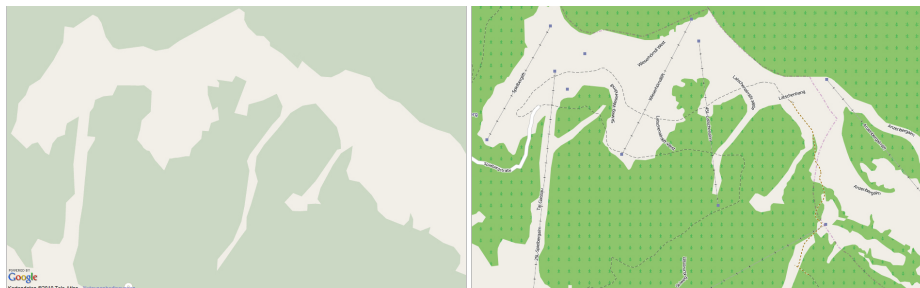


Fig. 1: Google Maps and Open Street Map (with Mapnik renderer) screenshot of the Spielberg mountain in the Austrian Alps.

There however is a problem when using Open Street Map, and that is the missing topographic data. To further examine the problem of the missing height, a closer look at the Open Street Map data model is necessary. The object data hierarchy consists of three main types: Nodes, Paths and Relations. The structure is strictly hierarchical, which means that relations consist of paths which again consist of nodes. Each and every of these elements can hold metadata and has a key-value pair for their identification. A node is frankly speaking a geographic point on the surface of the earth, specified by its latitude and longitude. A path consists of a list of nodes which the software has to combine.

As mentioned above, latitude and longitude are available but the height is missing. This gets more obvious when looking at the document type definition (DTD) of the nodes. The data model simply does not require a height. However, there is an optional elevation tag which is sometimes used for mountains and similar objects where the height is of overt importance. This problem may arise from the fact that it is technically more sophisticated to determine the height correctly via GPS. This is a problem for the implementation of the prototype because elevated points and paths would be displayed “flat” on the ground. This might not only look confusing but will make it hard to associate the virtual information with the real world since the objects will not overlap. This issue is definitely bigger in mountainous regions, but since the software is intended for hiking it can be assumed that these regions are of major interest. As a consequence from this situation, an alternative source for the topographic data had to be found.

3 Topographic Web Services

As source for the missing OSM height information, various topographic web services could be used. As the map data is requested online, it would be logical

to query the height of the geo-objects online as well. The 3D models could then be constructed with the combined data. Acquiring topographic information and creating a digital elevation model of the world has been an ongoing task for several years. There are three major projects which are able to provide almost gap less and accurate topographic information.

1. Shuttle Radar Topology Mission⁶ (SRTM).
2. GTOPO30⁷ (Global Topographical data set available at 30").
3. Adv. Spaceborne Thermal Emission and Reflection Radiometer⁸ (ASTER).

The data products from these projects can be accessed via web services at the geonames⁹ website. They take as parameters the latitude and longitude and return the height in meters. To give a better overview which facts lie behind the cryptical abbreviations of these projects, the key facts are outlined in Table 1.

	ASTER	SRTM3	GTOPO30
Data source	ASTER	SRTM	From organizations around the world that have DEM data.
Generation and distribution	METI/NASA	NASA/USGS	USGS
Release year	2009	2003	1996
Data acquisition period	2000 ongoing	11 days (in 2000)	unclear
Posting interval	30m	90m (3 arch seconds)	1000m (30 arch seconds)
DEM accuracy	7–14m	10m	30m
DEM coverage	83 degrees north – 83 degrees south	60 degrees north – 56 degrees south	Global
Area of missing data	Areas with no ASTER data due to constant cloud cover (supplied by other DEM)	Topographically steep area (due to radar characteristics)	None

Table 1: Comparison between different Digital Elevation Models.¹⁰

⁶ <http://www2.jpl.nasa.gov/srtm/>

⁷ <http://www1.gsi.go.jp/geowww/globalmap-gsi/gtopo30/gtopo30.html>

⁸ http://terra.nasa.gov/About/ASTER/about_aster.html

⁹ <http://www.geonames.org/export/web-services.html>

¹⁰ Source: <http://www.ersdac.or.jp/GDEM/E/2.html>

According to this information, the ASTER project provides data with the smallest raster (30 meters) and good accuracy (7 to 14 meters). To confirm these values a comparative metering was conducted using a Garmin GPSmap 60CS GPS receiver, which holds a very accurate barometric sensor¹¹.

The grade of the raster (posting interval) which is used by the digital elevation models, is very important. It has to be narrow enough to generate a usable topography. As a consequence the measurements took place on an open field (without any tree cover) in an hilly environment with the height slightly changing and the points close to each other (12 meters).

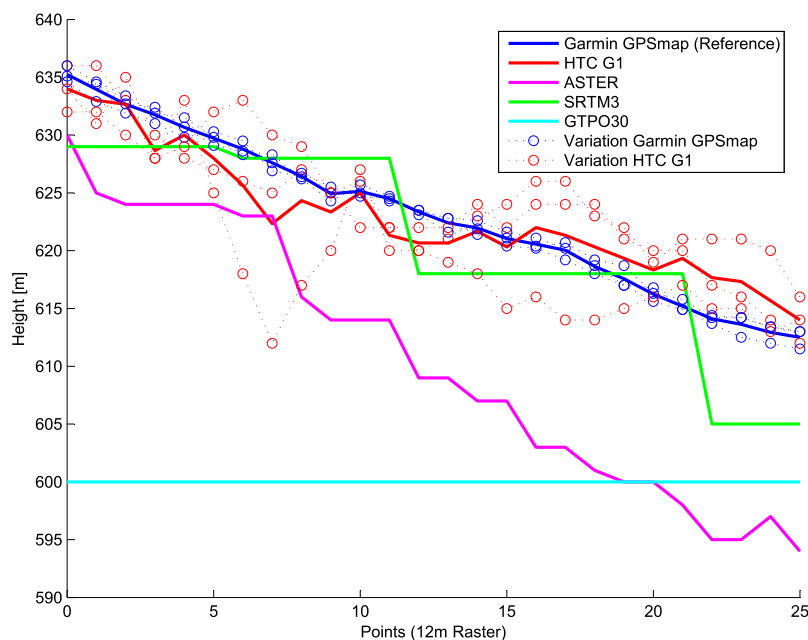


Fig. 2: Accuracy comparison of the height information between Garmin GPSmap 60SC, HTC G1 and Digital Elevation Models.

Analyzing the findings from the measurements, it can be seen in Figure 2 that the GTPO30 values do not change at all due to the big raster of 1000 meters and are therefore useless. The SRTM3 data has a bigger raster than ASTER, but the most accurate values. The ASTER data models the topology quite well but with a rather big deviation that seems like a constant offset. A deviation of five meters height between the user and the objects can be a big problem if

¹¹ The barometric sensor was calibrated before measuring.

the user is close to (or directly on) a geo-object. This would result in objects (especially paths) displayed above or too far under the camera.

Another problem with acquiring the height has not been mentioned yet: The GPS height of the smartphone itself. Receiving the height via GPS is quite inaccurate, and even more problematic on highly integrated devices like smartphones. A detailed evaluation of the smartphone GPS accuracy is carried out in Figure 2.

An alternative to bypass the erroneous GPS height would be to work with the relative topography from the digital elevation model. This means simply requesting the user's height from the web service as well. The absolute error from the DEM model would then lose importance since the user will be placed relative to it. This works fine for the points of interest but is not a fully satisfactory solution for the paths. A problem occurs if the user moves along the path in between two OSM nodes. On constructing the 3D models only the height for the nodes is queried, not for the whole path that will connect them. If now the height from the web service is incorrect for the space in between the nodes, it would have the effect that the user could move into, over or under the path.

Another possible solution is based on the assumption that the user is always placed on the (real) ground. If the user follows a path, his height could be set to the height of this path. If the user is in between two Open Street Map Nodes (of the path), the height has to be interpolated and assigned to the user. All the other surrounding objects would then be at correct height too, as their height is relative to the path and thus to the user. If the user moves farther away from the path, it could be switched back to only using the height from the web service again.

The drawbacks of these methods are, that they will need more processing time than the approach based on the GPS height. Not only CPU-time, but also time to permanently request data from the web service. Moreover, the user will need a permanent data connection to acquire the web service data. The ideal solution would be to construct a 3D mesh representing the topography of the whole area. All the geo-objects and the user could then be placed on it. Unfortunately, this is not possible yet due to the restricted memory and computing power of nowadays¹² smartphones.

Summarizing the discussion, it can be said that the issue is rather severe. The shown strategies do either not cover all of the possibilities, are very complicated and error prone or not realizable. As conclusion, the best approach for solving this problem has to be found by evaluating the prototype.

¹² At the time of writing.

4 OSM Import

Since the phone has a permanent network connection it can request the data directly from the Open Street Map. The problem with the OSM-API is, that it will simply return everything in the bounding box, and is not capable of providing categorized information. This means that the amount of data the application would have to deal with is rather big, which is not very beneficial on a device with limited CPU power, memory and bandwidth. It would have to filter and sort all the data, while most of the objects are discarded anyway since they are not important.

For this reason, a dedicated server (OSM Interaction Server) is interposed between the application and the OSM-API, which takes care of this task. The application can now request the bounding box with a specified category from the OSM Interaction Server. The server will then request the bounding box from the OSM-API and remove all the irrelevant data, only returning the specified category. The application now parses this data using a JSON parser, and stores it into a local SQLite database. This approach helps to reduce the amount of data the application has to process, since only the requested information will be stored in the database.

4.1 Height Web Service

Since the imported OSM geo-data does mostly not contain a height information; it has to be acquired from another source. As already discussed in Section 3, the digital elevation models (DEM) provided by the web services at the geonames website are used. Using height information from OSM can be a problem as it is not clear in which manner the information is stored. OSM uses the `<ele>` tag for defining the height of nodes, which is defined in the OSM wiki as:

*Elevation (height above sea level) of a point in meters. This is mainly intended for mountain peaks but could also be used for elevation of airport runways and many other objects. For Open Street Map, please use the elevation above sea level defined by the World Geodetic System, revision WGS84.*¹³

This definition is a bit confusing, because GPS (which uses WGS84) can only measure the height above the ellipsoid and hand held devices have to use undulation tables to determine the height above the mean sea level. Besides that, “mean sea level” is a vague definition as many countries have different zero levels, which differ slightly. Furthermore, if a barometric sensor is used for recording the height, the question again is, to which level it is calibrated.

Summarizing the discussion, it can be said that the whole situation is rather complex and unexperienced OSM users could, at worst, confuse the ellipsoidal height with the geoid height as it might not be clear to them how their GPS

¹³ OSM wiki for ele tag at: <http://wiki.openstreetmap.org/wiki/Key:ele>

receiver works. This would be much more dramatic than the differences between country specific mean sea levels since this difference can sum up to 50¹⁴ meters and the difference between the country specific zero level is only at some meters. Currently there are efforts to introduce a new `<alt>` tag in OSM for the height over the ellipsoid, which would be a better reference. The question is therefore, if it is wise to use the OSM height at all, since it is only available for very few objects and fraught with uncertainty.

Using a web service for the height of the geo-objects and the user, holds a notable benefit. The reference for the whole 3D scene is the same, which means that the whole height data is always consistent. The values do have inaccuracies (see Section 3) but a consistent height model is simply more important for the function of the application than the deviation. The height can be requested from the geonames¹⁵ web service via a HTTP GET request where latitude and longitude are passed as parameters. Since both interesting height web services (ASTER and SRTM3) follow the same specification for the request, the type of DEM data which should be used can be selected in the application. Using both sources to calculate mean values would also be possible. On the one hand this would result in a more accurate topography but on the other need twice the processing time to build the model.

A web service request takes approximately 200ms¹⁶ and geonames allows to define a list of 20 points per request¹⁷. This means that it can take some notable time to construct the geo-object database. Especially long paths are problematic due to the fact that they consist of a large number of points. An improvement could be made to the system by integrating the DEM model directly into the Open Street Map Interaction Server. In doing so the web service could be bypassed and the server could provide the height directly along with the requested geo-data.

5 Geo-Data Filter

Filtering the data is an important issue as we do not need all the data provided by Open Street Maps. For the intended field of application, a hiking AR browser, many objects can be ignored as they are simply not important to the user. The filtering is done by the OSM Interaction Server according to the category which is selected by the user over the application's user interface. The user can choose a specific category like mountain huts or hiking paths which will then be requested from the OSM Interaction Server. The server as well filters unusable objects such as points of interest without name or category. In addition, a line of sight (filter) can be defined with a slider on the user interface, to avoid cluttering the display

¹⁴ Depending on the position of the user.

¹⁵ Geonames website at: www.geonames.org

¹⁶ Mobile network 3G connection: 1.6Mbit down/0.6 MBit up

¹⁷ On the geonames free server.

with too much information. This filtering takes place on the device and will only affect which 3D objects are visualized.

6 3D Models

This functional block can be described as one of the centerpieces of the application since here the 3D models are constructed from the requested geo-data. However this part should not be confused with the OpenGL renderer, as the models alone are just static vertex data. The model creator holds different stages of construction and different construction facilities for each object category. First the objects are categorized into three categories which are points, paths and areas. Each category has a custom model builder since it has specific rules for its construction. When the AR-objects are instanced the appropriate builder will be called, which transforms the geo-data into vertex models and attaches further information to the object. The additional components, such as labels, are generated from the metadata of the object which is stored in the SQLite database.

6.1 Points

Points are the most basic data type in the AR hierarchy and can have three different appearances:

1. A basic point, belonging to a path or area.
2. A point holding a label, belonging to a path or area.
3. A point of interest.

A point can hold different data, depending on these three types. A basic point only has a position with X, Y and Z-coordinates, it even does not know which path or area it belongs to (since it can belong to multiple objects). If the renderer demands a text label at a point owned by another object it simply attaches it to a basic point. The render can as well remove this label again if it is needed elsewhere.

The third type of point is the most complex one and discussed in the next Section.

6.2 Points of Interest

Since points of interest (POI) are points by definition, there is not much of a model to create. Nevertheless there is some necessary additional information that needs to be attached to them. Points of interest are visualized as pictogram (icons) shown in Figure 3 which represent the category of the POI. They as well have a text label above the icon stating the name (see Section 6.4). Since the category of the POI is known, the appropriate icon image can be selected and attached to a 2D square as a texture. The size of this square is 10 by 10 “virtual” meters. The virtual size of the texture is not of great importance, since it will

be scaled to the correct size in relation to the distance. The process of scaling is described in Section 7.3 and does not alter the model but only the visualization.

The OpenGL vertex model (a 2D square) of the POI can be directly created from the points UTM position and stored in the instance of the point along with the attached texture (icon image). A problem that emerges from using this way to construct the POIs is, that they are actually 2D objects since a square with a attached texture does not have a depth. This problem as well applies for text labels (see Section 6.4) and is solved by rotating the texture 90° to the user.



Fig. 3: Examples for used points of interest pictogram representations.

6.3 Paths

A path is defined by a list of basic points that form a line and meta data stating name and category. Rendering paths as a line is adequate for distant paths but will not be very suitable for close paths, since a line has no width it does not have a perspective. This will make it very hard for the user to associate a street in the real word (which has a perspective) with the virtual path. A line representation of a path can be seen in Figure 4a. To create a perspective the path must have at least a width. However the line model is not completely unimportant, as it will be shown later.

A 3D path consists of single connected boxes, where each one is constructed by adding a width and height to the path. The width derives from the category (type) of way. Footpaths for instance are constructed with a smaller width than main streets to convey the difference to the user. Only using the width for constructing a path might be sufficient, if the user is very close to it. If the path is viewed sideways from a distance, it would be displayed as a very thin line. In Figure 4b a path is depicted that has a width but no height. It can be seen that in the distance, the path becomes very thin and pixels even start to disappear. This results in a flickering broken line because some of the pixels toggle between visible and invisible.

To prevent this behavior a height is added to the path, which is always constant throughout all the path models. Note that this height has nothing to do with the altitude (geo) height of the path. In Figure 4c a path is shown that has a width and height but is drawn in a single color. The 3D impression is slightly better, but still not very strong. This results from the fact that there is no lighting in

the OpenGL scene, so the edges of the path are not visible. To intensify the 3D impression, the top of the path is drawn in a lighter color. Such a colored path can be seen in Figure 4d. In the implementation of the prototype, it can be selected which of the path-types shown in Figure 4 should be used for the visualization.

The path is constructed with triangle strips using a single box for every two nodes, where the next node is always the previous last one. The boxes are then connected to each other by filling the space in between them to prevent having gaps. This means that (in the worst case) the angle between two segments (boxes) is 90° resulting in a connection edge of 45° . On doing so, the sharp bend on the opposite corner of the two boxes will be hidden inside the tunnel, where it is not visible from the outside. The bend can never move outside of the tunnel, as the width is always constant for the whole path. Using this way to construct the 3D path is very effective, since only one model is needed which can be created entirely in one loop. Another issue is the perspective, since paths

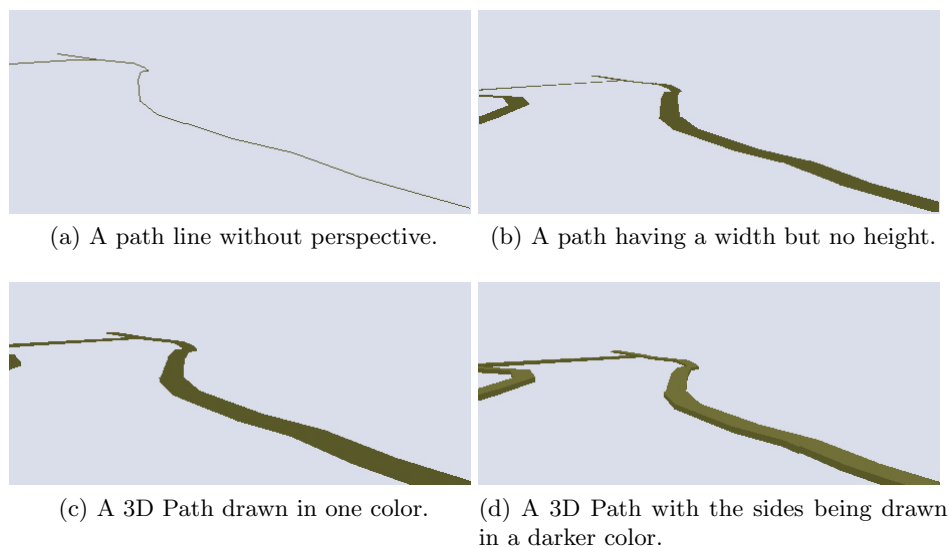


Fig. 4: Different Path models and visualizations.

will start to clip and become invisible past the OpenGL line of sight. This line of sight is defined, among other things, by the focal length of the phone's camera and cannot be changed (see Section 7). This behavior is quite problematic since distant paths would not be visible which would foil the whole use of the application. Fortunately this can be solved by using an OpenGL pinch, which is integrating a line into the path model. As mentioned the perspective does not

apply for OpenGL lines (GL_LINE_STRIP) since they are not really 3D objects. This means that they always will be rendered with the given line width, no matter how distant they are. The OpenGL line of sight, where the 3D path starts to disappear in the distance, is hard to determine and because drawing a line is not really demanding in terms of performance, the whole path can be drawn as line overlay into the 3D path. This also solves another issue, where the 3D path will start to flicker in the distance as it starts to disappear and its size toggles between zero and one pixel. Summarizing this, the whole path consists from three different models which can be combined.

For a street usually has a name it needs a text label as well (see Section 6.4 for labels). The major difference to the points of interest is the placement of the labels. The POI label can simply be displayed above the icon, the placement for paths however is a bit more sophisticated. One question is where on the path to place the label since a path has a number of points and moreover the user can change his position on the path.

To solve this problem the paths own render method determines the point of the path which is in a good distance to the user and places the label there. If the user changes his position the label has to be moved as well. To assure that the label does not cover the path, which would be counterproductive in terms of readability, it is displayed above the path. To help the user to associate the label with the appropriate path, a graphical arrow, again constructed from lines, points down from the label to the path. In Figure 5 a screenshot of a path with two text labels is shown.

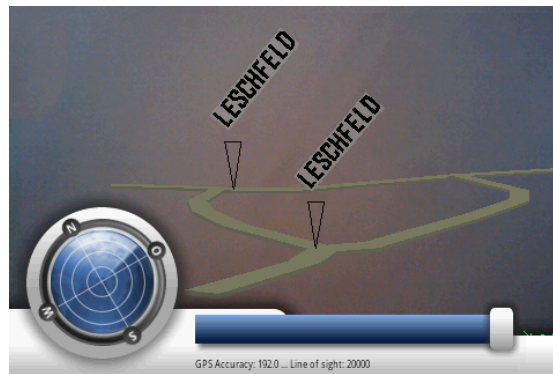


Fig. 5: A path with two labels indicating the name of the street. The path is viewed from a higher position and consists of two segments. The label of each segment is always placed at the node closest to the user.

6.4 Labels

A label is a box with fixed height and variable length holding the rectangles for the characters of the text. For this approach, a graphical representation of the (used) alphabet is needed. So to say an image with all the letters from A to Z and other frequently used characters¹⁸ in names of streets and places. For the sake of simplicity and readability all the characters will be converted to capitals. The characters in the multi texture file, shown in Figure 6, have to be symmetric to each other's middle position and the total length dividable without remainder through the number of characters supported. This is due to the fact that OpenGL defines the single textures in the multi texture file by using relations. The labelizer algorithm then determines the length of the given text and creates fixed size rectangles for each and every character. Finally, the algorithm will "paste" the required character from the multi texture file onto the appropriate rectangle.

Once the label is constructed, it can be attached to a point which will then know



Fig. 6: The layout of a multi texture file used for constructing AR label text.

to which category of points (see Section 6.1) it belongs to and how it has to draw itself. Using textures for labels moreover holds the benefit that the multi texture file (and thus the supported character set) can be changed very quickly and that they are the preferred method in terms of performance. To prevent overlapping, the text labels for the paths will be displayed in a 50° angle. Since the contrast is very important for readability, as stated in [4], the chosen solution is to use black for the color of the text and a white background layer with 60% opacity. This ensures that the contrast is always adequate, yet the text background is still firmly transparent.

7 Field of View Representation

This function block is responsible for the user's virtual representation. It holds the OpenGL camera, the values from the hardware sensors, the GPS position with the height and the line of sight defined by the user.

7.1 OpenGL Camera

The OpenGL camera is very important since it defines the aperture angle of the virtual camera, which has to match exactly with the one from the camera of the

¹⁸ Such as German umlauts in this case.

phone. On the HTC G1¹⁹ the (vertical) aperture angle is at 36° . The OpenGL camera can be set by using the `gluLookAt()` and `gluPerspective()` methods. An visual explanation of the cameras parameter is depicted in Figure 7. These two methods can be seen as a unit since both are needed to define the field of view.

The `gluPerspective` method takes as `fovy` the field of view in degree along the Y-axis, the `aspect` ratio that defines the ratio of width to height as well as `zNear` and `zFar` which define the distance from the user from where the near/far clipping will be applied.

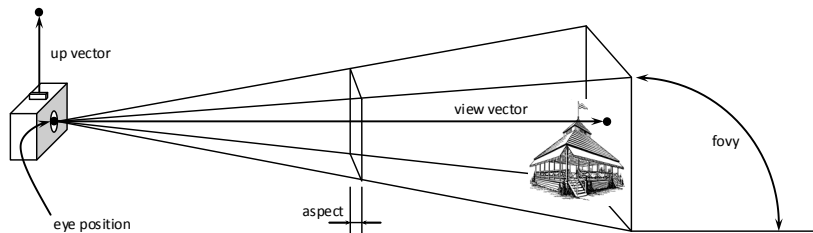


Fig. 7: The OpenGL camera defining the field of view, view- and up-vector.

For the implementation the following parameters were chosen: For the camera definition the `upX` parameter is -1 and the `upY` parameter is 1. This means that the camera is looking into the phone along the -Z-axis just like the real camera. These parameters are static and do not need to be changed, since it is easier to rotate the whole scene around the camera, than the camera itself. For the perspective the important parameters are the already mentioned aperture angle of 36° used for the `fovy`, and the aspect ratio (`aspect`), which can be determined by dividing the display width by the height. Less important are the `zNear` and `zFar` distance, which can be set to a low value for near and to a high value for far because we do not want OpenGL to clip any objects. This will be done by the renderer according to the user interface defined line of sight.

Summarizing the camera set-up, it can be said that the camera's definition is a quite static. The aperture angle is of significant importance because if this angle does not match up with the one from the phone's camera, the whole scene will be clinched or stretched and the virtual objects will not match with their real world positions. This moreover holds a significant drawback, the whole im-

¹⁹ It was not possible to find any documents stating the angle, so the focal length was measured and the angle calculated.

plementation of the OpenGL camera becomes device specific²⁰ since most of the phones have different camera lenses.

7.2 Scene manipulations

Drawing the objects starts with rotating and translating the whole scene according to the values in the field of view representation. These parameters apply to all of the objects so the transformations can be applied to the whole scene. Since the OpenGL camera cannot be rotated by an angle (see Section 7.1), it is preferable to translate the whole scene, in such a way that the user is placed at the OpenGL origin of coordinates. In other words the whole scene is translated by the users UTM position.

Now the whole scene can be rotated around the user by using `glRotatef()`. The method uses as parameters the angle in degrees and which axis to rotate around. The Y-axis rotation is therefore the compass angle and the X-axis rotation the G-force sensor. Now the specific draw implementations for all the objects throughout all AR-layers are called, which will draw the 3D models described in Section 6.

7.3 Perspective Scaling

A problem when using OpenGL arises from the camera's perspective. It causes closer objects to be displayed excessively large while they will quickly disappear in the distance (in relation to the aperture angle). This problem does not apply for paths any more, since their 3D models were created to prevent this behavior as described in Section 6.3. Points of interest and text labels however still suffer from this problem.

Another issue which even worsens the situation is that, because they are textures, they do not scale very well (as they consist of pixel images). Especially text will become unreadable very soon if reducing the size. On the counterpart the perspective is important for the user's perception of the scene. A scene with fixed size POIs would seem unrealistic and the user might not be able to tell off-hand which objects are closer to him.

The chosen solution for this problem is to scale the text labels to a fixed size as readability is the primary interest here, and apply a custom perspective graph to the POI icons if they are in distances under 100 and over 700 meters. For the range in between, the scaling will happen according to the distance. The two threshold points therefore define the steepness of the graph. This custom scaling prevents the POI icons from growing too big or disappearing, while the perspective impression will still be preserved. Adding transparency to the POI can furthermore help the user to orientate.

²⁰ In Android 2.2 a new method `getHorizontalViewAngle()` may be introduced which would be of great service towards building a more generic application since the angle does not have to be measured any more.

The size of the labels in contrast should be constant at every distance. This means that the scale factor has to be the exact counterpart of the perspective, enlarging the label while the perspective reduces it's size. The scale factor can therefore be directly derived from the distance between a (geo) point and the camera.

This way the virtual content is presented to the user in a clear, unambiguous manner, while still keeping the 3D impression of the whole scene.

7.4 Texture Rotation

Since a texture is 2D it will appear to the user as a thin line if viewed from the side. Certainly a box could be used where the texture is placed on all four sides. However this approach is not favorable since the text will be hard to read if the user is facing the edge of the box.

The chosen solution is to leave the texture on a 2D rectangular shape and rotate the shape to an angle of 90° to the users view vector. This is realized by determining the view vector between the user and the texture, which can be calculated from both positions.

This way the user will always have a perfectly visible texture. Moreover it will result in the labels and textures slightly "bending" towards the view vector which is in the middle of the screen. In Figure 5 it can be seen that the text is not entirely parallel but slightly inclines towards the vertical middle of the screen. This alignment helps to create a more appealing visualization of the scene.

8 Prototype Introduction

A set of screenshots of the application is shown in Figure 8. The compass in the lower left corner provides an overview to the user, but is not further important for the Augmented Reality view. The line of sight can be adjusted with the slider on the bottom and reduces the amount of displayed elements in a certain radius around the user. The different layers of information like mountain peaks, hotels and hiking paths, can be selected from the user interface.

The screenshots²¹ in Figure 8 show different augmented situations. In Figure 8a a distant mountain with a path winding up to the top is depicted. As the objects are quite far away the perspective scaling algorithm uses the smallest available scaling factor for the mountain peak icon. The path is displayed as a 2D line since it is too distant to be displayed as a 3D path. (The reasons for this are explained in Section Section 6.3.)

Figure 8b shows a point of interest, which is scaled to its maximal allowed size, since the shop (the yellow house on the left) is only several meters away.

²¹ As there is no possibility to take screenshots on a device without root permissions, a laptop computer with DDMS has to be used which negatively affects the quality of the pictures.

In Figure 8c and 8d the user is situated in an urban setting. In Figure 8c the crossroad called “Radlerstrasse” is visible in the AR-view but not yet in the real world because it is hidden by the bend of the street and the hedgerow. In Figure 8d the user has moved around the bend and reached the crossroad.

Figure 8e shows the user is in a more rural area, where the path is winding over and around the hilly countryside.

9 Findings

9.1 Introduction

The prototype was realized to examine the feasibility of a mobile Augmented Reality navigation system, which uses free map data. The focus hereby was to determine which problems especially arise from using a smartphone as platform. The findings from the prototype can be divided into a range of different subjects. Most interesting hereby is the accuracy, since it is one of the key criteria of the system.

9.2 Geo Data

Using digital elevation models for the topographic information has proven as a good solution to compensate the missing height in Open Street Map and the highly inaccurate GPS height from the smartphone’s GPS sensor.

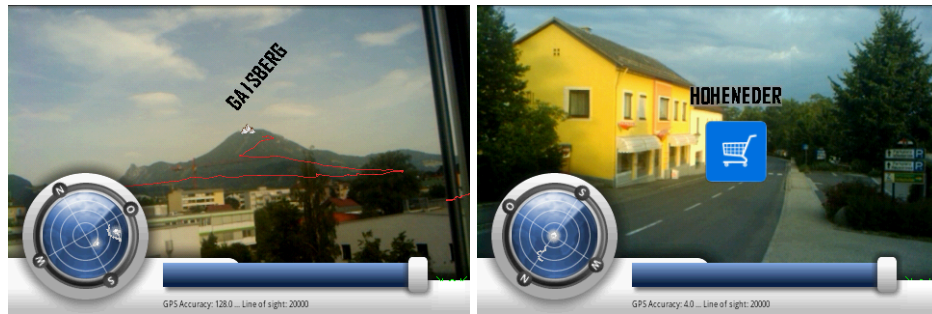
The biggest drawback at this is the relatively long latency time of the web service. Building a data model with 250 OSM nodes takes around 60 seconds. Therefore it would be beneficial to integrate the web service request into the OSM interaction server.

For the user height however the latency is not a problem as only one position at a time will be requested. This does however mean that the user needs a data connection all the time.

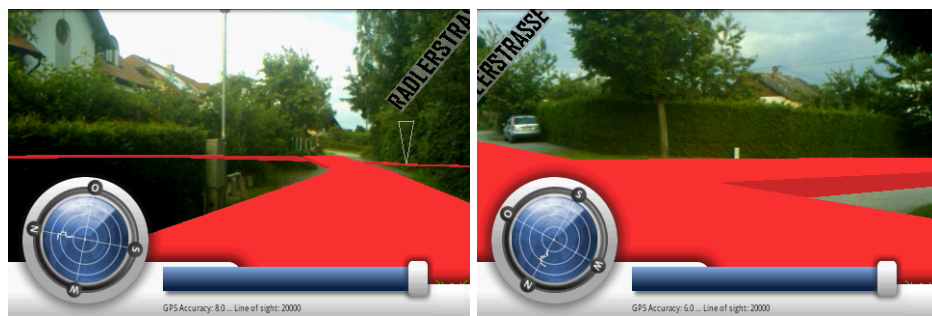
Open Steep Map has, aside from the missing topographic information, proven as an accurate source for the geo-data. A problem which emerged upon testing the prototype was that especially in an urban environment with many crossroads, paths can consist from very short segments. Every segment contains the name of the street so it will get a label stating this name. Although correct from an conceptual point of view, a myriad of labels could confuse the user. Using the line of sight slider will not help here, as the path will disappear along with the labels.

9.3 Visualization

When looking at the screenshots of the application, it might be tempting to use transparent paths so the user could see the real path beneath it. While a transparent path might look more appealing, it should not be underestimated that it



(a) A distant mountain with a path (red) winding up to the top. The peak has its own icon and a label indicating the name. (b) A very close point of interest which cannot get scaled any bigger by the perspective since the scaling-algorithm prohibits it.



(c) A street in an urban area with a cross-road shortly ahead. (d) Further down the street at this crossing.



(e) A rural path leading through the countryside.

Fig. 8: Screenshots from the prototype in different situations.

will not be as clearly visible as a solid one. Upon testing, it became clear that a transparent path with 50% opacity is hardly visible on the phone's display in the bright sunlight.

The perspective is a very important part of the visualization. While it is absolutely necessary for matching the real and the virtual world, it has to be manipulated in such ways that very distant objects do not disappear and close objects do not grow too big.

The paths will not get any smaller than one pixel in width and the points of interest will not scale below the predefined size. This perspective scaling, described in Section 7.3, is vital for a good user experience.

For the visualization of the points of interest the pictograms have proven as a good way to display this information, since they are easy to identify from every distance due to their plain appearance.

It can be seen in Figure 8 that the text labels do not change their size with the perspective because the scaling algorithm was set to a constant scaling factor. This results from the fact that text does not scale very well and gets unreadable very rapidly. For the user's perception of the scene, the geo objects such as paths and points of interest will scale with the perspective and the text can be seen as a separate layer which does not have to follow these rules.

A better approach would be to generate a 3D model from the whole surrounding topography (DEM) on which all the objects, including the user, could be placed on. The user would only be allowed to move on this grid and thus could never be positioned under or over it. Furthermore, using a web service for his height on the device would then also become obsolete. Unfortunately this approach will need better hardware on the smartphone which was at the time of writing not available.

9.4 Accuracy

The accuracy is one of the key requirements for the application, because it's necessary for providing an convincing augmented image. The closer the user is located to an object be more important the accuracy of the GPS gets. An example for such a situation would be if the user is standing on the left shoulder of a road and moves on to the right shoulder. If now the GPS does not recognize this movement the system would display the path on the wrong side of the user. Regarding the effects of accuracy, three different visualization situations can be identified. The first one is to visualize points of interest, no matter at which distance. The second one is distant paths and the third and most delicate one, close paths. While the first and second scenario are not a problem at all and work very well, the third one can, under certain circumstances, have heavy inaccuracies. As already mentioned, the accuracy of the system depends on many factors. If however, all these deviations add up to a worst case scenario the visualization will fail.

9.5 Conclusion & Outlook

The prototype has shown that the Augmented Reality system proposed in this paper is feasible. The advantage of the introduced prototype towards other 2D AR-browsers is, that it is able to visualize paths and use the correct perspective for the whole visualization. Furthermore, it combines Open Street Map data with digital elevation models and OpenGL visualization. The software allows the user to freely move and browse through the virtual paths and points of interest.

Using the Open Street Map data along with digital elevation models has proven as an adequate solution for acquiring the geo-data. The limitations of the technologies do affect the system but could mostly be solved. However the visualization is not completely accurate in certain situations due to the restrictions of the used technologies. Better geo-data and better hardware sensors would improve the whole system greatly.

Other improvements could be made to the system by transferring the whole 3D model creation process to an external back-end server. This means that the phone application only sends it's GPS position and requested data to the server. This server then acquires, filters and sorts the geo-data. It could furthermore already construct the 3D vertex arrays and send them to the phone where they will be converted into vertex buffers. A model containing the whole topography would as well be advantageous as with it the problems in the visualization could be solved. More accurate DEM models or using geo-data sources which already include a height would be particularly advantageous.

Acknowledgements

This work has been funded by Salzburg Research.

References

1. Wolfgang Narzt, Gustav Pomberger, Alois Ferscha: A New Visualization Concept for Navigation Systems, Springer Verlag Berlin Heidelberg, 440–451, (2004).
2. Wolfgang Narzt, Gustav Pomberger, Alois Ferscha: Augmented Reality Navigation Systems, Springer Verlag Berlin Heidelberg, 177–187, (2004).
3. Ronald Azuma: A Survey of Augmented Reality, Teleoperators and Virtual Environments 6:4, 355–385, (1997).
4. Joseph Gabbard, Edward Swan, Deborah Hix, Active Text Drawing Styles for Outdoor Augmented Reality: A User-Based Study and Design Implications, IEEE Virtual Reality Conference 2007, Charlotte, North Carolina, USA, 35–42, (2007).

Modeling Railways in OSM: from 2D to 1D

Berend Schotanus

First Flamingo Enterprise B.V.
Utrecht, The Netherlands

June 2011

Abstract. This paper discusses a practical method to extract network level information for railways from OpenStreetMap data. It introduces the ‘heartline’ concept that represents the geographically accurate position of linear infrastructure without further detail. And it reports about the experiences of the author to model the Dutch railway network with such heartlines.

Keywords: Railway, Heartline, Network, Extracting data

1. Introduction

As a professional in the railway industry I have often looked with some contempt at the pace of innovation in computer business. When I started my own company one of my goals was to understand at least some of the new possibilities of information technology and make a translation back to my old profession.

I have been working on these subjects:

- **Transport economy**, the optimization of which services to run in order to get an attractive offer for passengers and an economic match between supply and demand.
- **Passenger information**, the art of explaining those complicated services to passengers so they can comfortably use them.
- **Traffic management**, steering switches, signals and vehicles in such a way that the promised services are actually performed.

If you want to improve your business you’ll have to improve your tools. It’s not only about software, data is equally important. In my search I came across OpenStreetMap, that would be a terrific data source for railway ap-

plications. Geographical information is the basis of everything else when you are running a transport system.

In this paper I want to share some of my reconnaissance of getting OSM data useful for my purposes. The result is a practical method for modeling a railway line, that hopefully will be a basis for further development.

2. Background

2.1. Reference systems

OpenStreetMap uses the Cartesian coordinate system with latitude and longitude in degrees. It covers the whole globe and can be amazingly accurate, so for building a worldwide digital map this would probably be the right choice.

But Cartesian coordinates are not the only way to indicate a position on earth. The most common way is to name country, city, street and house number. Obviously there is a need for translating between the 'coordinates' reference system and the 'name' reference system, leading to the well known (reverse) geocoding.

In fact it, what's the most practical reference system, depends upon your purpose. For most linear infrastructure it is logical to use a one-dimensional distance measurement along the line, resulting in a kilometrage or mileage (that's what milestones are for). So you would be able to get a '1D' representation like (*route="nl.os05" km="32.853"*) that refers to the same position as a '2D' representation like (*lat="52.02822" lon="5.53104"*).

In railway business this kind of '1D' representation is omnipresent, while accurate '2D' representations are relatively rare. By nature the length of a train is much larger than its width and this is even more true for the tracks. To get a practical representation, for instance of a track diagram, you must exaggerate its width and suppress its length. So what you would mostly do



Figure 1. That's what milestones are for...
(Source: Jujuinconnu, fr.wikipedia.org)

is use a schematic representation and add a track kilometrage to tie it to reality (figure 2).

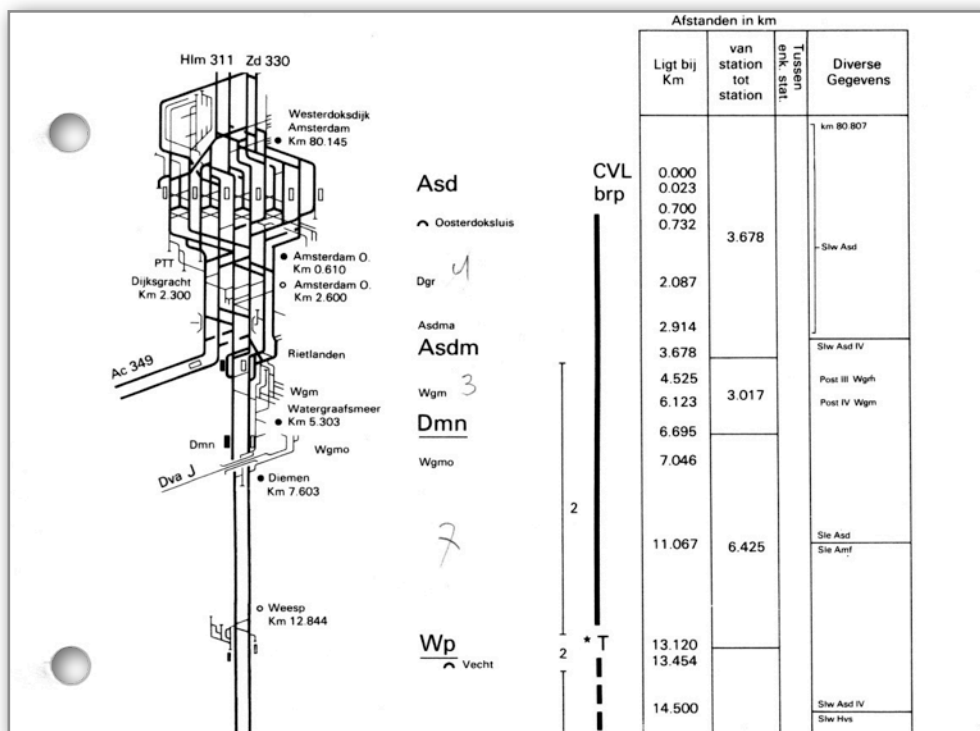


Figure 2. Track diagram; the width is exaggerated, the length is suppressed, kilometrage references are added (Source: gestyleerde baanvakken NS).

2.2. Levels of detail

When looking at data it is also important to distinguish between levels of scale. At a low level of scale you get all the details but miss overview. At a high level you get overview but miss the details. The required level of detail depends upon the purpose of your map.

Network level

On a network level you will see routes and stations. You will see the junctions, where routes interconnect, although more often than not these will remain unnamed. And often you will see some kind of metadata, like the operator of a route, single or double track, kind of electrification etc.

The most obvious task for a railway map is route finding, this is where a network level of scale is most appropriate.

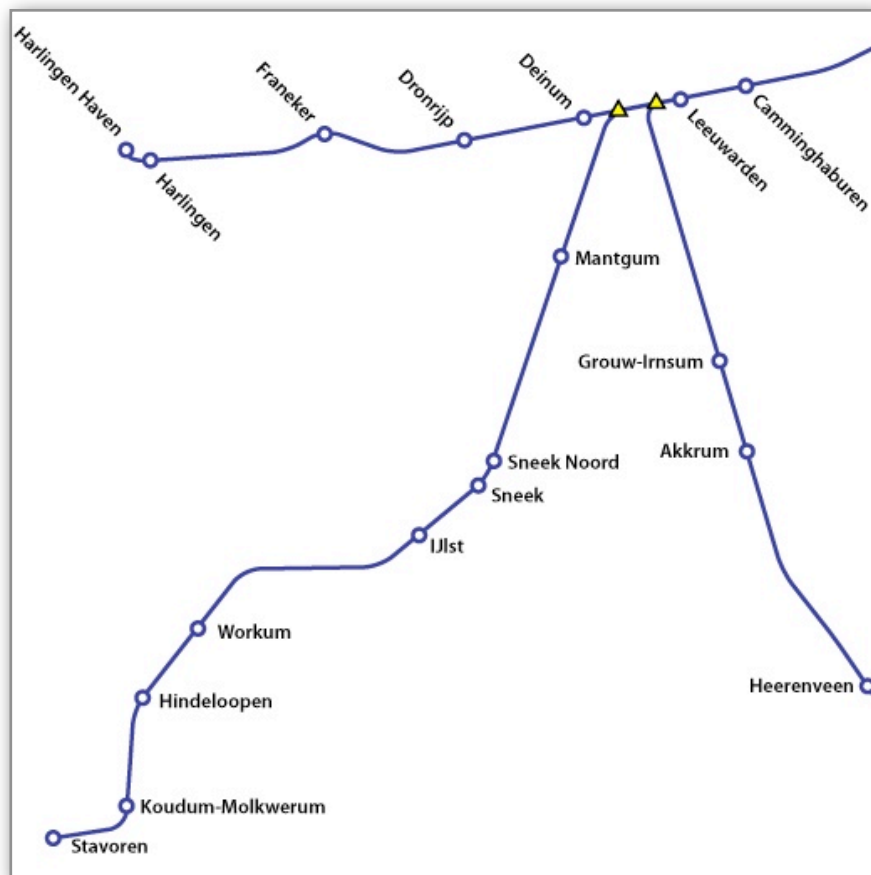


Figure 3. Map on network level: part of the Dutch railway network.

Information about the offered services (or: lines) can be superimposed upon a network map, leading to a more stylized, less geographically correct map. Many people use a map for visual route finding, even with a schematized map like the London Underground's.

But also for computerized route finding algorithms a network map can be a good basis (figure 3).

Station level

A map on a station level explains the layout of the station, for instance to help passengers find their way (figure 4).

Track level

On a track level each individual track and switch is visible. A signalman or dispatcher will need this kind of information to operate the system (figure 5).

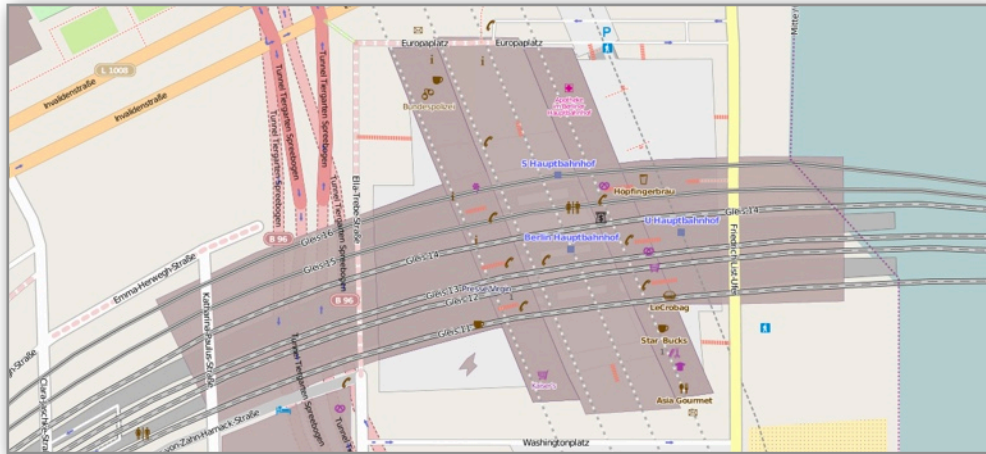


Figure 4. Berlin Hauptbahnhof on OpenStreetMap, showing information to help passengers find their way. (Mapnik)

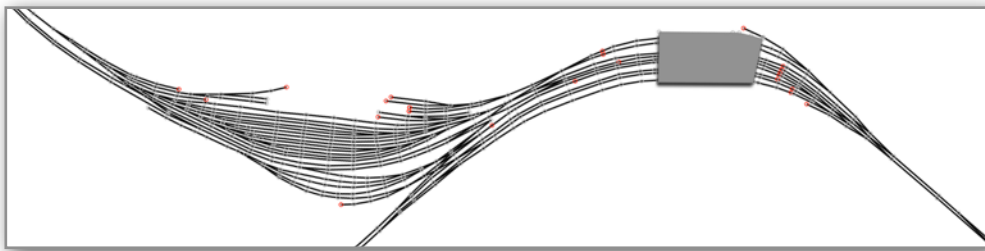


Figure 5. Track level of detail: Aachen Hauptbahnhof. (OpenStreetMap data)

2.3. Goals

My goal is to extract a good network level railway map from OpenStreetMap data:

- Consistent representation of the network (incl. junctions)
- Showing position of the stations
- Geographical accuracy; interface to geographical coordinates
- Interface to railway measurement (kilometrage / mileage)
- Providing hooks for lower level information
- Allowing for distance calculations

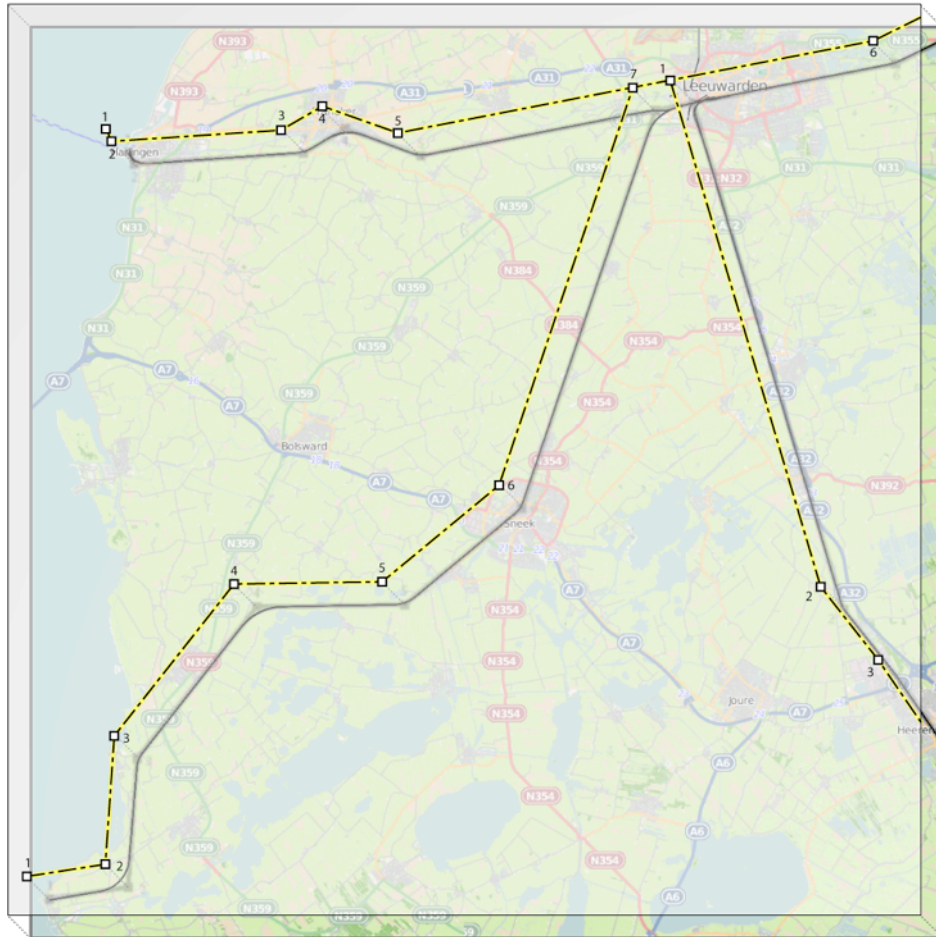


Figure 6. Modeling railway routes with heartlines, using a minimum amount of nodes.
(Sketch by the author upon OpenStreetMap background)

3. Approach

The basic idea is to define a heartline, marking the geographically accurate position of the railway line, with a minimum of nodes.

Normally a railway consists of:

- a perfectly straight section
- an easement curve
- a perfectly circular curve
- an easement curve
- a perfectly straight section
- ... and so on

Knowing this geometry it would be possible (and easy) to model a railway by just modeling the straight sections and filling the curves in, later on. (figure 6)

4. Creating the heartline

The first step is to get a dataset to work on. To get a whole railway route you obviously have to get data for a larger area. This is not a trivial task because OpenStreetMap datasets tend to get very large. An OSM contact could give me a dataset of the Netherlands, filtered to only railway information, which resulted in a manageable file size.

The next step is to find the OSM ways you want to base the new heartline upon. I have built my own editor to see and manipulate the data. Once I had that in place it was relatively easy to point one element of the desired railway and extend it to the whole route. When more parallel tracks were mapped (like in stations) I had to choose one of them. The result is an array (of typically several hundreds) of point, marking the route.

Then I had to distinguish the straight and the curve sections. I have done some efforts to make this a fully automated process, but my goal was to get results, not to get the ideal algorithm. My solution was to make a diagram that plots the length of the railway against the direction (figure 7). In the diagram I could manually indicate curve and straight sections, that would be translated by the program into a heartline. This would be a reasonable approximation that I could further improve with manual correction.

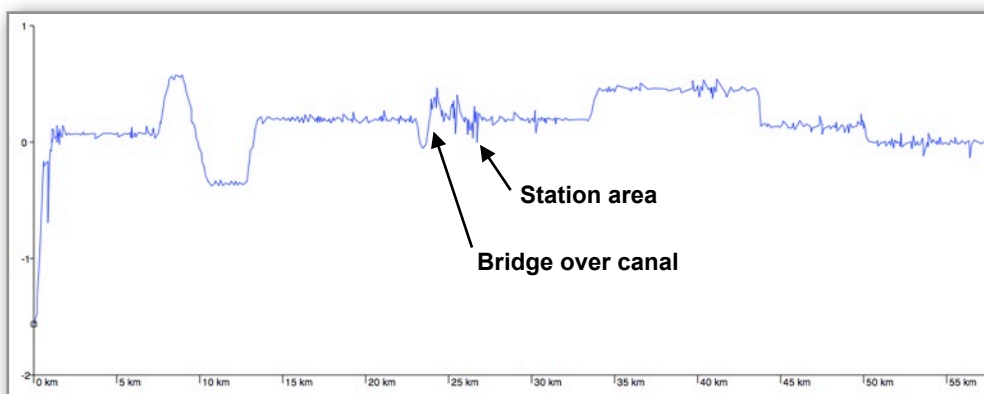


Figure 7. Diagram showing the direction of the railway (y-axis) against distance (x-axis). In the middle deviations, from a perfectly straight line, are visible, caused by bridge and station construction.

Finally I wanted to add kilometrage information. I used an old railway handbook that indicates kilometer position of all stations with a meter accuracy.

The point here is to find the right km value for the starting point at the downstream end of the railway route. I calculated the station positions with a zero value in my starting point and compared those with the values in the handbook. Then I made a correction for what the starting point should be. In most cases the accuracy of the length calculation over the whole railway route (typically 20 - 100 km in length) proved to be around 100m. In some cases the starting point of the route (situated at a junction or terminus station) appeared to be the exact zero point of the real railway measurement.

Of course other users won't have 'old railway handbooks'. But km indications ('milestones') can be easily spotted on an on-site survey. Because the heartline method appears to be accurate, just a few of such observations will allow to extend the kilometrage measurement over the whole route.

4.1. Modeling curves

The most obvious way to describe a straight section is to describe the end points, where the railway line transitions into a curve. The most obvious way to describe a curve is to point its center point and radius.

However, after some thinking and experimentation, I decided to use the (virtual) crossing point of two extended straight sections as the modeling node for the heartline. When manually editing the heartline it is relatively easy to position this nod in such a way that it makes a good fit for both straight sections. Next, a radius can be chosen that fits the curve.

As a consequence, the curve center and transition points have to be calculated by the computer. I did not use easing curves, but within this approach it would be possible as well.

For kilometrage calculations the curve has to be taken into account (calculating the length of the heartline from node to node would result in a larger value than the actual length of the railway). I solved this by calculating the kilometrage positions of the transition points and adding them to the meta-data. If you want to know the position of a point on the line you can simply refer to the closest transition point and calculate the distance from there.

This approach does imply some inflexibility: as soon as a node or radius on the heartline changes, all the distance information (at least upstream) has to be recalculated (figures 8 and 9).

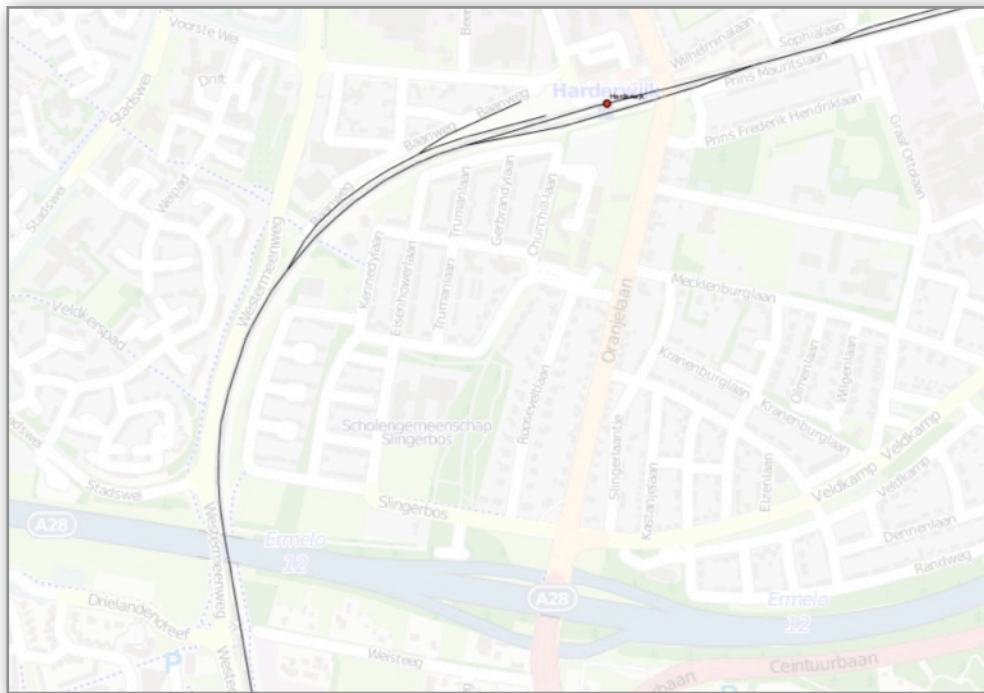


Figure 8. The “Harderwijk curve”, one of the more infamous curves in the Dutch railway network. (OSM railway data rendered over the ‘Mapnik’ map)

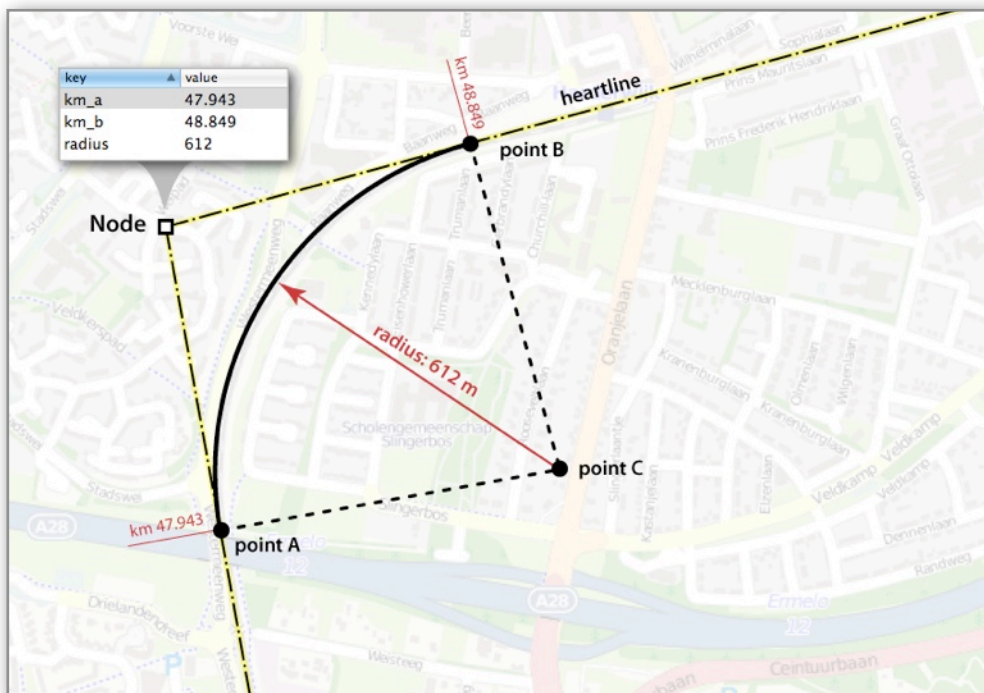


Figure 9. The same curve, modeled with a heartline. Information about curve radius and kilometrage has to be added as tags to the node.

4.2. Route naming

When building a map based upon routes, each route will need a unique identifier. Although I didn't find an unambiguous naming scheme, Wikipedia was a good reference. Wikipedia is very complete in summarizing all railway routes and in most cases the routes are consistent with (and explicitly show) kilometrage measurement.

To create identifiers I used the following scheme:

- A two character country code, followed by a dot
- A two to four character identifier that refers (in most cases) to the builder of the railway line, or the building scheme, or a well known surname of the route.
- A two digit index to distinguish different parts of the route, or branches.

Some countries (like Belgium) have more strict naming or numbering conventions for their railway routes and it would be logical to follow those. The appendix shows the identifiers I used for the Netherlands (see appendix).

5. Conclusion

5.1. Lessons learned

An editor is needed

My goal was to produce the heartlines. However, I needed a tool to do so and writing an editor was kind of a side effect. The editor is just a prototype, as I tried to focus my attention on the outcome, but still it took most of my effort.

Major simplification

It was possible to reduce the number of nodes in a railway route from several hundreds to a few dozen, while remaining geographically accurate. This makes it really interesting, when your need is on the network level, to use heartlines instead of the underlying detailed information.

Railway properties can be translated to 1D coordinates

Railway properties and attributes, like stations, bridges or level crossings, can be translated to 1D coordinates (km positions). For some purposes this would lead to a more manageable set of data.

Curve as a property to the node

The most effective way to model curves is to use the (virtual) crossing point of the straight sections as an anchor node and add the radius of the curve as a property to the node.

Spatial restrictions lead to more complicated railways

When railways grow older (or when they get built in more spatial restricted environments in the first place), they tend to get more twists and curves to adapt to their environment. For instance, when a new bridge is built the building site might be outside of the original heartline of the railway. When the building is finished the railway gets bended towards the new bridge, creating an additional curve in a previously perfectly straight line.

Sometimes it was interesting to be able to reconstruct history just by looking at the lines curves. Sometimes it was clearly visible, especially at large railway yards, how railway engineers must have struggled to fit their needs within available space. Sometimes subtle curves that made tiny adaptations to the railway environment made it more difficult to find an accurate heartline.

5.2. Discussion

Downloading aggregated data from OSM

In my eyes the lack of possibilities to download aggregated or higher level of scale vector information is a shortcoming of OpenStreetMap. By showing the railway heartlines I hope that I was able to give some idea of how such higher level information could look like. I think it would be beneficial for OpenStreetMap if you could filter out this kind of information.

Heartline as a template

Once a heartline is accurate it could be used as a template to position nodes in the desired straight or curve positions. I don't think a heartline could or should replace the current OSM "railway" ways.

Integrating heartlines in OSM data and editors?

It would be possible to upload heartlines to the OSM database. They can be tagged in such a way that current renderers don't see them. But they will be visible and editable in current editors. The likely consequence is that the heartlines will get edited in less desirable ways: they will get intermediate nodes and be slowly absorbed by the highly detailed OSM vector network.

I don't have an answer, your suggestions are welcome.

References

N.V. Nederlandse Spoorwegen (ca. 1990) Gestyleerde Baanvakken (company publication)

Appendix: Railway routes in the Netherlands

identificer	route
nl.am01	Aken - Maastricht
nl.ar01	Antwerpen - Lage Zwaluwe
nl.ar02	Roosendaal - Breda
nl.as01	Almelo - Salzbergen
nl.asd10	Amsterdam Muiderpoort - Amstel (verbindingsboog)
nl.asd11	Amsterdam Lelylaan - Zaandam (Hemboog)
nl.asd12	Amsterdam RAI - Amsterdam Bijlmer (Utrechtboog)
nl.br01	Rotterdam - Zevenaar (Betuweroute)
nl.gls01	Sauwerd - Roodeschool (Hogelandspoor)
nl.gols01	Winterswijk - Zevenaar
nl.gv01	Den Haag CS - Den Haag Laan van NOI (verbindingsboog)
nl.gv02	Den Haag CS - Den Haag HS (verbindingsboog)
nl.hsl01	Schiphol - Rotterdam (HSL-Zuid)
nl.hsl02	Rotterdam - Breda grens (HSL-Zuid)
nl.hsm01	Amsterdam - Rotterdam (Oude Lijn)
nl.hsm02	Haarlem - Uitgeest
nl.hsm03	Zaandam - Enkhuizen
nl.hsm04	Heerhugowaard - Hoorn
nl.hsm05	Schiedam - Hoek van Holland (Hoekse Lijn)
nl.hzsm01	Haarlem - Zandvoort (Zandvoortse Lijn)
nl.ijr01	Weert - Roermond (IJzeren Rijn)
nl.kl01	Nijkerk - Ede-Wageningen (Kippenlijn)
nl.knls01	Dieren - Apeldoorn
nl.knls02	Apeldoorn - Deventer
nl.knls03	Deventer - Almelo
nl.lm01	Luik - Maastricht
nl.lw01	Woerden - Leiden
nl.ncs01	Utrecht - Zwolle (Centraalspoorweg)
nl.ncs02	Zwolle - Kampen (Centraalspoorweg)
nl.ncs03	Den Dolder - Baarn (Stichtse Lijn)
nl.nols01	Zwolle - Emmen
nl.nols02	Mariënborg - Almelo
nl.nrs01	Amsterdam - Elten (Rhijnspoorweg)
nl.nrs02	Utrecht - Rotterdam (Rhijnspoorweg)
nl.nrs03	Harmelen - Breukelen (Rhijnspoorweg)
nl.nrs04	Gouda - Den Haag (Rhijnspoorweg)
nl.nwsm01	Zutphen - Winterswijk
nl.nzos01	Tilburg - Vught (Zuid-Ooster Spoorweg)
nl.nzos02	s-Hertogenbosch - Nijmegen (Zuid-Ooster Spoorweg)
nl.nzs01	Sittard - Heerlen (Zuiderspoor)

nl.os01	Amsterdam - Amersfoort (Oosterspoorweg)
nl.os02	Amersfoort - Zutphen (Oosterspoorweg)
nl.os03	Hilversum - Lunetten (Oosterspoorweg)
nl.os05	De Haar - Rhenen (Veenendaallijn)
nl.rtd11	Rotterdam - Maasvlakte (Havenspoorlijn)
nl.ssa01	Arnhem - Zwolle (Staatslijn A)
nl.ssa02	Zwolle - Leeuwarden (Staatslijn A)
nl.ssb01	Harlingen - Nieuweschans (Staatslijn B)
nl.ssb02	Leeuwarden - Stavoren
nl.ssc01	Meppel - Groningen (Staatslijn C)
nl.ssd01	Zutphen - Glanerbrug (Staatslijn D)
nl.sse01	Breda - Eindhoven (Staatslijn E)
nl.sse02	Venlo - Eindhoven (Staatslijn E)
nl.sse03	Maastricht - Venlo (Staatslijn E)
nl.ssf01	Roosendaal - Vlissingen (Staatslijn F)
nl.ssg01	Venlo - grens (Staatslijn G)
nl.ssh01	Utrecht - Boxtel (Staatslijn H)
nl.ssi01	Breda - Rotterdam (Staatslijn I)
nl.ssk01	Den Helder - Amsterdam (Staatslijn K)
nl.ssl01	Arnhem - Venlo
nl.ssm01	Schaesberg - Kerkrade Centrum (Miljoenenlijn)
nl.ssn01	Zwolle - Almelo
nl.sso01	Elst - Geldermalsen (Betuwelijn)
nl.sso02	Geldermalsen - Dordrecht (Betuwelijn)
nl.ssp01	Groningen - Delfzijl
nl.ssq01	Eindhoven - Weert
nl.ssr01	Heerlen - Schin op Geul
nl.sss01	Gouda - Alphen a/d Rijn
nl.sst01	Leidschenveen - Zoetermeer (Zoetermeerlijn)
nl.ssu01	Weesp - Duivendrecht (Schiphollijn)
nl.ssu02	Duivendrecht - Leiden (Schiphollijn)
nl.ssu03	Amsterdam - Riekerpolder asl. (Westtak)
nl.ssv01	Weesp - Lelystad (Flevolijn)
nl.ssv03	Naarden-Bussum - Almere-Muziekwijk (Gooiboog)
nl.ssv04	Lelystad - Zwolle (Hanzelijn)
nl.zhe01	Rotterdam-Hofplein - Scheveningen (Hofpleinlijn)
nl.zl01	Zwolle westzijde - Zwolle aansluiting (Zwolle lokaal)

OpenStreetMap For Traffic Simulation

Michael Zilske, Andreas Neumann, Kai Nagel

TU Berlin, Fachgebiet Verkehrssystemplanung und Verkehrstelematik

Abstract. Micro-simulations of traffic systems are becoming more important as highly disaggregated data, such as mobility diaries or GPS traces, become available. For accurate results, a high-quality model of the road network is required. Recently, OpenStreetMap has proven to be a valuable data source, often on par with and in some respects surpassing proprietary network models provided by administrations in usefulness.

Keywords: Traffic simulation

1. Introduction

MATSim, a software package for micro-simulation of traffic in urban systems, which has successfully been applied in transport policy analysis, is available on an open-source basis (Balmer et al. 2009). However, it is often difficult to obtain and maintain input data for such simulations. For instance, information about the road network needs to be collected from different government agencies and map providers, who often deliver it in non-standard formats, and datasets provided by different agencies are typically not explicitly referenced to each other.

In this paper, we describe our experience using OpenStreetMap (OSM) as a data source for traffic simulation studies, either on its own or augmented with additional data. The focus is on use cases in the preparation of simulation scenarios where OSM comes into play. MATSim couples a time step based queuing simulation of road traffic flow with a multi-agent travel demand model based on individual people who carry out daily plans. Both parts of the model can benefit from OSM as a data source.

2. Generating the Network

2.1. Road Characteristics

The queuing model of traffic flow (Gawron 2008) uses a graph-based model of the road network, where nodes are intersections and edges are road segments. The characteristics of each road segment are modeled as three parameters: *free speed*, *flow capacity* and *storage capacity*.

The flow capacity limits the number of vehicles which can leave a link in each time step, while the free speed sets a lower bound for the time each car has to spend on a link. Storage capacity is the maximum number of vehicles which a link can contain at any given time. When the link is full, the simulation does not add any more vehicles to a link. The congestion "spills back" upstream. Storage capacity is calculated from the link length and the number of lanes.

When such a network is constructed from OpenStreetMap, the link length is taken directly from the geometry. This is the only geometric information required for the model. After taking the link length into account, a road segment is simplified to a straight line, and intermediary nodes between road crossings are discarded. We use a custom tool which simplifies the network when it is already in our own data format. It has the option to block certain link and nodes from being removed, which is important for things like bus stops or links referenced by traffic counts (*Section 2.2*).

Free speed and flow capacity are characteristics of road types (FGSV 2009) which we identified with the values of the OSM `highway` tag. Free speed is not identical to the legal speed limit, but the two are related. For instance, while large roads within city limits typically have a legal speed limit of 50 km/h, the average free speed is decreased by traffic signals. An average of 25 km/h was determined for the primary road network of Berlin. During a green wave or when driving faster than the legal limit, this can at times rise to 44 km/h, but this is countered by poorly coordinated signals on other parts of the route. On the freeways (Autobahnen), on the other hand, the legal limit (which typically does exist within city boundaries) is seldom honored. Here, we defined the free speed to be 1.2 the legal limit. In areas where speed limit zones are extensively used (mostly to 30 km/h), time which is lost at unsignaled nodes governed by „right-before-left“ priority is often countered by driving in excess of the speed limit, leading to an overall factor of 0.8 for tertiary roads. On streets tagged as `living_street`, the legal speed limit is prescribed as „walking speed“, which in reality is almost never honored. In lack of better data, we assume 15 km/h. The parameters per highway tag which we used in our studies are summarized in the table below.

	Legal speed limit (km/h)	Average speed / Legal speed limit	Capacity (veh/h)	Number of lanes
motorway	100	1.2	2000	2
motorway link	60	1.2	1500	1
trunk	50	0.5	1000	2
trunk link	50	0.5	1000	1
primary	50	0.5	1000	1
primary link	50	0.5	1000	1
secondary	50	0.5	1000	1
tertiary	30	0.8	600	1
minor	30	0.8	600	1
unclassified	30	0.8	600	1
residential	30	0.6	600	1
living street	15	1.0	300	1

Table 1. Highway characteristics inside city boundaries

The queuing model uses directed edges, so for each road segment except for those which are tagged as `oneway`, two links are created.

The region of interest for our studies has typically been an urban area. The road network of the region of interest is extracted from `planet.osm`, or more conveniently from one of the excerpts made available by `geofabrik.de`, by means of the open-source tool `Osmosis`, which permits extracting the road network of the entire region of interest by specifying a bounding box for the city, and a tag filter selecting all highway types accessible by car. When traffic is later assigned to the network, it is helpful not to cut the network at the city limits, but to embed it into its environment by adding the primary roads of the commuting range of the city. It can then be determined by routing if and on which way a person commutes from A to B by passing through the city.

2.2. Working with Additional Data

To calibrate simulation parameters and to validate results, traffic counts are used. Local authorities obtain them by means of induction loops or traffic cameras. These data are typically referenced to a proprietary network used by the respective agency, or they only carry coordinates but no explicit reference to a road segment. For one of our studies, we manually referenced count stations to OSM nodes and saved the OSM node ID to the counts database. In rare cases, where no suitable node was available, we inserted a node into OSM specifically as a reference point for our data. It turned out that nodes are stable enough to enable this workflow. Still, with every

update from OSM it has to be verified that the reference nodes still exist, and any changes to their location have to be monitored. This is not different from other data sources, where there are also no guarantees about map entities keeping any properties from one version to the next.

In another case, we obtained a large set of floating car data (FCD) within the study area. They were given as point measurements of velocity, georeferenced and annotated with the approximate driving direction, but not matched to road segments. We referenced each data point to the nearest network link which approximately matched the given direction and used it to update the free speed of the referenced road segment to the observed maximum speed.

Matching points to links is only a special case of the more general problem where a network generated from OSM must be augmented with data referenced to a different network. In a future project, we plan to evaluate using a general network matching algorithm to create a simulation scenario where the base data comes from OSM, but specific extra data like traffic signal plans or volume counts come from different network graphs which are semi-automatically matched to OSM.

2.3. Junction Layout

Apart from a higher tagging rate of attributes from which flow capacity, free speed and number of lanes can be inferred, traffic simulation applications would benefit from a detailed representation of junction layouts (Krajzewicz et al. 2005). In the queue model as described above, the link capacity controls the rate at which cars flow out of a link. In reality, different turn relations at nodes have different capacities, which among other things (like traffic signaling) depend on the number of turn lanes available. If this is modeled by decreasing the capacity of certain turn relations, this raises the question of how to handle the traffic going in other directions: If cars are removed from a link strictly on a first-in, first-out basis, a car going straight will be forced to wait behind a car going left which is kept by a capacity restriction on the turn, while in reality it may well be that turning traffic has its own lane and can be passed. On the other hand, if cars going in either direction actually share the same lane, this would again be inaccurate.

A straight-forward solution is to split the link into several ones, one for each lane. For this to work, it needs to be known how many lanes are available per turn relation, and also their length, since once a turn lane runs full, turning vehicles will spill back to the main lane. This kind of data is not yet available in OSM. Benjamin Schulz (2011), a student in our group, proposed a tagging scheme for this information as part of his thesis, and he also developed a graphical editor as a plug-in for the comprehensive

OpenStreetMap editor JOSM. It allows adding extra turn lanes to the links leading up to a node, specifying into which link each turn leads, and moving the beginning of the lane with a slider. The turn relation can lead through complex junctions consisting of multiple nodes and permits special cases like U-turn lanes. While the tagging scheme is an early proposal and the software is a prototype, we intend to develop it further and to use it to investigate if there is a demand for this kind of detailed information and a willingness on part of the community to contribute it once a convenient editor is available.



Figure 1. The turn lane editing pane in JOSM.

2.4. Public Transport

If the focus of a traffic simulation is on private cars, a user switching to public transport can be modeled as removing herself from the system and experiencing a travel time estimated by aggregate characteristics of the transit system, for example twice the time required by car on an uncongested network (Rieser et al. 2009).

This simple approach cannot take into account the varying transit accessibilities of different locations. A more fine-grained approach is possible if a complete transit schedule is available, with route definitions, stop locations and departure times. In this case, agents plan a route through the transit network, wait for the next departure and arrive at their destination according to the schedule, although the simulation can take vehicle capacities into account, as well as access and egress times. This means that effects like overcrowded vehicles, where passengers have to wait for the next departure, can be simulated, and vehicles are delayed if a larger number of passengers are boarding or leaving than the schedule accounts

for (Rieser 2010). The users can be charged fares, so the utility of the ride can be weighed against using a private car. With the General Transit Feed Specification (GTFS), developed by Google, there is a data format available which contains all the required information. However, as of 2011, not one transit authority in Germany publishes a GTFS feed. Any study including public transport at schedule-level requires coordination with the local transit authorities, and manual data conversion.

While some public transport modes, like railway, normally operate without interaction with car traffic, this does not hold for busses, which can be caught in congestion together with private cars. If it is desired to simulate this kind of interaction, one requires not only the transit schedule, but the exact routes the busses take through the road network. Transit authorities do not publish their schedules referenced to the OSM network, of course. A GTFS feed can provide detailed geometric route information as an optional feature and at the discretion of the transit authority. But even if one is available, the shape of this route must still be matched against the OSM network to insert detailed link-based routes into the transit schedule.

Public transport tagging in OSM itself is in a proposal stage. Despite efforts, as of 2011, we have not been successful in extracting sufficiently consistent route information for even the main lines of the transit network in major German cities.

3. Generating the Synthetic Population

The synthetic population which models the transport demand is typically generated from survey data. The available data may be as fine grained as complete mobility diaries of a population sample, containing georeferenced activity locations, durations and the means of transport taken in between. However, this information is often not available. The least amount of data required for a meaningful simulation is home and work locations for the synthetic population. These can be drawn from a *commuter matrix*, which for each pair over a set of polygons contains the number of people commuting from a home location in polygon 1 to a work location in polygon 2 on a typical work day. These polygons are typically administrative areas.

Depending on the scale and the desired accuracy of the simulation scenario in question, it may be sufficient to randomly draw points from the entire administrative area and designate them as home and work locations for a person. In our simulation, activity locations are on links, so each randomly drawn point is matched to its nearest link. This approach is sufficient if the scale of the given polygons matches the simulated network level. For example, if only commuter traffic on the primary network is simulated,

locations may be drawn from cities or counties without much consideration for different land use, since the locations are snapped to the next primary road. On the other hand, if commuter data and polygons are available on the level of city quarters or even building blocks, the entire road network may be used without any ill effects, such as small roads becoming unrealistically congested because the simulation places homes where, in reality, nobody lives.

In this process, OSM can be used in two different respects: First, the administrative areas themselves may be obtained from OSM. In real-life data obtained from local governments, it is interestingly not always the case that the polygons referenced by the commuter matrix are delivered in a common data format or even with a common coordinate reference system. If, however, the reference is made using a common identifier as a key (for example German administrative areas, which are multipolygon relations tagged with `de:amtlicher_gemeindeschluessel`), the shape may be taken directly from OSM, which saves writing a converter for the specific data.

Secondly, OSM may help to bridge different levels of aggregation. For instance, if commuter data is only available at the level of cities and counties, but a simulation of the entire road network is desired, drawing locations evenly is problematic, for the reasons stated above. This could be solved by employing a distribution which takes land use or even buildings into account, as available in OSM. A first step in this direction is to restrict the area from which activity locations are drawn to certain values of the `landuse` tag, or to assign weights to different values, so that home locations are concentrated in areas tagged as `landuse=residential`. How to refine this, and in particular how to deal with the problem that the availability of this information may vary widely within the area of one simulation scenario, will be the subject of future work.

4. Visualization

An agent-oriented traffic simulation produces trajectories of individual vehicles or pedestrians. The simulation environment MATSim features an interactive visualization tool which can display the current location of each agent during a simulation. This is an important tool for the user in order to spot regions with implausible traffic patterns caused by errors in link attributes or in the population structure.

Agents are rendered as discs moving over a simple schematic view of the network graph, where links are lines with a width corresponding to the number of lanes. A user who is familiar with the simulated scenario will be

able to identify locations from the road layout alone. However, for a detailed analysis, it may be preferable to use a properly rendered street map as a background image.

We were able to extend our visualization tool with a background layer of OSM tiles using the JMapView component. In principle, this would also be usable if the network data for the simulation were taken from a different source. However, as with any map overlay, this would lead to inaccuracies. When the cars move on the same network which is rendered as a background, the visual impression is much clearer.

5. Conclusion

In this paper, we described a workflow for generating multi-agent traffic simulation scenarios based on OpenStreetMap. For detailed, calibrated studies, additional data will be necessary, but in principle, it is possible to set up a basic simulation based only on OSM and a commuter matrix. While it is still difficult to link public transport information to OSM, the ongoing efforts to establish a tagging scheme for transit routes is promising. Finally, using a network for the simulation which at the same time can be rendered as a high-quality background map has substantial benefits for the visualization of results. We think that Volunteered Geographic Information will play a large part in making it easier to set up and maintain large-scale traffic simulation scenarios.

The software component which converts OSM data to a MATSim network is part of the MATSim distribution. The JOSM plugin for turn lanes has been committed to the JOSM plugin repository by its author.

References

- Balmer M, Rieser M, Meister K, Charypar D, Lefebvre N, Nagel K, Axhausen K (2009) MATSim-T: Architecture and Simulation Times. *Multi-Agent Systems for Traffic and Transportation*, IGI Global, 57-58
- Forschungsgesellschaft für Straßen- und Verkehrswesen (2009) *Handbuch für die Bemessung von Straßenverkehrsanlagen*, FGSV Verlag GmbH
- Gawron C (1998) An Iterative Algorithm to Determine the Dynamic User Equilibrium in a Traffic Simulation Model. *International Journal of Modern Physics C* 9(3):393-407
- Krajzewicz D, Hertkorn G, Ringel J, Wagner P (2005) Preparation of Digital Maps for Traffic Simulation; Part 1: Approach and Algorithms. *Proceedings of the 3rd Industrial Simulation Conference 2005*:285-290

- Pendyala R.M., Development of GIS-Based Conflation Tools for Data Integration and Matching. Research Center, Florida Department of Transportation
- Rieser M, Grether D, Nagel K (2009) Adding Mode Choice to a Multi-Agent Transport Simulation. Transportation Research Record 2132:50–58
- Rieser M (2010) Adding Transit to an Agent-Based Transportation Simulation. Dissertation, TU Berlin
- Schulz B (2011) Abbiegespuren und Kreuzungslayout in OpenStreetMap. Bachelor's Thesis, TU Berlin
- Strippgen D (2009) Investigating the Technical Possibilities of Real-time Interaction with Simulations of Mobile Intelligent Particles. Dissertation, TU Berlin

Evaluating Urban Traffic Congestion using OpenStreetMap and Sparse Public Transport Data

Andrei Bejan

Statistical Laboratory, Centre for Mathematical Sciences
University of Cambridge
Wilberforce Road, Cambridge CB3 0WA, UK

Abstract

Congestion in urban areas causes financial loss to business and increased use of energy compared with free flowing traffic. Providing citizens with accurate information on traffic conditions can encourage journeys at times of low congestion and uptake of public transport. Installing the measurement infrastructure in a city to provide this information is expensive and may invade privacy. Public transport location data recorded over long periods of time as well as “live” feeds containing such data are increasingly becoming available—public transport vehicles are equipped with sensors to provide real-time arrival time estimates, but one important feature of these location data is that they are sparse. Our recent work within the TIME project [21] has shown how to deal with the sparsity and use these data to estimate journey times and evaluate congestion dynamics across a city. We describe the (public transport) bus data and show how the specifics of these data may be used in conjunction with OpenStreetMap data to reconstruct useful information on transport traffic flow dynamics. We have found that OpenStreetMap data and some of its features (not only the topological description of the road network, but its more detailed infrastructure and meta characteristics as well) have a great potential in analysing public transport data in a meaningful way. This, however, relies heavily on the accuracy, coverage and level of detailisation of the corresponding features of the map, that is on the *state of the map*. We discuss these issues using real data in the context of the city of Cambridge, UK. This talk is based on a joint work with Jean Bacon, Alastair Beresford, David Evans, Adrian Friday, Richard Gibbens and Ken Moody.

Keywords. Transport traffic, bus probe data, congestion, OpenStreetMap

1 Introduction

Public transport location data recorded over long periods of time as well as “live” feeds containing such data are increasingly becoming available. Over the last decade, the bus fleet in the UK public transport sector has become equipped with Real Time Information (RTI) systems. For example, it was reported in the recent UK Public Transport Technology Survey (Knoop and Eames 2009), based on data received from local authorities in England, Wales, Scotland and Northern Ireland in 2008, that 56 Local Authorities (compared with 37 in 2007) had operational RTI systems covering approximately 185 towns and cities each with a population over 10,000.

Congestion in urban areas causes financial loss to business and increased use of energy compared with free flowing traffic. Providing citizens with accurate information on traffic conditions can encourage journeys at times of low congestion and uptake of public transport. Installing the measurement infrastructure in a city to provide this information is expensive and may potentially invade privacy. Increasingly, public transport vehicles are equipped with sensors to provide realtime arrival time estimates, but these data are sparse. Our work within the TIME project (Bacon et al. 2011, Bejan and Gibbens 2011, Bejan et al. 2010) has shown how these *floating probe* data can be used to

- (i) estimate journey times experienced by road users, and
- (ii) evaluate congestion levels in urban areas and thus provide better information for road users and traffic managers.

In this talk we will describe the bus data and show how their specifics may be used in conjunction with OpenStreetMap data, in order to process and analyse the data, and reconstruct useful information on transport traffic flow dynamics. We have found that OpenStreetMap data and some of its features (not only the topological description of the road network, but its more detailed infrastructure and meta characteristics as well) have a great potential in analysing public transport data in a meaningful way. This, however, is heavily dependent on the accuracy, coverage and level of detalisation of corresponding features of the map, that is on the state of the map. We will discuss these issues using real data in the context of the city of Cambridge, UK.

The rest of this paper is organised as follows. Section 2 contains data description; this includes the bus location data used in this study as well as various information extracted from OpenStreetMap database. These data and additional information are then used in Section 3 which provides details on the analysis of journey times and traffic congestion in urban areas; this analysis is illustrated using case studies and the role of the OpenStreetMap data is highlighted. Finally, Section 4 offers concluding remarks.

2 Data Description

2.1 Sparse and High Resolution Bus Probe Data

The bus location data used in this work were provided by the company providing real-time information facilities to the largest bus operator in Cambridge, England. The data represent a set of bus location points recorded over a period of 4 years (2007–2010). Many of the buses whose positions were recorded travel along main radial roads and multi-lane highways; some connect Cambridge to villages in the surrounding area.

Table 1 shows the average number of buses contributing probes throughout most of 2009 and beginning of 2010. Values for the following three areas are given: the city of Cambridge, Cambridge and surrounding villages, and the encompassing area of Cambridgeshire and Bedfordshire. In 2009 there were on average 115 buses equipped with GPS units within Cambridge on weekdays, 100 buses on Saturdays, and 65 buses on Sundays. In April 2010 the required GPS units and transmission hardware were installed in approximately 50 more buses of the fleet in question. (Figure 5 in Bacon et al. 2011 provides more detailed information on the dynamics of the number of instrumented buses for the specified period of time and geographic areas.)

area	weekdays	Saturdays	Sundays
City of Cambridge	115	100	65
Cambridge and surrounding villages	165	140	90
Cambridgeshire and Bedfordshire	325	305	190

Table 1: Average number of buses equipped with GPS units which were operating in Cambridge and Cambridgeshire, England throughout 2009.

Bus location sampling points were given in a local “easting/northing” (EN) coordinate system (similar to the British Ordnance Survey national geographic reference system—e.g., see [17] in the reference list) which can be regarded as a Cartesian system within the scale of a medium-sized city. The grid squares of this coordinate system are of size 10 m $\times$ 10 m and, given the area covered by Cambridge, there is a simple linear transformation to conventional latitude/longitude coordinates. We estimated the accuracy of each location report as ± 30 m. Each bus reported its location once every 20 or 30 seconds, though rarely every 60 or 120 seconds. There were missing observations, meaning that there were periods of ‘silence’ when some or all buses were not transmitting their positions due to communication failure, temporary hardware malfunction, weather, or other reasons. Finally, the location reports did not describe which bus route each bus was taking.

It has been shown in Bejan et al. (2010) that the bus probe data, as described above, give a good coverage of the main city roads including arteries. The temporal coverage is similarly good: the data are available throughout the day, including peak morning and afternoon hours when traffic conditions are of most interest.

In Section 3.3 we outline our approach, based on monotonic spline interpolation, in using the sparse bus data to determine bus behaviour between sampled locations. In order to validate this approach a set of high resolution (“High-Res”) traces was collected using a portable GPS device and carrying it on board the buses. The HighRes traces recorded the bus locations once per second with an accuracy of 5 m to 15 m.

2.2 OpenStreetMap

2.2.1 Road Network Description and Features

We have made an extensive use of OpenStreetMap (OSM) to obtain information about the road network and some of its infrastructure, including locations of junctions and traffic lights as well as bus lanes. We also used OpenStreetMap for visualisation purposes. OpenStreetMap (<http://www.openstreetmap.org/>) is a collaborative project with the aim of creating a free, public map of the world. The content is contributed voluntarily, using geographic data from portable GPS devices, aerial photographs, and other free sources (e.g., see Haklay and Weber 2008). The quality of this volunteered information has become increasingly high in some countries and geographic regions and is comparable to commercially produced maps (e.g., see Haklay 2010 and OSM Analysis by Ito World Ltd¹).

2.2.2 Bus Stops and Bus Route Information

Our analyses required knowing for each bus stop its location and the direction of bus travel past it (the stop’s “bearing”). Although some information about locations of bus stops can be obtained from OSM, our primary source was the UK National Public Transport Access Node (NaPTAN) database, an official list of public transport access points in the UK. The UK Department for Transport has offered to make this database available to the OpenStreetMap project and at the time of writing the incorporation of NaPTAN data into

¹http://www.itoworld.com/static/osm_analysis.html

OSM was in progress.² More details on our journey extraction procedure are given in Section 3.1.

For map-matching we used knowledge of the sequence of roads which buses follow while assigned to a certain route. This was initially derived using timetable information from the bus operator supplemented by bus probe data. However, at later stages of our study we found that, for Cambridge at least, this information was sufficiently complete in the OSM data. (Graphical representation of bus routes that are described in the OSM data can be accessed at <http://www.openbusmap.org/>.) However, our experience and observations suggest that temporary changes of bus routes due to road works related to gas or water main repairs or upgrades may often not be reflected sufficiently quickly (or not reflected at all) in the OSM data. This may be a serious limitation on using this kind of information from the OSM data for any application which may need to rely on this information.

3 Evaluating Congestion in Urban Areas using Sparse Public Transport and OpenStreetMap Data

3.1 Extracting Trajectories

We used bus stop location and bearing information derived from the NaPTAN and OSM data sets in order to extract bus journeys from sparse bus data. Journeys were extracted by constructing a line segment for each stop, perpendicular to buses' travel, called a "bus stop gate". A sequence of these gates defines a bus journey. Fig. 1 includes a trace of a single bus' location samples (shown as red crosses) connecting central Cambridge with the village of Histon in the north of the city. Horizontal lines show bus stop gates for a series of stops along the way and, with corresponding bearings, and define the route of interest (black solid line). These gates can be used to extract the depicted journey (and any other such journey) and estimate its duration from the entire dataset of sparse bus data. We give more detailed explanation of how this has been done next.

Let us assume that the movement of bus k can be described as a sequence of vectors $\mathbf{y}_i^{(k)} = (t_i^{(k)}, x_i^{(k)})$ whose components are the timestamps $t_i^{(k)}$ and bus positions $x_i^{(k)}$. Let $r_{AB} \equiv A \rightsquigarrow B$ be a predefined route connecting two points A and B on the road network map. We are interested in extracting bus journeys connecting A and B taking into account the sparseness of the data, that is the fact that consecutive bus locations were recorded every 20,

²<http://wiki.openstreetmap.org/wiki/NaPTAN/Import>



Figure 1: This map shows a northern part of Cambridge, England with Histon Road connecting the town to the villages of Impington and Histon. The map includes bus stop gates (horizontal segments) at selected bus stops (shown by larger symbols), introduced in text, and used to extract the south-bound journeys along Histon Road (all such journeys will lie on the black solid line). Red crosses show an example of a bus trajectory extracted from the sparse bus data set using this sequence of bus stop gates. © OpenStreetMap (<http://www.openstreetmap.org/>) and contributors, CC-BY-SA (<http://www.creativecommons.org/>).

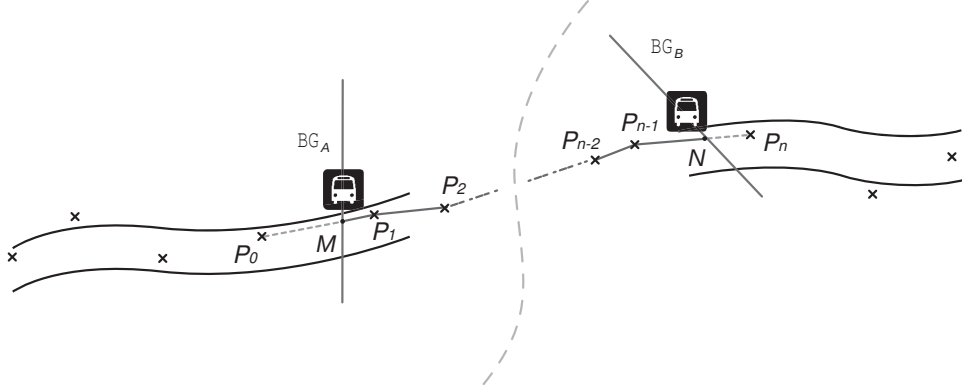


Figure 2: Extraction of a journey and journey time adjustment for intersection with the first and last bus stop gates. In the depicted situation the bearing of the bus stop A is E (East) and the bearing of the bus stop B is NE (North-East).

30 or more seconds apart ($t_i^{(k)} - t_{i-1}^{(k)} \geq 20$ sec). In order to do this, we used information about locations of bus stops and the direction of travel of buses served by these bus stops. The NaPTAN database, as well as OSM, includes coordinates in latitude/longitude of bus stops and assigns each bus stop a direction of travel from the set $\{N, NW, W, SW, S, SE, E, NE\}$. This information is used to approximate r_{AB} by a sequence of bus stops and bus stop gates as introduced above. We then find journeys corresponding to the route r_{AB} by looking at those piecewise linear path approximations to the bus trajectories which intersect all the bus gates which approximate r_{AB} . Moreover, it has been shown in Bejan et al. (2010) that this approach can be expressed and efficiently implemented as a string matching problem.

We now briefly describe how one can estimate durations of journeys extracted in this manner. Let $\rho \equiv P_0, P_1, \dots, P_n$ be a sequence of observations corresponding to bus k made at times $t_{i_1}^{(k)}, t_{i_2}^{(k)}, \dots, t_{i_n}^{(k)}$ such that ρ represents a journey along the route r_{AB} which is specified by a sequence of bus stop gates $BG_A \equiv BG_1, BG_2, \dots, BG_s \equiv BG_B$. Let $M = P_0 P_1 \cap BG_A$ and $N = P_{n-1} P_n \cap BG_B$ as shown in Figure 2. We estimate the corresponding journey time $JT(\rho; r_{AB})$ as

$$JT(\rho; r_{AB}) = t_n^{(k)} - t_0^{(k)} - \left(\frac{P_0 M}{P_0 P_1} \delta_1^{(k)} + \frac{N P_n}{P_{n-1} P_n} \delta_n^{(k)} \right),$$

where $\delta_i^{(k)} := t_{i+1}^k - t_i^k$. Thus, the time taken on the journey is the time it took for the bus to travel from P_0 to P_n minus the time spent by the bus before it crossed the first bus stop gate and after it crossed the last one.

It is worth noting that there is a longstanding debate within the OSM commu-

nity on whether bus stops, e.g. tagged by `highway=bus_stop`, should be part of the way describing the road or beside the road where their actual positions often are.³ The journey extraction procedure described above is invariant with regard to this choice—all that this procedure requires in order to work correctly are the coordinates of the bus stop, regardless whether it is on or beside the road, and its bearing. If a bus stop is a part of a one way road, then the bus stop direction simply coincides with the direction of travel on this road and follows from the order of the way’s nodes in the way’s description. If, however, the bus stop is a part of a bidirectional road way, then the bearing of this bus stop can be inferred by comparing the way’s description to the value of the tag `direction` in the description of the bus stop; this, if present, takes either value `forward` or `backward`. Appendix A contains examples of bus stop descriptions of various kinds; these include:

1. an example of a bus stop whose description was imported from NaPTAN database (Appendix A.1); bus stops are described in this database according to side-of-the-road/location of pole convention and bus stop bearings are described by a special tag;
2. an example of a bus stop which is a part of a road way and whose bearing is specifically indicated by the tag `direction` (Appendix A.2);
3. an example of a bus stop which is mapped using ‘beside the road’ approach and whose bearing is impossible to identify unambiguously (Appendix A.3);
4. an example of a bus stop which is a part of a road way, but whose bearing is still impossible to identify, as no additional information is provided (Appendix A.4).

Finally, it is worth noting that an additional advantage of the bus stop gate technique described above is that one can easily artificially define additional bus stop gates in order to represent the route of interest or make it more precise wherever the existing bus stops are not sufficient for this purpose.

3.2 Journey Times and Contextual Factors

Journey times experienced by users and their variability is one possible measure of traffic conditions and congestion on roads. We can use the bus stop gate technique described above to extract and statistically describe journeys along a route of interest. Our example here refers to Histon Road, Cambridge presented in Figure 1 and first appears in Bejan et al. (2010). We use four bus stop gates, as depicted in this figure, each of 300 m length to define a route along Histon Road towards the city centre; that is the direction of travel is

³http://wiki.openstreetmap.org/wiki/Talk:Tag:highway=bus_stop

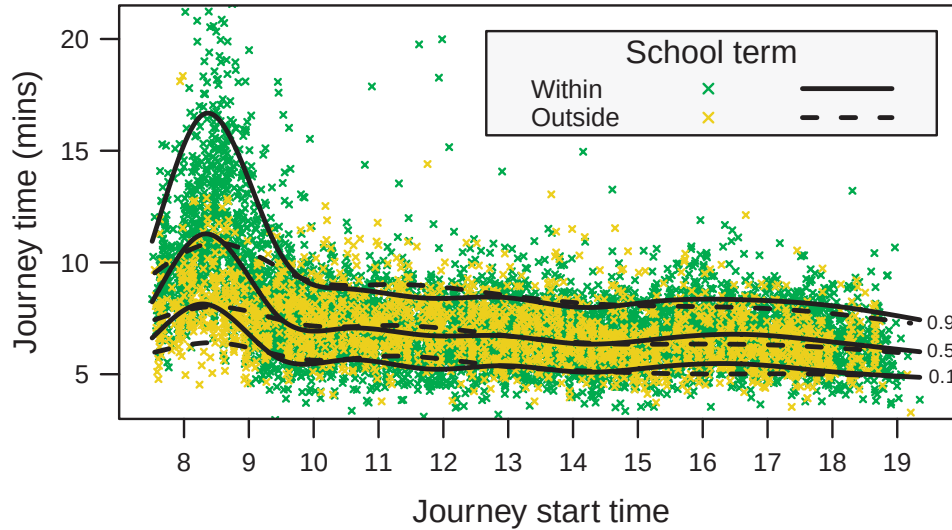


Figure 3: Green crosses show the duration for journeys taken on school days (non bank holiday weekdays during the school term). Yellow crosses show journeys for non bank holiday weekdays outside of school term. The solid and dashed lines are quantile regression lines corresponding to the 10th, 50th and 90th percentiles for term and non-term days, respectively.

from north to south and all four horizontal bus gates have bearing S (South). The NaPTAN codes of the four bus stops used in this example are as follows: `cmbgadwd`, `cmbgjdag`, `cmbdadap`, `cmbdagmd`. Thus, the bus stop corresponding to the first of the listed codes, `cmbgadwd`, is described in the `planet.osm` as a node with the tags `highway` and `naptan:NaptanCode` whose values are `bus_stop` and `cmbgadwd` respectively; the value of the tag `naptan:Bearing` describes the bearing of this node and is `S`. See Appendix A.1 for the full OSM description of this bus stop and its NaPTAN tags.

The extracted data set consists of 13,653 inbound journeys (in the direction from north to south) between Sunday, 2 November 2008 and Saturday, 9 January 2010. The minimum journey time was 2.95 minutes and 99.5% of the journeys took less than 20 minutes. The mean journey time was 7.22 minutes with a median value of 6.75 minutes. Associated with each journey is the start time and the date (and thus the day of the week) when the journey took place. Bejan et al. (2010) explored various ways to graphically illustrate statistical variability of journey times and dependence on the journey start time, day of the week or time of year. These graphical exploration methods include scatter and box and whisker plots as well as quantile regression technique and associated shaded plots.

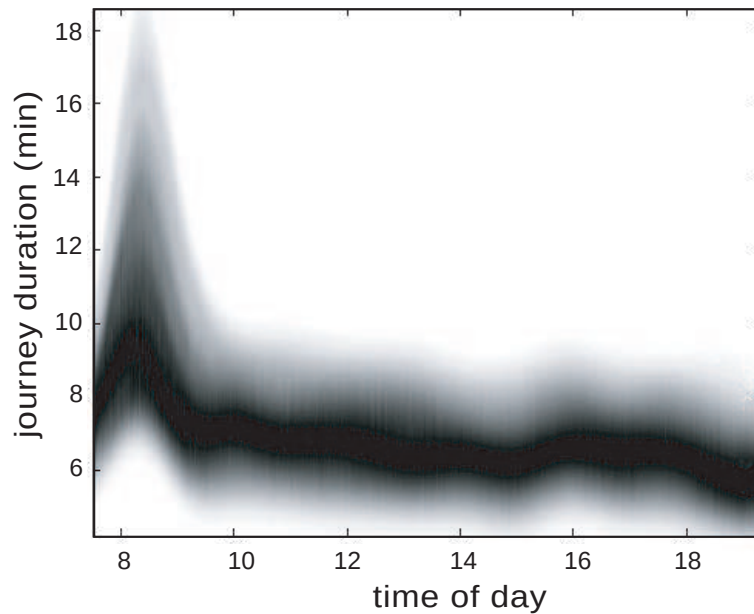


Figure 4: This figure shows a grey-scale heatmap plot of conditional densities derived from the set of quantiles from the quantile regression. Each such conditional density corresponds to a fixed time at which travel has been made and is obtained by resampling from the corresponding conditional cumulative distribution function with subsequent kernel density estimation.

Figure 3 illustrates journey times for the extracted journeys along Histon Road and highlights the effects mentioned above. Each individual dot refers to a single journey—green dots refer to journeys undertaken on school days while the yellow ones denote journeys made on non-school days (journeys taken on bank-holidays were removed). The two overlaid scatter plots show the striking effect of school terms as well as time of day effect on journey times. A quantile regression model was fitted to these data (see Bejan et al. 2010 for further details). During the busy period in term time journey durations range over a much larger interval. One can see this most clearly with the quantile regression models (lines) for the 10th, 50th and 90th percentiles. During school term the quantiles (solid lines) show a strong variation during the morning busy period. However, outside of school term (dashed lines) the variation is notably reduced. The technique of monotonic spline interpolation and local times outlined in Section 3.3 allows one to pin-point precisely where, and for how long, journeys were delayed.

Figure 4 further generalises the idea of using quantile regression lines in order to graphically represent distribution of journey times conditional on the time of the day at which a journey was made. This figure shows a grey-scale heatmap of conditional densities of such distributions obtained from the corresponding conditional cumulative distribution functions for fixed start times combined with kernel density estimation. Further details on this technique can be found in Bacon et al. (2011).

3.3 Evaluating Congestion on Roads using Sparse Bus Data and OpenStreetMap

Floating probe data from GPS sensors or GPS-enabled devices attached to vehicles or carried by people can be used to measure congestion by reconstructing point-to-point journey times, traffic speed or its intensity. Probe data consist of a sequence of coordinates recorded over time and may deliver considerably more information than is typically available from fixed sensors (e.g., see Rose 2006 and Naranjo et al 2010). Recently, Google’s “Maps for Mobile” combined location data crowdsourced from GPS-equipped mobile phones with traditional sensor infrastructure to overlay road maps with congestion information on arterial roads (Weitz 2009) and the Mobile Millennium Project’s Mobile Century field experiment demonstrated the feasibility of traffic monitoring systems based on mobile phone probes (Herrera et al. 2010). Researchers also explored the feasibility of using commercial transport fleets of vehicles, such as taxis, in estimating traffic conditions on arterial roads—e.g. Herring et al. (2010) analysed arterial travel time distributions using a Hidden Markov Model technique applied to sparsely observed probe taxi vehicles and Braxmeier et al. (2009) studied positions and velocities obtained from a fleet of 300 test GPS-equipped vehicles by regarding the recorded velocities as realizations of a random velocity field sampled at selected points only.

Many public transport fleets equipped with automated vehicle location (AVL) systems use GPS to collect probe data and support RTI systems and the potential of using buses as probes has been studied recently. For example, Bertini and Tantiyanugulchai (2004) and Chakroborty and Kikuchi (2004) studied transit buses as traffic probes by investigating to what extent travel characteristics of bus vehicles are related to those of general traffic. Pu and Lin (2008) and Pu et al. (2009) investigated the effect of bus-specific operations and behaviour on bus probe performance as well as interrelation between bus and car speeds; these studies showed that using buses as probes to detect general vehicle traffic conditions could form a real-time traffic monitoring mechanism in an urban advanced traveller information system. Notably, in all these test

studies the control characteristic was chosen to be travel time and bus probes' locations were assumed to be recorded at a relatively high sampling rate. Recently Bejan and Gibbens (2011), building on their previous work with other authors on recovering speed information from sparse bus movement data (Bejan et al. 2010), showed how to infer traffic conditions using this kind of data and taking into account specifics of bus behaviour and road infrastructure usage. These authors demonstrated that with exception of the bus data all other necessary and relevant information can be usefully extracted from an open map, such as OpenStreetMap. This information can be used in providing characterisation of traffic by a local 'per route' description as well as by a global 'network-wide' one. In what follows we discuss advantages and disadvantages of using bus data and provide a brief summary of the methodology of using sparse bus data in inferring urban and suburban roads' traffic conditions and their dynamics highlighting the role of OSM data in this analysis.

Since each bus travels through traffic, that is exists in traffic conditions, bus probe data can be used to characterised the behaviour of traffic on the road network—this approach regards buses as 'particles' moving in the whole vehicular flow. In addition, buses can give good coverage of the main city roads, including arteries (Bejan et al. 2010), and they operate throughout the day and evening, that is when traffic conditions are of most interest. However, there are several drawbacks with using location data from buses alone and these drawbacks are related to the fact that buses may use roads in a way different from other vehicles. Specifically,

- (i) buses spend time at bus stops, and
- (ii) roads may have bus lanes, traffic lights, barrier or bollard control systems which may be used to give buses a higher or even absolute priority.

A typical concern with regard to the case (i) is how to efficiently detect and evaluate bus stop dwelling times and how to tell whether the slowdown near a bus stop is actually due to operating the bus stop and not due to traffic. To mitigate or remove these effects behaviour of buses in vicinities of bus stops can be analysed and taken into account by adjusting the mapping between bus speeds and road network accordingly. An additional assumption here is that outside bus stop neighbourhoods buses largely exist in traffic conditions. Although factors like those listed in the case (ii) represent a serious limitation in using buses as probes in assessing traffic conditions in certain areas of a road or city, the detailed knowledge about the specifics of the road infrastructure and its use by buses and other road users greatly assists in analysing and mitigating associated effects. In particular, the OpenStreetMap data set has a great potential in keeping and accessing such knowledge.

Our use of bus data for assessing traffic conditions and congestion is based

on the notion of the velocity field—a spatio-temporal function defined on the road network and changing in time with values which are to be interpreted as velocities. We construct an approximation to the velocity field by mapping recovered bus speed profiles onto the linear segment representation of the road network (we use the OSM data to extract the road network and keep the OSM linear road segmentation unchanged). The speed profiles are adjusted, where necessary, by (i) making a specific choice of the transformation used to map the bus speed profiles onto the road network (this transformation can be thought of as a way of summarising observed buses’ behaviour) and (ii) accounting for the uniqueness of bus behaviour as discussed above.

We discuss now how speed profiles are recovered from the probe data and use an illustration diagram presented in Figure 5 for this. We first apply map matching to a bus trajectory and perform monotonic spline interpolation to the cumulative distances travelled (step A); red stars in the upper panel correspond to probes and the solid curve is the fitted monotonic spline. Analytically differentiating this spline we derive a continuous approximation to the bus speed (steps B and C). Shown as a black solid curve in the middle panel, this is a good approximation to a smoothed version of the real speed profile shown as a thin grey line in the same panel, taken from “HiRes” traces collected manually as discussed in Section 2.1. We then detect behaviour of the bus in the proximity of a bus stop using the concept of a “local time” (step D). The local time profile corresponding to a bus trajectory characterises the progression of the bus along its route and measures how long that bus spent in the small neighbourhood of each point. When plotted, as in the bottom panel of Figure 5, the spikes indicate locations where the bus spent relatively longer times, whereas long and deep valleys identify parts of the journey that were passed with higher speeds.

After processing a set of trajectories, we apply a certain velocity field transformation⁴ to the restored speed profiles (the result of the differentiation in step C) to summarise and map speeds back to the road network (step E). The utility of using the velocity fields can be seen by an example. In the early morning of Tuesday 22 June 2010 a lorry caught fire on the A14 main road north-west of the city of Cambridge, England. Major disruption followed the incident with adverse consequences across Cambridge. Figure 6 shows the velocity field mapped to the road network based on snapshots taken at 10 am on 15 and 22 June 2010. On the day of the lorry fire, severe disruption has spread widely across the city and the figure demonstrates a dramatic effect of this accident resulting in an overall substantial decrease in average traffic speeds right across the city.

Not only is the OSM database used in this analysis to obtain the description

⁴See Bejan and Gibbens (2011) for details on the choice of velocity field transformation.

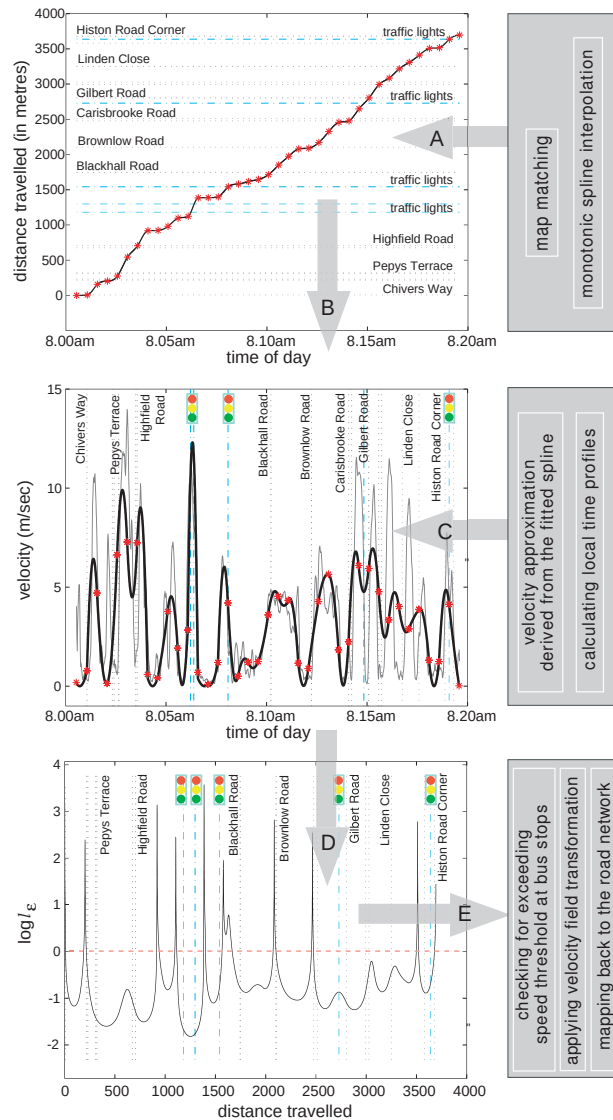


Figure 5: This diagram illustrates how speed profiles are restored from sparse bus data and mapped back to the road network. This is shown for the trajectory of a bus journey made along Histon Road in the morning hours on 23 February 2010, inbound. Details on each of these steps in this diagram can be found in the text.

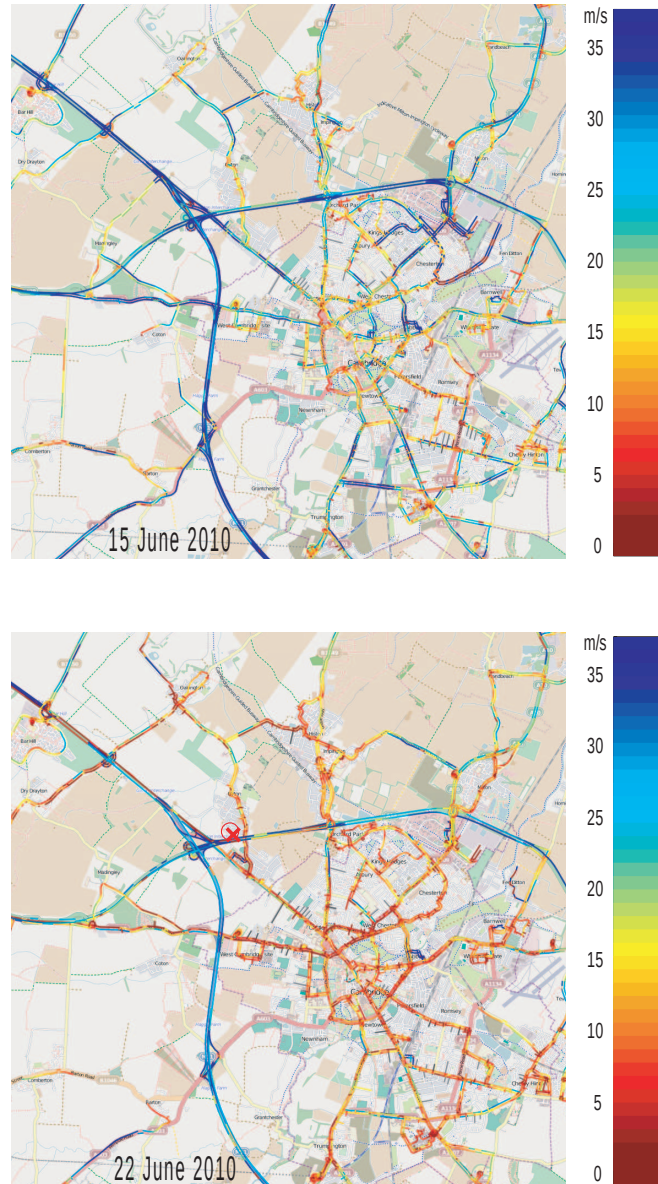


Figure 6: Snapshot visualisation of the velocity fields obtained for the area of Cambridge for two Tuesday: 15 June 2010 (top) and 22 June 2010 (bottom). The velocity fields were evaluated using the average velocity table transformation $f_{\omega}(t; \text{seg}^{(i)})$ at $t = 10$ am using the window length $\omega = 30$ min. This visualisation uses the velocity-to-colour correspondence (presented to the right of the plots) and shows the city gridlock effect of the lorry fire accident (marked by a circled red cross in the lower snapshot) occurred on 22 June 2010. For details on this accident see Brown (2010).

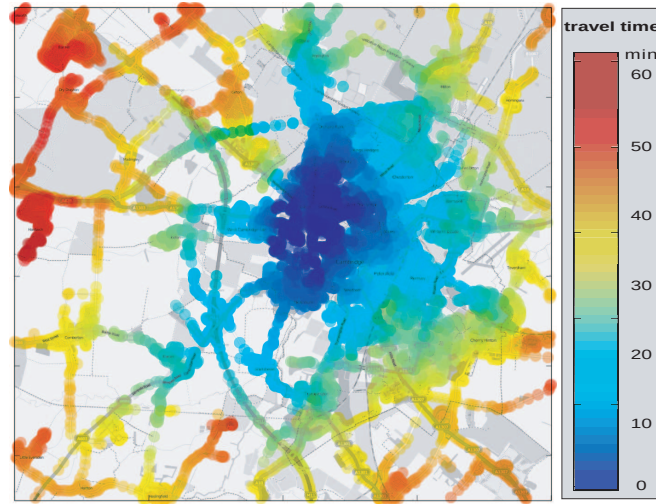


Figure 7: This figure represents a travel-time heat map for the area of Cambridge and surrounding villages obtained for 22 June 2010 when a major early morning traffic accident on A14 affected traffic across the city for hours ahead. Each node of the road network is marked by a colour corresponding to the time which a vehicle traveller would have needed if he or she were to arrive to central Cambridge (Cambridgeshire County Council) by 10 am. This figure is reproduced from Bejan and Gibbens (2011).

the road network, but it is also used to obtain information on the location and direction of bus stops, traffic lights, bus lanes as well as description of the bus routes. Finally, OSM maps are also used for illustration purposes.

Representing scalar fields on road networks and changing dynamically in time, velocity fields implicitly determine travel times from a point to point on the map. We conclude this section by presenting an example of using velocity fields based on this observation.⁵

Figure 7 depicts a coloured travel-time heat map plot where the colours are used to reflect how much time a road traveller (using a vehicle of a general kind) would take to get from a particular point on the road network to central Cambridge arriving at time $T = 10$ am on 22 June 2010: cooler colours are used to denote shorter journeys, whereas warmer colours are used for travel which would have taken longer. This plot was obtained by applying a label-correcting Dijkstra-type algorithm (e.g. Nannicini 2009) to calculate point-to-point shortest time paths on a road network with time-dependent inter-node travel times and suitably reverting the temporal orientation of the weighted network.

⁵This example is taken from Bejan and Gibbens (2011).

This example was inspired by a recent set of diagrams by [MySociety.org](http://www.mysociety.org) in which colours and contour lines are used to show how long it takes to travel between one particular place and every other place in the area, using public transport.⁶ However, the approach described above in building such maps uses traffic-dependent velocity fields reflecting the spatial and temporal aspects of road congestion, whereas MySociety's heat maps are traffic-independent ones, as only transport maps and timetables were used in producing them.

4 Discussion and Concluding Remarks

In this review we have shown how sparse location data from a fleet of public transport buses can be effectively used in evaluating traffic behaviour in urban and interurban areas. A great deal of useful information in our use of bus data can be derived from existing data which often provide good coverage of the urban road network and are already collected and used by transport operator or local authorities, that is the required infrastructure is often in place.

We have used the concept of velocity fields to characterise road traffic, classify its congestion levels and provide a means of estimating journey times of a vehicle (not only a bus!) along an arbitrary route of interest within the area of coverage.

Our analysis, which allowed us to use specific bus data in studying traffic as a whole, would have not been possible without additional information about the road network, road infrastructure, information about bus stops and bus routes. We have shown that OpenStreetMap has a great potential in providing such information. Specifically, OpenStreetMap data were used in our analysis for the following purposes:

1. **Road network:** linear segment representation of the road network has been extracted from OSM database and used for map-matching purposes; the natural OSM segmentation of roads has been used in the velocity field analysis.
2. **Bus stops:** information on locations and bearings of bus stops has been used in order to efficiently extract journeys corresponding to a route of interest from a large data set of sparse bus data. This was done using the journey extraction technique based on the concept of bus stop gates.
3. **Bus routes:** description of bus routes has permitted to identify a subgraph of the road network giving a set of roads which are most likely to be used

⁶mySociety, Travel-time Maps and Their Use. Online: <http://www.mysociety.org/2006/travel-time-maps>. Accessed 20th June 2011.

by buses; this subgraph has helped to reduce uncertainty in (i) snapping of the bus data to the road network and (ii) map-matching process.

4. **Bus lanes, barriers and bollards:** these have been used to obtain knowledge on the parts of roads which may be used by buses in an exclusive way and thus to restrict the analysis to other parts of the road network.
5. **Maps:** these were used for illustration purposes (Figures 1, 6 and 7).

We have also discussed what difficulties one may encounter when using and relying on this additional information extracted from OSM database: for example, inconsistencies in tagging bus stops and identifying their bearings may be an obstacle for making the journey extraction procedure based on bus stop gates automated; slow or no reflection of any changes in bus routes may also affect the performance and validity of the velocity field analysis. These issues need to be studied further.

The analysis on road traffic characterisation and evaluation of congestions levels presented in this review were performed using MATLAB. The author's experience of dealing with OSM data in this mathematical package is summarised in Bejan (2011).

Finally, it is important to mention that the velocity field analysis equally applies to the processing of both historical data and live data feeds with buffering of the most recent streamed data. Dr David Evans (Computer Laboratory, University of Cambridge) is currently involved in implementing an online real-time version of this analysis (*personal communication*).

Acknowledgements. The discussion and examples presented in this paper are based on previous papers of Bejan and Gibbens (2011), Bacon et al. (2011), Bejan et al. (2010), and are a joint work with Jean Bacon, Alastair Beresford, David Evans, Adrian Friday, Richard Gibbens and Ken Moody. We acknowledge the support of the UK Engineering and Physical Sciences Council (EPSRC) through grant EP/C547632 (TIME). We thank Stagecoach in Cambridge, ACIS and Cambridgeshire County Council for allowing us access to archived and real time bus location data feeds.

References

- [1] Bacon J, Bejan A, Evans D, Gibbens R, Moody K (2011) Using Real-Time Road Traffic Data to Evaluate Congestion. *Lecture Notes in Computer Science* **6875**. Springer.
- [2] Bejan A (2011) OpenStreetMap for MATLAB Users. Technical Report (including support functions). To appear on *MATLAB Central* <http://www.mathworks.co.uk/matlabcentral/>.
- [3] Bejan A, Gibbens R (2011) Evaluation of Velocity Fields via Sparse Bus Probe Data in Urban Areas. To appear in *14th International IEEE Conference on Intelligent Transportation Systems*, Washington DC, USA, October 5-7. Preprint available at <http://www.cl.cam.ac.uk/~aib29/itsc2011.pdf>. Accessed 20th June 2011.
- [4] Bejan A, Gibbens R, Evans D, Beresford A, Bacon J, Friday A (2010) Statistical Modelling and Analysis of Sparse Bus Probe Data in Urban Areas. *13th International IEEE Conference on Intelligent Transportation Systems*, Madeira Island, Portugal, IEEE Intelligent Transportation Systems Society (September 2010): 1256–1263.
- [5] Bertini R, Tantiyanugulchai S (2004) Transit Buses as Traffic Probes: Use of Geolocation Data for Empirical Evaluation. *Transportation Research Record* **1870**: 35–45.
- [6] Braxmeier H, Schmidt V, Spodarev E (2009) Kriged Road-traffic Maps. In *Interfacing Geostatistics and GIS*, Springer: 105–119.
- [7] Brown R (2010) A14 Blaze Brings Exam Chaos. *Cambridge News*. <http://www.cambridge-news.co.uk/Home/A14-blaze-chaos.htm>. Accessed 20th June 2011.
- [8] Chakroborty P, Kikuchi S (2004) Using Bus Travel Time Data to Estimate Travel Times on Urban Corridors. *Transportation Research Record* **1870**: 18–25.
- [9] Weitz J (2009) Arterial Traffic Available on Google. *Google Inc.* <http://google-latlong.blogspot.com/2009/08/arterial-traffic-available-on-google.html>. Accessed 20th June 2011.
- [10] Haklay M (2010) How Good is Volunteered Geographical Information? A Comparative Study of OpenStreetMap and Ordnance Survey Datasets. *Environment and Planning B* **37**(4): 682–703.

- [11] Haklay M, Weber P (2008) OpenStreetMap: User-generated Street Maps. *IEEE Pervasive Computing* **7**(4): 12–18.
- [12] Herrera J, Work D, Herring R, Ban X, Jacobson Q, Bayen A (2010) Evaluation of Traffic Data Obtained via GPS-enabled Mobile Phones: The *Mobile Century* Field Experiment. *Transportation Research Part C*, **18**: 568–583.
- [13] Herring R, Hoffleitner A, Abbeel P, Bayen A (2010) Estimating arterial traffic conditions using sparse probe data. *13th International IEEE Conference on Intelligent Transportation Systems*. IEEE Intelligent Transportation Systems Society. Madeira Island, Portugal: 929–936.
- [14] Knoop L, Eames T (2009) Public Transport Technology in the United Kingdom: Annual Survey 2008. *Technical Report*, RTIG Ltd. <http://www.rtig.org.uk/web/docs/PDF/Downloads/RTIG-PR010-D001-1.0-Survey-2008.pdf>. Accessed 20th June 2011.
- [15] Nannicini G (2009) Point-to-point Shortest Paths on Dynamic Time-dependent Road Networks. *PhD dissertation*. École Polytechnique.
- [16] Naranjo J, Jiménez F, Serradilla F, Zato J (2010) Comparison between Floating Car Data and Infrastructure Sensors for Traffic Speed Estimation. *IEEE ITSC 2010 Workshop on Emergent Cooperative Technologies in Intelligent Transportation Systems*, Madeira Island, Portugal, IEEE Intelligent Transportation Systems Society.
- [17] Ordnance Survey (2010) A Guide to Coordinate Systems in Great Britain. http://www.ordnancesurvey.co.uk/oswebsite/gps/docs/A_Guide_to_Coordinate_Systems_in_Great_Britain.pdf. Accessed 20th June 2011.
- [18] Pu W, Lin J, Lon L (2009) Real-Time Estimation of Urban Street Segment Travel Time Using Buses as Speed Probes. *Transportation Research Record* **2129**: 81–89.
- [19] Pu W, Lin J (2008) Urban Travel Time Estimation using Real Time Bus Tracking Data. *Transport Chicago 2008*. Chicago, Illinois, USA. <http://www.transportchicago.org/uploads/5/7/2/0/5720074/intelligentbus-pulin.pdf>. Accessed 20th June 2011.
- [20] Rose G (2006) Mobile Phones as Traffic Probes: Practices, prospects and issues. *Transport Reviews*, **26**(3): 275–291.

- [21] The TIME Project: Transport Information Monitoring Environment. Computer Laboratory, University of Cambridge. <http://www.cl.cam.ac.uk/research/time/>. Accessed 20th June 2011.

A Bus Stops in OpenStreetMap: Examples

The following snippets provide examples of how bus stops are currently tagged in the OpenStreetMap. Description of an OSM node may be obtained at <http://www.openstreetmap.org/browse/node/id>, where `id` is the corresponding ID of the node of interest. Map data © OpenStreetMap contributors, CC-BY-SA.

A.1 Bus Stop Description Imported from the NaPTAN Database

The following is the OSM description of a bus stop in Cambridge, UK which has been used as one of the bus stops in defining bus stop gates to extract journeys in our example in Section 3.2. The NaPTAN code of this bus stop is `cmbgadwd` and its bearing is S (South).

```
<node id="533796533" version="2" timestamp="2010-08-09T14:09:40Z"
uid="*****" user="*****" changeset="5444346"
lat="52.238168" lon="0.1100898">
<tag k="highway" v="bus_stop"/>
<tag k="naptan:CommonName" v="Highfield Road"/>
<tag k="naptan:NaptanCode" v="cmbgadwd"/>
<tag k="naptan:PlusbusZoneRef" v="CAMBDGE"/>
<tag k="source" v="naptan_import"/>
<tag k="naptan:Landmark" v="Highfield Road"/>
<tag k="name" v="Highfield Road"/>
<tag k="naptan:Street" v="Cambridge Road"/>
<tag k="naptan:Bearing" v="S"/>
<tag k="naptan:verified" v="no"/>
<tag k="naptan:Indicator" v="opp"/>
<tag k="naptan:ShortCommonName" v="Highfield Road"/>
<tag k="naptan:AtcoCode" v="0500SIMPIO16"/>
</node>
```

A.2 Bus Stop as a Node in the Way Including Direction

The following is an example of a bus stop which is a part of a way and whose direction is explicitly identified by the tag `direction`.

```

<node id="325207058" lat="48.2007396" lon="16.337144"
changeset="8412090" user="*****" uid="*****" visible="true"
timestamp="2011-06-12T06:33:19Z" version="7">
<tag k="name" v="Urban-Loritz-Platz"/>
<tag k="highway" v="bus_stop"/>
<tag k="railway" v="tram_stop"/>
<tag k="direction" v="forward"/>
</node>

```

The most recent update in the description of this bus stop (as of 20th June 2011) has been made with the following comment: *“Fixing directions of stops on tram line 49 (and others nearby) in Vienna, Austria”*. This bus stop is a node of the following ways and relations: Märzstraße (24867720), Relation Linie 9 Ri. Westbahnhof (1306732) (as stop), Relation Linie N49 werktags Ri. Oper (1306478) (as stop), Relation 296962 (as forward_stop). There is no ambiguity as to what is the direction of travel of buses which stop at this bus stop, but the coordinates of the bus stop in this or similar description do not necessarily coincide with the actual location of the bus stop.

A.3 Bus Stop Beside the Road with no Information on Travel Direction.

The following is an example of a bus stop description in which the bus stop is mapped beside the road and there is no information which would unambiguously determine the direction of travel at this bus stop.

```

<node id="287168106" lat="52.2114569" lon="0.0912475"
changeset="4648617" user="*****" uid="*****"
visible="true" timestamp="2010-05-09T13:38:31Z" version="2">
<tag k="highway" v="bus_stop"/>
<tag k="note" v="Computer display for bus times.
                Not sure which buses stop here."/>
<tag k="source" v="survey"/>
<tag k="shelter" v="yes"/>
</node>

```

This bus stop node is not a member of any road way or relation.

A.4 Bus Stop as a Node in the Way with no Information on Travel Direction

The following is an example of a bus stop which is a part of a single bidirectional road way. The actual direction of buses which stop at this bus stop is

opposite to the direction which could have been inferred taking into account the node order in the sequence of nodes of this road way. Thus, without additional tagging it is impossible to determine the bearing of this bus stop. This bus stop is in Berlin, Germany and it can be seen on Google Maps with Street View.

```
<node id="26906141" lat="52.4363783" lon="13.2732836"
changeset="4653955" user="*****" uid="*****" visible="true"
timestamp="2010-05-09T20:37:56Z" version="4">
<tag k="name" v="S Sundgauer Strasse"/>
<tag k="highway" v="bus_stop"/>
<tag k="network" v="VBB"/>
<tag k="operator" v="BVG"/>
<tag k="ref" v="104155"/>
</node>
```

This bus stop node makes part of the following ways: 27472475, Sundgauer Strasse (41764794), Sundgauer Strasse (26005183), 80762920, Relation Buslinie 184 (63718). Notably, the two road segments of which this bus stop is a joint endpoint enter the description of the relation 63718, which describes the bus route 184, without indication on the direction of travel; the order in which these segments appear, however, is opposite to the actual direction of buses which stop at this bus stop.

Using Open Street Map (OSM) for logistics and planning: A case study

Ben Abelshausen
ben@14k.be - www.14k.be

12 june 2011

1 Introduction

Routing on Open Street Map (OSM) has been possible since the beginning. An advanced use case of routing is considered, more specific the routing requirements of a delivery company handling over 150.000 customers every day. About 1800 different delivery rounds are used on a daily basis in order to organize delivery.

Calculation and optimisation of these rounds is a complex and difficult problem. The routing requirements are diverse and complex and the actual deliverers vary in type of vehicle that is used, the amount of time available each day and efficiency of work. Customers can be individuals requiring one product but also companies requiring larger quantities of products delivered at once. Therefore routing has to be fast, efficient and flexible and has to be able to handle a number of different situations. The goal is meeting all these requirements and getting usable realistic results to enable a more efficient delivery system.

After listing the requirements in detail and based on this realistic case [1], a proposal for a generic implementation of routing on OSM will be suggested. The calculation of delivery rounds is done based on research, a pragmatic approach, and specific test cases. A genetic algorithm (GA) will be implemented to search for a feasible solution based on a defined cost function within certain constraints. The result of this case study, the actual software, can be found at <http://sourceforge.net/projects/osmsharp/> [2].

2 Solution requirements

The requirements consist of two main parts. The first being the regular routing requirements specifying the kind of routing options that should be available when calculating regular point-to-point routes. The result of these

calculations will be a measure t_{ij} of time between two customers c_i and c_j . The second part of the requirements, based on the calculations of the times t_{ij} , are the timing specifications of the required delivery rounds and their parameters. Each round will have to meet upper and lower bound requirements.

2.1 Routing requirements

First of all the routing requirements describe what kind of routing options should be available. Some important aspects are the choice of vehicle, the resulting variables, and the total distance and time. The routing algorithm will have to calculate travel distance and more important, travel time t as accurate as possible.

2.1.1 Choice of vehicle

A number of different vehicles are possible: Bike, Motorbike, Car and Truck. Some deliveries can even be done on foot. Attached to these are varying maximum speeds and types of routes that can be taken. A general system will have to be developed allowing for almost any vehicle type.

2.1.2 Calculated variables

When routing, several variables need to be calculated to obtain deliverer fees and costs for the company.

Fuel price and fuel consumption A good heuristic needs to be found to get good estimates of fuel consumption. Given the current fuel prices the actual cost needs to be calculated.

Total distance and time The total distance d_k and time t_k of a route k need to be known.

2.1.3 Road network

The road network is in constant movement, usually due to road construction, detours or events that block roads or even entire areas of the road network. These obstructions also need to be taken into account when delivering products. Calculating routes that ignore this can be completely useless in the field and result in a failed or late delivery.

Blocked roads or areas When an area is blocked for whatever reason and for some vehicles a different route needs to be calculated. The option should exist to continue on foot to make the deliveries inside the blocked areas. In case of road constructions the travelling time can be increased or

the road can also be unreachable. The routing algorithm needs to take these variables into account and still serve as many customers as possible.

Time-constraints: start-end date and start-end time All these variations in the road network can have start and end dates as well as start and end times.

2.2 Optimisation of a new area/re-optimisation of an existing area

Optimisation of a new area with new customers entails calculating K routes and matching this to an amount of K vehicles serving the customers as efficiently as possible. Some constraints are set on the resulting rounds based on vehicle type and the requirements of the company paying the deliverers.

Time-constraints: upper and lower bounds t_{max} and t_{min} Each round k has a minimum and maximum amount of time available for deliveries. The maximum is trivial; the minimum serves to prevent deliveries with only a few customers, as no deliverer will accept work on those rounds. There are always some fixed costs related to organizing a delivery round and many small rounds would yield very inefficient results.

Variable delivery times per customer t_{d_i} The time it takes to deliver a product or products to a given customer c_i can vary greatly depending on the physical situation at the customer location and the amount or types of products being delivered. A variable delivery time t_{d_i} at the customer level is required to be able to model this situation.

Formal Definition A more formal definition of the problem can be stated using a cost function. First a route cost t_k for each round or vehicle $k \in \{0, \dots, K\}$ is defined in equation 1 using a variable $x_{k_{ij}}$. If $x_{k_{ij}} = 1$ both customers c_i and c_j are part of round k and customer c_i is served just before customer c_j , $x_{k_{ij}} = 0$ if not. t_{ij} is the time it takes to go from customer c_i to customer c_j .

$$t_k = \sum_{i \in k} \sum_{j \in k} x_{k_{ij}} t_{k_{ij}} + \sum_{i \in k} t_{d_i} \quad (1)$$

The total cost, equation 2, describes the cost t_{total} of the delivery to all considered customers c_i .

$$t_{total} = \sum_k t_k \quad (2)$$

This cost function would not fit requirements without the constraint in equation 3.

$$\forall k : t_{min_k} \leq t_k \leq t_{max_k} \quad (3)$$

3 Solution

Finding a solution that fits all these requirements is based on various previous research [3] [4] [5] on the Multiple Travelling Salesman Problem (MTSP) and the Vehicle Routing Problem (VRP) and inspiration was generated from [6].

Before starting the optimisation algorithm the weights t_{ij} need to be known. The first step will be routing and calculating all these weights using OSM. Secondly an algorithm will be suggested to optimise the delivery rounds.

3.1 Using OSM data

A typical disadvantage when considering usage of OSM data is data quality. Past studies have shown that the quality of data obtained from the OSM project can be compared to that of commercially available maps, especially in densely populated areas in Europe [7] [8]. A number of existing routing applications have been developed already based on OSM [6] [9] [10] [11]. The situation in Belgium however, where the company in this case study is located, is not yet at the level of, for example, Germany [7].

A clear advantage compared to commercial maps is the reduced cost when using OSM and the freely available tools and editors that are focussed on allowing users with no knowledge about map building or Geographic Information Systems (GIS) edit maps. The fact that editing the maps can be done by anyone can also be a great asset when an error or inconsistency is discovered. The reduced quality in certain areas in Belgium can be improved using the local knowledge of the deliverers of the company in this case study. This way the company will also contribute to OSM and become part of the community.

After editing the map the data used for routing has to be updated immediately. It is not acceptable to have a waiting period of more than a few seconds before the new data can be used. The routing engine in this case study is not only used to reorganize but also to optimise existing rounds manually by adding and removing customers. This will prevent a solution based on pre-processing the routing data.

Structure

Data in OSM is organized in a very straight forward manner. There are only three main data types:

Nodes An object with a longitude and latitude

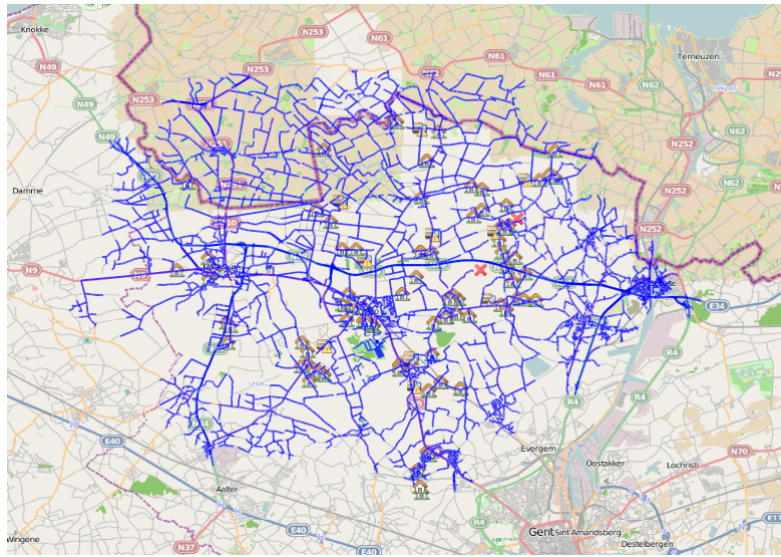


Figure 1: Dykstra algorithm visualisation

Ways A collection of nodes linked together in an ordered list

Relations A collection of relation members linking together Nodes, Ways and Relations in an ordered list

All the positional data is described using Nodes but all other information is described through tags, small key-value collections that can exist on any object describing its properties.

3.2 Routing on OSM

Calculation of the weights is done by the Dykstra shortest path algorithm [12]. Starting at a source customer c_i and stopping only when routes to all other customers have been found. One run will calculate all weights t_{ij} for one customer c_i . Repeating this for every customer will result in all costs t_{ij} . These calculations can be optimized by only calculating weights to neighbours that are relatively close when considering the total number of customers. Feasible routes will only put together customers relatively close to each other.

Basic routing on OSM data can be implemented quickly because the data is already structured as a multi-attribute graph. An interpreter, figure 2, is used to interpret the data and to filter out data that is not part of the road network. No pre-processing is done and the algorithm can run on any live source of data.

There are a few difficulties however in extracting the road network from the data when considering turn restrictions and implementing turning weights.

Turn Restrictions

A turn restriction is a turn in a road network that one is not allowed to take. Turn restrictions in OSM are implemented by using the Relation object between Way and Node objects [13].

Turning Weights

Turning weights, in short, are weights applied to turns. This notion is introduced to create a route that includes more convenient turns compared to inconvenient turns. In most parts of Europe turning right is more convenient compared to turning left.

Turning weights have a similar problem as turning restrictions. Dijkstra's shortest path algorithm cannot handle weights on vertices. The approach is to abstract the turning weights from the routing engine by allowing the interpreter to return a slightly adjusted version of the original graph including new edges and nodes representing the turning weights.

Interpreter

The interpreter will have a crucial role in calculating the best route and will determine what kind of routes will be calculated by the routing engine. The interpreter will be used to implement the following:

- Turning weights
- Turn restrictions
- Vehicle types
- Road network variations

This is achieved by abstracting the actual data from the routing engine and presenting a *virtual* graph by interpreting the data based on vehicle type, turn restrictions and turning weight settings. The interpreter can also be used to implement variations in the road networks, for example road works, blocked roads, etc... How the interpreter interacts with other components can be seen in figure 2.

Turning weights will be handled for example by splitting nodes with weights as shown in figure 3. Splitting of nodes will only occur when a weight needs to be applied.

4 Optimisation of a new area/re-optimisation of an existing area

The solution to this problem exists in minimizing the cost function in equation 2. A genetic algorithm (GA) is implemented to search for a correct orga-

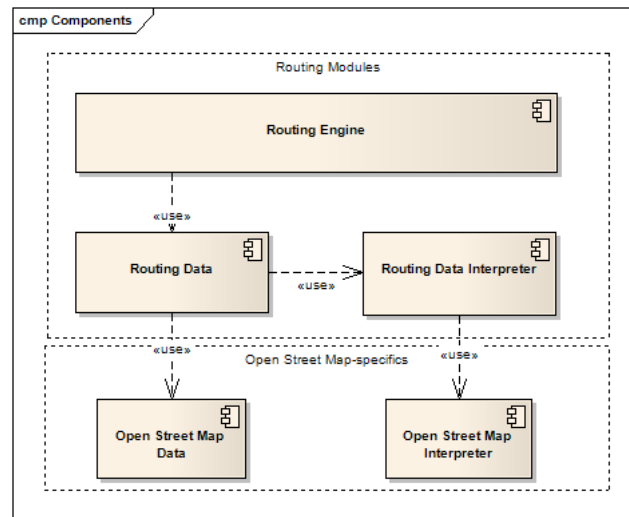


Figure 2: Open Street Map data interpreter

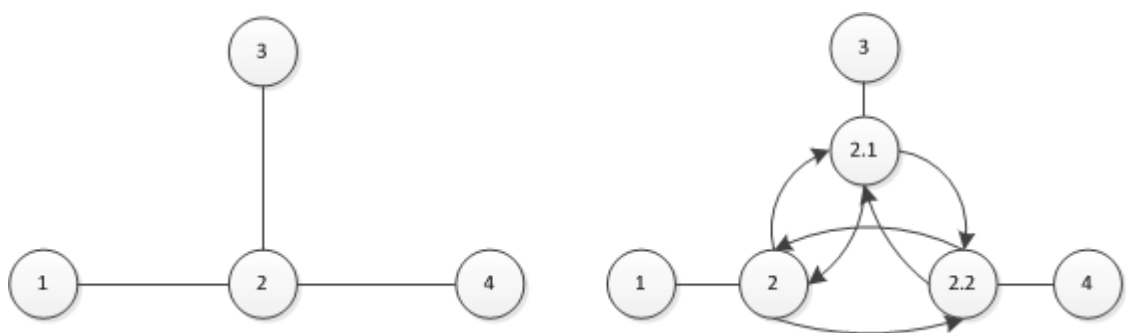


Figure 3: Turning weight handling in the interpreter

nization of rounds and customers to minimize this function [14]. A GA mimics the process of natural evolution in order to implement a search heuristic. Using selection, mutation and crossover operations a population of individuals, or solutions, will be directed to evolve into better individuals. A fitness evaluation is implemented to evaluate the quality of individual solutions and to allow individuals to compete for crossover and survival into the next generation.

4.1 Fitness

Fitness evaluation is one of the more crucial parts of the optimisation algorithm. Implementing conflicting goals in the fitness calculation could lead to an inefficient evolution of the population or even diverging solutions.

A new notion of target time t_{target} is introduced in equation 4.

$$t_{target} = \frac{t_{max} + t_{min}}{2} \quad (4)$$

The goal will be set to generate rounds as close to the target time t_{target} as possible and at the same time to minimize the travelled time t_{total} . This would mean a minimisation of the number of vehicles k as a second objective to the target time t_{target} .

This should all be reflected in the fitness calculation; fitter being individuals that closely represent this goal. Because there are two distinct but sometimes conflicting goals in this case, the target time t_{target} and the travelled time t_{total} , it is important to make sure that the evolution of individuals take into account both these goals. This is done by discretising the measure of closeness to the goal of the target time t_{target} into C categories. This is done in such a way that individuals with similar target time quality can compete with regards to travelling time t_{total} .

Each individual will have a few metrics calculated.

t_{target} categories The size of the categories is calculated in equation 5.

$$C_{size} = \frac{t_{target}}{C} \quad (5)$$

By using equations 6 and 7 a minimum and maximum category metric is calculated.

$$C_{big} = \max \left\lfloor \frac{t_k - t_{target}}{C_{size}} \right\rfloor \forall k \quad (6)$$

$$C_{small} = \max \left\lfloor \frac{t_{target} - t_k}{C_{size}} \right\rfloor \forall k \quad (7)$$

Final fitness with regards to t_{target} will be calculated in 8.

$$C_{fitness} = C_{small} + C_{big} \quad (8)$$

Fitness Fitness evaluations when comparing individuals a and b will be conducted using a few separate steps:

Constraints Individuals a and b are considered equal in this stage when the constraint in equation 9 is met. An individual a will be considered fitter compared to individual b if $C_{fitness_a} < C_{fitness_b}$ but only when the constraint in equation 9 is not met. This will allow the population to evolve to feasible solutions before competing in the total time t_{total} . When $C_{fitness_a} = C_{fitness_b}$ the number of rounds k will be used to break the tie. This to prevent an explosive increase in the number of vehicles when decreasing t_{total} without bounds.

$$\forall k : t_{min_k} \leq t_{R_k} \leq t_{max_k} \quad (9)$$

Total time If both individuals are feasible, meaning valid in the constraint 9, the total time t_{total} will be the only metric for fitness. The number of vehicles will be secondary to this goal.

4.2 Initialisation

The initialisation of the individuals in the population is done in two stages. First an estimate is calculated to get a reasonable number of vehicles K_{init} , secondly the population is initialized creating individuals all having the same number of vehicles K_{init} .

Best-placement In all stages of the solution a best-placement algorithm is used, placing a customer in a position where it will incur the least increase in time. When placing a list of customers, the customer with the least increase in time will be placed first, repeating the process until all customers are placed.

Vehicle Estimation The estimation algorithm starts by choosing a random customer and placing it in a first round for the first vehicle. The best-placement algorithm is used for selecting a customer c_i that causes the least increase in time t_k and placing it into the current round k . When the required target time t_k is exceeded a new random customer is selected from the customers still left and a new round is created. This will lead to a collection of rounds close to the target time t_{target} and will result in a fair, but high, estimate of the total number of vehicles or rounds K_{init} needed in a good solution.

Population generation The second part of the algorithm generates solutions based on the same number of vehicles K_{init} . First, each round of

each vehicle is initialized with a random customer. Best-placement is used to place the remainder of the customers in the generated rounds.

4.3 Crossover

The crossover operation is applied to individuals selected from the population. The fittest individuals will have a higher probability of being selected compared to individuals with lower fitness. A tournament-based selection algorithm [15] is applied to select fit individuals for reproduction or crossover.

The crossover operation is implemented by selecting random rounds from each parent individual and putting them into the new individual. Only rounds with customers not yet in the target can be selected. When no more rounds can be selected from either parent, best-placement is used to place the remaining customers while at the same time taking into account the target time t_{target} . All the remaining customers if any will be placed in additional rounds using the same initialisation algorithm.

4.4 Mutation

The mutation operation will make small modifications, or mutations, to the individuals. Several different mutation operations are implemented to reach different goals. Selection of which operators to use will be based on configurable probabilities which can be dynamically adapted while the algorithm is running. When the individuals are already feasible the only target that needs to be taken into account is the total time t_{total} . The probabilities will be adapted according to the fitness of the individual being mutated. Three categories of mutation operators are constructed, each with a specific goal in mind.

Goal 1: Reduce the size of bigger rounds

Re-divide from big A round is selected with increased probability of selection of a big round. Customers are taken out of the round and redivided to others using best-placement until the selected round has reached the target time t_{target} .

Best in smallest-operator The smallest round k is selected and best-placement is used to select the best customer to add to the round minimizing the increase in t_k .

Take-out operator A round is selected with increased probability of selection of a big round. A customer is taken out of the round and placed in the other rounds based on best-placement.

Goal 2: Increase the size of the smaller rounds

Re-divide to small A round is selected with increased probability of selection of a small round. Customers are placed in the round until the selected round has reached the target time t_{target} .

Take-in operator A round is selected with increased probability of selection of a small round. One customer is placed in the round using best-placement minimizing the increase in time.

Goal 3: Reduce the total time t_{total}

Exchange operator Select two rounds and search for the best two individuals to exchange, minimizing the increase or maximizing the decrease in time.

Round-merge operator Select two rounds and merge them into one round using best-placement.

Round-cut operator Select a round and select a customer. Generate two new rounds by cutting the round at the selected customer.

Round-reduce multiple operator Select three rounds and generate two new rounds out of the customers in the three rounds selected.

5 Results

Results have only been evaluated in the field and have not been compared with other research results. The generated rounds can be used without modification. Calculations of times and fuel consumption could be better and have to be adjusted based on experimental data but these will be minor changes. The routing results have been validated by usage of the same basic algorithm in the daily operations of the company. The algorithm yields satisfactory results at almost no cost using only Open Street Map (OSM) as a source of data.

Calculations to reorganize an existing area have been tested on an area of about $600Km^2$ with 1000 customers. A population of 50 individuals was used with 10% crossover, 70% mutation rate and 10% elitism. The 46 generated rounds are shown in figure 4 after a calculation time of approximately 1h33 minutes on an *AMD Athlon 64 X2 Dual Core Processor 5000+ @ 2.60Ghz with 4Gb RAM*.

The algorithm is clearly better suited when calculating for a large amount of customers evenly distributed geographically. Secluded customers as show in figure 5 will not leave a lot of options organizing different settings of

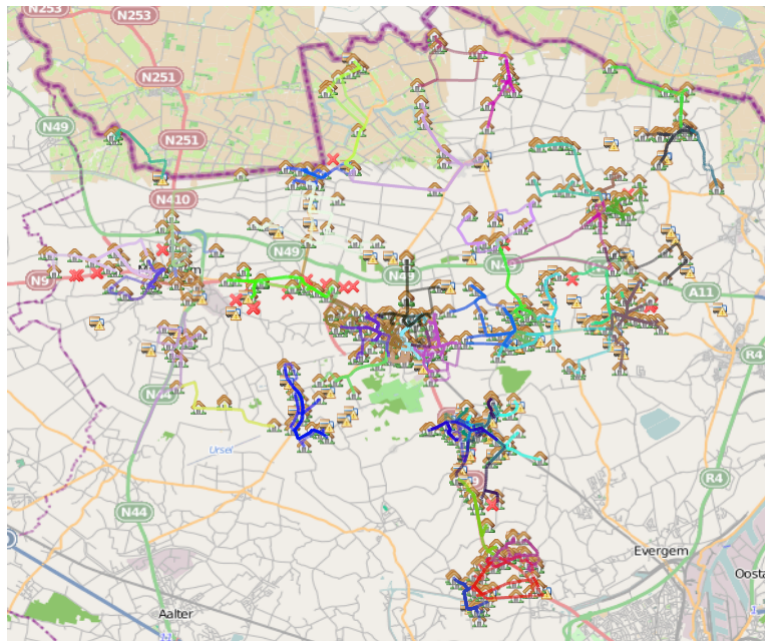


Figure 4: An example of a visualized result

rounds. A solution with upper and lower timing bounds may even be impossible and the algorithm will never find any individuals in a feasible state in this case. An option would be to relax the bounding conditions to allow for more flexibility in timing.

6 Conclusions

The OSM project is an ideal catalyst for this kind of project. Without the availability of raw map data, this project would not have been possible and the company in this case study would never have been able to optimise its operations in this way. Not only the availability of the data itself but also the various tools and the support of an active community lie at the basis of the succes of this project. This case study can be considered another succes story related to the OSM project.

The optimisation of the delivery rounds can be done using only open source software and open data. Something that would not have been possible a few years ago. The possibilities that OSM has generated here also show that the project has reached a certain level of maturity.

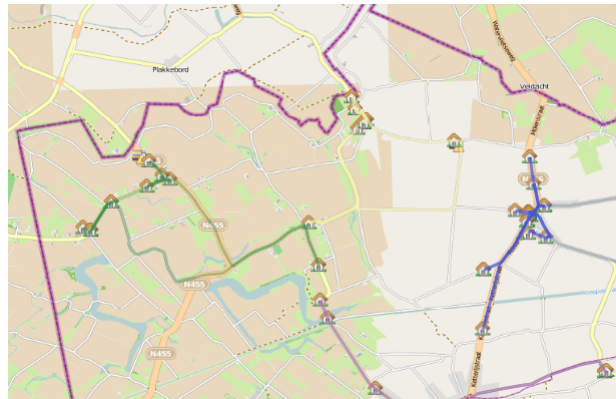


Figure 5: Difficulties exist when dealing with secluded customers

7 Future Work

Comparing the results in this case study to actual research results will be a better evaluation of the algorithms used. This however is not the main goal of the company this software was developed for. Additions to the algorithm need to be made to take into account weight limits for each vehicle type and thus upper and lower bounds on each round. An improvement in calculation time should also be possible.

Improving the routing engine is also possible in the future when the road network quality in Belgium has reached a more stable state. Pre-processing the data will become a valid option and can be used to implement a combination of the current online way of working now with the more performant offline routing.

References

- [1] Deltamedia nv. <http://www.deltamedia.be/>.
- [2] Ben Abelshausen. Osmsharp - open street map library and tools developed in c#. <http://sourceforge.net/projects/osmsharp/>.
- [3] G.B. Alvarenga, G.R. Mateus, and G. de Tomi. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers Operations Research*, 34(6):1561 -- 1584, 2007. Part Special Issue: Odysseus 2003 Second International Workshop on Freight Transportation Logistics.
- [4] Jean Berger and Mohamed Barkaoui. A route-directed hybrid genetic approach for the vehicle routing problem with time windows. *INFOR*, 41:179--194, 2003.

- [5] Marius M. Solomon and Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254--265, 1987.
- [6] F. Graf, H.-P. Kriegel, M. Renz, and M. Schubert. MARIo: Multi attribute routing in open street map. In *Proceedings of the 12th International Symposium on Spatial and Temporal Databases (SSTD)*, Minneapolis, MN, USA, 2011.
- [7] Ina Ludwig, Angi Voss, and Maike Krause-Traudes. A comparison of the street networks of navteq and osm in germany. In Stan Geertman, Wolfgang Reinhardt, and Fred Toppen, editors, *Advancing Geoinformation Science for a Changing World*, volume 1 of *Lecture Notes in Geoinformation and Cartography*, pages 65--84. Springer Berlin Heidelberg, 2011.
- [8] D. Zielstra and A. Zipf. Quantative studies on the data quality of openstreetmap in germany, 2010.
- [9] Simon Nuttall and Martin Lucas-Smith. Cycle streets. <http://www.cyclestreets.net/>.
- [10] Yours, yet another openstreetmap route service. <http://www.yournavigation.org/>.
- [11] Cloudmade routing. <http://developers.cloudmade.com/projects/show/routing-http-api>.
- [12] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269--271, 1959. 10.1007/BF01386390.
- [13] Open street map wiki. http://wiki.openstreetmap.org/wiki/Turn_restrictions.
- [14] John H. Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA, 1992.
- [15] Brad L. Miller, Brad L. Miller, David E. Goldberg, and David E. Goldberg. Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9:193--212, 1995.