



universität
wien

MASTERARBEIT

Titel der Masterarbeit

Real Time Point Cluster Solutions in Web Mapping Applications: An Alternative Approach Using Grid-based Structures

verfasst von

Andreas Kiefer, BSc

angestrebter akademischer Grad

Master of Science (MSc)

Wien, 2015

Studienkennzahl lt. Studienblatt:

A 066 856

Studienrichtung lt. Studienblatt:

Kartographie und Geoinformation

Betreuerin / Betreuer:

Ass.-Prof. Mag. Andreas Riedl

Acknowledgement

This thesis marks the end of a long, exciting and challenging time in my life. Cartography is not only a tool to transfer geospatial data onto a map – it helps to create pictures, imaginations and ideas of how our world looks like. I am glad I could contribute a little to this discipline by developing a small prototype. I would like to use this opportunity to express my gratitude to all those people who helped me during this time.

This work would not have been possible without the everlasting, caring support and help I received during all this time from my lovely wife Maria, she always was there for me and cheered me up when I got stuck.

Also, I want to thank my parents, who always covered my back and gave me the means and support to succeed my studies. Thank you so much.

A special ‘thank you’ goes to my supervisor and tutor, Ass.-Prof. Mag. Dr. Andreas Riedl, whose door always stood open whenever I got stuck with this work. Thank you for your patience and your ideas!

I would further like to thank my former colleagues in the Working Group of Cartography and Geoinformation Sciences, especially Mag. Dr. Alexander Pucher and Ass.-Prof. Mag. Dr. Karel Kriz, who gave me a lot of inspiration and were always available for fruitful discussions.

Last but not least, I would like to say thank you to my study colleagues Hubert Gimpl and Christoph Kubasa, with whom I mastered many projects and spent long afternoons in the science lab. They made studying Cartography a breeze. I wish them all the best!

Abstract

In a data-rich world, the key to knowledge discovery and cognition of relevant information lies in efficient processing and visualization of data. Applied to geospatial information, web maps are a common means to communicate phenomena and relevant information to the user. Since the establishment of the Web 2.0, map mashups have evolved, and are used in many domains, particularly to display Points of Interests (POI) on a map.

A common problem in this application is known as ‘icon cluttering’, which occurs, when too many features cover up the available map space. As a consequence, map legibility suffers, and the information presented on this map cannot be understood anymore. Therefore, generalization procedures have to be applied, which clear up the map space by using aggregation, filtering, displacement or clustering techniques. For the processing of vector-based data in real time, only few implementations exist in web mapping today.

In this thesis, the application of a grid-based clustering algorithm is examined. A prototype implementation was developed to test the feasibility of this approach in two common use cases. The evaluation of these use cases has shown that grid-based structures can serve as an alternative clustering method, especially for thematic mapping purposes and for the visualization of temporal developments.

Key words: Web Mapping, Cartographic Generalization, Feature Aggregation, Clustering, Icon Cluttering, Thematic Mapping

Kurzfassung

In Zeiten von ‘big data’, automatischer Datenerfassung durch hochentwickelte Sensoren, Informationsgewinnung durch ‘Crowdsourcing’, und leistungsfähiger Systeme zur Datenspeicherung ist es von enormer Wichtigkeit, effiziente Methoden zur Informationsverarbeitung und vor allem Darstellung von Informationen einzusetzen. Informationen mit geographischem Kontext werden meist über digitale Web Maps vermittelt. Seit der Etablierung des Web 2.0 werden Karten und ‘mashups’ zu allen möglichen Themen auf Webseiten eingebunden.

Ein allgemein bekanntes Problem stellt dabei die Überlagerung der Karte mit Symbolen dar. Dieses Phänomen tritt auf, wenn zu viele Symbole den verfügbaren Kartenausschnitt überdecken. In der Folge wird die Karte für den Nutzer unlesbar. Eine Lösung ist die Anwendung kartographischer Generalisierungsmethoden, wie Punktaggregation, Selektion oder die Bildung von Clustern. Bis heute existieren jedoch kaum Lösungen, die diese Methoden in Echtzeit im Browser bewältigen können.

In dieser Abschlussarbeit wurde die Eignung eines auf regelmäßigen Gitternetzen basierenden Clustering Algorithmus untersucht. Dazu wurde ein in Echtzeit operierender Prototyp entwickelt, welcher anschließend in zwei Anwendungsszenarien getestet wurde. In einer Online-Umfrage wurde die Eignung auf Gitternetz basierender Clustering Algorithmen untersucht. Die Auswertung der Umfrage hat gezeigt, dass regelmäßige Strukturen unter gewissen Umständen durchaus dazu geeignet sind, als alternative Methode zur Clusterbildung zu dienen. Vor allem thematische Anwendungen und die Visualisierung raum-zeitlicher Entwicklungen können von dieser Darstellungsmethode profitieren.

Schlüsselwörter: Web Mapping, Cartographic Generalization, Feature Aggregation, Clustering, Icon Cluttering, Thematic Mapping

CONTENTS

Preface	1
1 Introduction	3
1.1 Motivation.....	3
1.2 Research Objectives.....	5
1.3 Methodology	6
1.4 Structure of the Thesis	7
1.5 Related Work	8
2 Web Mapping	10
2.1 How Maps Are Provided on the Internet	11
2.1.1 <i>A Brief History of Web Mapping</i>	11
2.1.2 <i>Standards and Technologies</i>	14
2.2 State of the Art Web Mapping	16
2.2.1 <i>Web Mapping APIs</i>	18
2.2.2 <i>Raster-Based Web Mapping</i>	19
2.2.3 <i>Vector-Based Web Mapping</i>	20
2.3 Obstacles and Limitations	22
2.3.1 <i>Quality Issues</i>	22
2.3.2 <i>Icon Cluttering</i>	23
2.4 Common Types of Web Maps	24
2.4.1 <i>Map Mashups</i>	25
2.4.2 <i>Thematic Web Maps</i>	25
2.4.3 <i>Animated Web Maps</i>	26
2.5 Map Design and Composition	27
2.5.1 <i>Cartographic Communication and Data Visualization</i>	28
2.5.2 <i>Design Principles</i>	29
2.5.3 <i>Symbolizations</i>	30
3 Cartographic Generalization	33
3.1 Decluttering of Point Markers	34
3.2 Distance-based Clustering	36
3.3 Density- or Grid-based Clustering.....	37
3.4 Comparison.....	40
3.5 Data Classification	41
3.6 Grid Structures	41

4	Evaluation.....	43
4.1	Heuristic Evaluation	43
4.2	Deductive Tests	44
4.3	Scenarios and Use Cases	44
5	Leaflet.js	46
5.1	The Core Library	46
5.2	Proj4Leaflet Plugin.....	46
5.3	MarkerCluster Plugin	47
6	Use Cases	47
6.1	Identification of Possible Use Cases	47
6.2	Use Case 1: Locating a Bike Station in Paris.....	48
6.3	Use Case 2: Visualization of Temporal Change of Data.....	51
7	Implementation of the Prototype	53
7.1	Requirements and Settings	53
7.2	Design.....	54
7.3	Evaluation of the Performance	59
8	Survey	65
8.1	Goal of the Survey.....	65
8.2	Survey Design	65
8.2.1	Use Case 1	66
8.2.2	Use Case 2	67
8.3	Survey Results.....	67
8.3.1	Survey Audience.....	67
8.3.2	Use Case 1	69
8.3.3	Use Case 2	74
8.4	Limitations.....	78
9	Key Findings and Recommendations.....	78
9.1	Chances and Advantages.....	78
9.2	Restrictions for the Usage	79
9.3	Recommendations	80
10	Conclusions and Outlook	82
	Literature.....	84
	List of Figures.....	89
	List of Tables and Listings	90
	Appendix.....	91

PREFACE

In our daily live maps are all around us – today usually in digital form. Web mapping is omnipresent: On smartphone apps, in online shop-locators of retailers and companies, in social networks, online communities, trip planners and so on. Additionally, in times of *big data* and powerful sensors, more data than ever is collected and available for processing (KEIM et al., 2005). For the understanding of all that available information, it is therefore essential to find a visualisation technique that emphasizes the interesting pieces of this vast pile of data (WOOD et al., 2007).

Maps are a decent way to represent data with geospatial context. Point-based information is usually displayed on web maps using marker representations. In many applications and map mashups, a lot of markers on dense areas can clutter up the available map space and as a consequence, the map is becoming unreadable.

For these situations several point clustering strategies have been developed. The most common one is based on the distance of the markers to each other. A certain amount of markers in a defined area is aggregated into only one point which will be labelled with the number of points it is representing. Popular web mapping libraries like Google Maps, OpenLayers, Leaflet and ESRI Viewer for Flex all use this algorithm (GOOGLE n.d.; OPENLAYERS n.d; MARKERCLUSTER-1; ESRI 2014). This on the fly method offers the advantage of leaving the raw input data of the map unchanged. The workflow of publishing the data therefore is kept straightforward: No filtering or generalization steps have to be done in advance, enabling also non-professionals to make use of that technology.

The downside of this approach, on the other hand, is that a change in the zoom level most often makes the cluster points change their location as the clusters are calculated dynamically for each level. This behaviour not only makes it hard for the user to identify the core of a cluster, it can also complicate the orientation on the map (TREVES 2011).

Therefore, an alternative method of grid-based clustering exists. A grid of equal sized square or hexagonal-shaped cells is laid over the map, and then the frequency of occurring features is counted for each cell. The cells can be coloured according to that results, or the centroid of each cell may be used as a cluster point. The advantage of this method lies in the provided comparability in means of space and time and the clear visual message that is transported by such a representation (JOHNSON 2011, ARNBERGER 1977).

Online- and mobile-applications clearly are the future of maps. Geospatial visualizations can be integrated by a broad audience today, without much theoretical background or expertise needed. Tools and frameworks often are developed rather by computer scientists than by cartographers, resulting in varying adequacy of the data visualization. The discipline of cartography has to contribute to the development of new technologies, integrating the knowledge about data communication, visual variables, human cognition and data visualization into the developing process.

This thesis is devoted to analysing the feasibility of a grid-based clustering approach as a real time implementation for web mapping applications. For that purpose, a prototype implementation was developed and released for public use. It is the sincere goal of this work to contrib-

ute to the diversity of available web mapping products, and to help to increase map legibility with cartographic pleasing mapping tools. By no means can this complex topic be treated thoroughly within the scope of a Master thesis, but the first insights gained by the investigations performed throughout the course of this thesis may help to develop and enhance tools for better maps.

In order to perpetuate a convenient reading experience, a gender-neutral wording was foregone. Throughout this thesis, both genders are addressed equally, and it was attempted to use gender-neutral language as often as possible.

1 INTRODUCTION

This chapter is devoted to giving an overview over the thesis. At first, the motivation of the research is laid out. The problem which will be addressed by this work will be described and a working hypothesis will be devised. Following, the research questions, which will guide through this thesis, will be introduced, leading to the structure of the thesis. With the presentation of related work, the value of this research will be put into context to the contemporary scientific work which has been done so far.

1.1 Motivation

A conventional, printed map usually results from the knowledge, labour and input of a cartographer: Selected map features are displayed in a generalized, visually optimized way according to the design rules of the cartographic discipline to facilitate orientation and perception for the map reader. In other words, a classical map is usually the result of several preprocessing steps and a rather long working process.

With the rise of digital maps and web mapping, especially since the establishment of the so called *Web 2.0*, a broad audience is now enabled not only to consume maps, but also to create their own maps, mashups and the like. Mapping frameworks like *Google Maps*, *Mapbox* or the *ESRI Flex Viewer* make it easy to quickly publish a map showing custom data on a predefined base map. From a cartographic point of view, this can lead to quality issues, as many map creators are unaware of cartographic design principles.

However, prerendered base maps undergo the same cartographic workflow as described above: a scale level depending preselection and generalization of features is usually part of the process of delivering maps. But there is also the thematic part of a web map, which consists of vector based features such as Point of Interests (POI), borders, areas and the like. These features are usually rendered in an unprocessed way, often leading to a cluttered map as shown in Figure 1.

At this zoom level, the point markers touch and overlap each other and hide the underlying base map completely. The map legibility suffers, the information presented on this map cannot be understood anymore, and the map fails its purpose to communicate information to the user.

Although one can argue that an interactive web map enables the user to zoom and pan around, if he wishes to gain more information, this visualization is far from being perfect. It is obvious that cartographic generalization has to be applied in order to preserve the map readability in situations like the one just mentioned.

HUANG & GARTNER (2012, 157) identified two possible solutions for this cluttering problem: Either, the amount of features shown on the map has to be reduced by applying some kind of semantic filtering and thus showing only features which are presumably relevant to the user, or the features on the map are aggregated by identifying clusters. In both approaches, the resulting amount of features is reduced in a way that restores the maps legibility. This thesis will concentrate only on analysis of the latter solution.



Figure 1: Example of suboptimal symbolization resulting in a cluttered map. (<http://en.velib.paris.fr/Stations-in-Paris>)

Cluster methods are widely used in science to determine regions with denser feature aggregations in a large multidimensional space. These methods are also applicable on the two-dimensional map space. For extremely big datasets, the clustering can be quite time consuming, but for typical web mapping applications, the number of features is processable in real time. With increasingly powerful hardware and browser environments, the task of processing huge amounts of features can be performed on the client side in real time as the user is browsing the map.

Until today, there exist only few openly accessible implementations of a real time point feature aggregation method, which are all more or less based on one algorithm: a distance-based method which aggregates points and visualizes them with a marker in the geographical centre of the cluster. Unfortunately, this algorithm is not suitable for several use cases as it lacks for example the ability to visualize temporal developments. Furthermore, the available implementations do not always follow cartographic design principles, resulting occasionally in a suboptimal visualization of data.

This thesis with the title *Real Time Point Cluster Solutions in Web Mapping Applications: An Alternative Approach Using Grid-based Structures* examines the feasibility of a grid-based cluster method for web mapping applications. One goal of this work is to develop and assess a prototype implementation of a grid-based clustering algorithm, which can be used to aggregate point-based features on the fly. As a side product of this work, the prototype will be available to the public. The eligibility of this alternative approach will be investigated by applying several methodologies on this prototype. It is the sincere hope of the author, that the results and insights gained through this work might help to increase the quality of cartographic products on the web, and that this contribution will give users and map makers an alternative tool to map geospatial data.

1.2 Research Objectives

In this chapter, the research objectives of this thesis will be laid out. As already mentioned in the motivation, this work aims to examine an alternative approach to solve icon cluttering problems in web mapping applications with high point information density. Therefore, a prototype implementation of the grid-based method will be developed. It shall be analysed, under which circumstances, and for which use cases and scenarios it is eligible to act as a substitute and supersede the popular distance-based method. The working hypothesis therefore is:

Grid-based cluster approaches can be used in web mapping to improve map readability on web maps with a high, point-based information density, compared to algorithms used in popular mapping libraries.

Several questions have been developed, which will help to solve this hypothesis, and which will be used as guiding research questions, with the first one to be:

- **Which common issues exist, when dealing with large point-shaped datasets in web maps?**

To answer this question, a certain theoretical knowledge about web mapping and the methods available to generalize and visualize (point-based) geospatial data on web maps, is essential. After gathering an overview over the most common problems, it will be shown how these issues can be addressed by answering the question

- **Which clustering methods exist and what are their benefits and shortcomings?**

This question will be answered using literature research and examination of existing popular projects and approaches. For the second part of the question, the results of performance evaluations and the feedback given by participants of a small survey will be evaluated, as well.

Following, the issue of suitable scenarios will be addressed. It has to be expected that no clustering method will be suitable and appropriate for every possible use case. Furthermore, different scenarios demand different solutions. As the feasibility of a grid-based clustering approach shall be analysed, the question to ask is:

- **Where can the application of a grid-based clustering strategy be advantageous?**

Using the answers from this research question, possible use cases will be identified and created. They will be used later to build a small online survey which will focus on map usability and user experience.

The last question is more of a technical nature, as the application of real time clustering solutions makes sense only if the performance of the algorithm is sufficiently fast. This issue is addressed as follows:

- **Where are the limits in terms of computing performance for real time clustering solutions?**

To answer this question, a prototype algorithm will be developed and implemented into the JavaScript library *Leaflet.js*. Using this implementation, a performance evaluation of an existing distance-based algorithm, and the prototype will be performed. Subsequently, the online survey will be created.

With the help of these research questions and the results of the survey, the working hypothesis shall be validated. In the following chapters, the structure of this work will be explained, and the content of the chapters is laid out.

1.3 Methodology

The objectives defined in the preceding chapter will be pursued using three levels of methodology, namely: *conceptual*, *operational* and *implementation* levels. Figure 2 shows a sketch of the research steps.

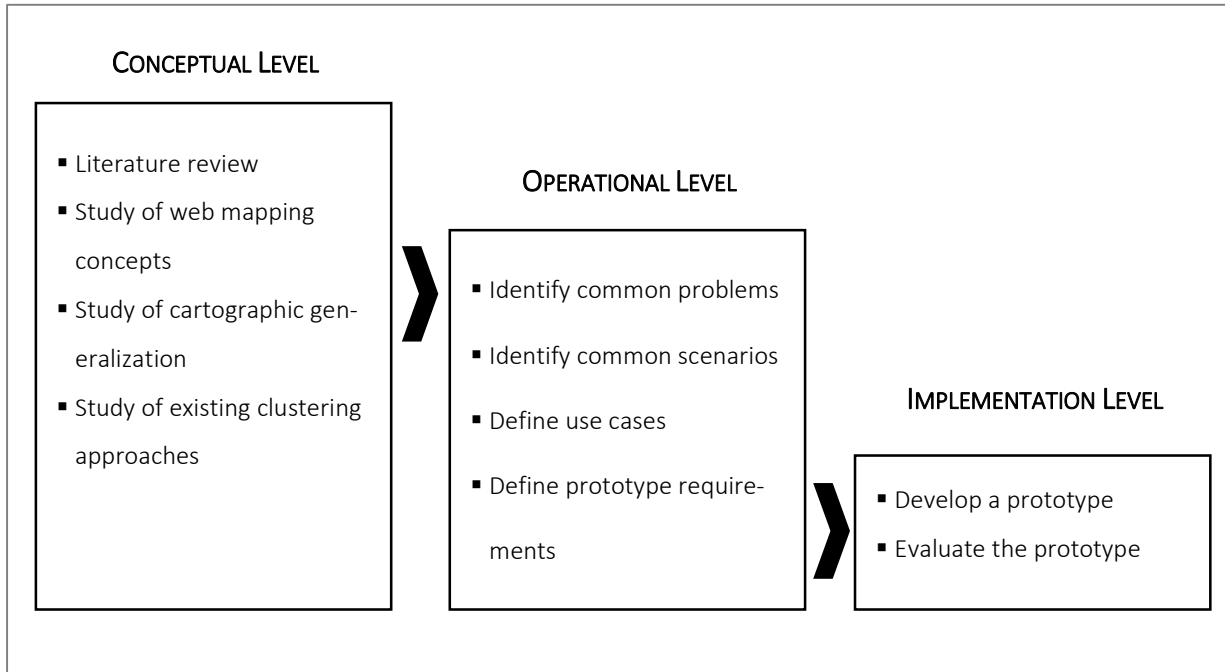


Figure 2: Stages of methodology used in the thesis

On the *conceptual level*, this thesis explores the existing approaches to declutter icon- and point-based symbolizations. The review of common concepts in web mapping, cartographic generalization and point clustering approaches forms the basis of this work, as it leads to the *operational level*.

The main interest in this level is to identify common problems and to further work out typical scenarios for the application of grid-based clustering approaches. From these scenarios, use cases will be derived, and requirements for the prototype will be defined.

The last step, the *implementation level* of this work, encompasses the creation of a prototype that demonstrates the clustering and visualization functionalities identified in the operational level. The prototype combines the two distinct cartographical generalization procedures aggregation and classification with the application of a regular grid structure. As a consequence, the prototype facilitates the cognition of relevant information on the map: The user is enabled to identify, locate, compare and associate geospatial features.

No approach to declutter a map is suitable in every situation, and as a consequence, the evaluation of the prototype will show the feasibility of a grid-based clustering approach.

1.4 Structure of the Thesis

This chapter will give a brief overview over the structure of this thesis. The particular chapters will be introduced in the following paragraphs:

Chapter 1 is devoted to giving an introduction into this thesis. The motivation of this research, as well as the research hypothesis, objectives and the methodology used to solve the hypothesis will be discussed. This first chapter will be concluded with a survey over related research projects.

In **chapter 2**, the theoretical and conceptual backgrounds of web mapping and related fields will be explained. After a short excursion into the history of web mapping, it will be shown, how maps are delivered on the internet today. The limitations and obstacles of web maps will be described as well as the most common types of maps used on the web. A discussion of cartographic communication and design principles will round up this section of the thesis.

In **chapter 3** cartographic generalization procedures will be examined. Available strategies to declutter map space and to create clear, meaningful visualizations will be explained, and methods needed to develop the prototype implementation will be investigated.

In **chapter 4**, methods to evaluate the usability of software and user interfaces will be introduced. Algorithms and software are used to solve real world problems, hence, the development of scenarios and use cases helps to design the software in a way that fits to the needs of the users, best.

As part of the development of a prototype implementation, **chapter 5** will give an introduction to the web mapping library which was used to implement a grid-based clustering solution in JavaScript.

Chapter 6 is devoted to the identification and design of conceivable use cases. The evaluation of the preceding chapters lead to two plausible use cases, which will be used in the following empirical part of the research.

In **chapter 7**, the implementation of the prototype algorithm is documented. First, an overview over the requirements and preconditions is given, followed by the explanation of the actual prototype design. The technical performance of the prototype will be assessed and compared to an existing real time clustering solution. After the implementation in *Leaflet.js* this developed method will be used to create and simulate the use cases in a small survey.

The evaluation of clustering approaches, with a special focus on the newly developed prototype will be subject of **chapter 8**. An online survey was conducted to examine the eligibility of that prototype in the use cases developed earlier. In the first part of this chapter, the design of the survey will be discussed, and in the second part, the results will be presented. This section will be concluded with a critical discussion of the survey.

Chapter 9 will present the key findings of this research. The working hypothesis will be validated, and the chances and benefits, as well as restrictions and downsides of a grid-based clustering approach will be laid out. Furthermore, the insights gained throughout this thesis will be composed and transformed into recommendations for map makers and developers of data visualization tools.

Chapter 10 will complete this research in summarizing the essential outcomes of this work, drawing conclusions and giving an outlook to further possible research directions.

1.5 Related Work

This chapter will give an overview over the research that has been done until the time of writing. In web mapping, only a few clustering methods are widely used today, with the distance-based clustering method being the only one available for real time point aggregation in form of a ready-to-use plugin for popular web mapping frameworks.

Grid-based approaches are implemented using either individually coded, or prerendered solutions (e.g. MARINETRAFFIC-1; DELORT, 2010; FIELD, 2012; ROICK et al. 2011; ROICK et al. 2012). They require expert knowledge and can be implemented only with a remarkable amount of effort. Moreover, there has been done only little research about the *usability* of these different cluster strategies, despite the fact, that the Web 2.0 and therefore map mashups exist already a couple of years.

However, research has been done to solve the problem of marker cluttering, which occurs when too many markers cover up the map screen. In the following section, selected approaches will be presented and discussed.

A comprehensive summary over the available approaches is given by HUANG & GARTNER (2012, 157-175). According to the authors, there exist basically two ways how to approach the issue of marker cluttering: The first solution is to use more or less intelligent, context aware filtering techniques which will reduce the amount of point data on the map to a relevant minimum. The second way is to manipulate the location of the markers in a way that the overlapping of the icons is reduced to a minimum. This effect can be achieved by applying either displacement operators or aggregation operators. However, the focus of their article is set on the examination of the first solution.

BEREUTER & WEIBEL (2013) present an interesting solution using *quadtree structures* to develop algorithms for feature detection and caching. The quadtree structure acts as a spatial index, which leads to a data structure that is suitable for generalization purposes like simplification, selection, displacement or aggregation of features. Several algorithms can be applied to fulfil one or several of the generalization operators. The authors experimented with several algorithms, which already led to impressive results. They implemented a prototype in Java which seems to be very fast. Still, this implementation is not available for the common web cartographer, and the algorithms have not been tested for usability yet.

TREVES (2011) gives a good overview over the marker cluttering problem in web maps and presents several solutions on his blog. He also experiments with heatmaps and grid-based clustering of features as alternative solutions, thus approaching the problem from a less technical but more usability-driven side.

BURIGAT & CHITTARO (2008) analysed methods to declutter icon based symbolization in mobile maps using icon aggregation algorithms. Their implementation uses a conflict graph to detect mutually overlapping icons, but is capable only of handling a few hundred features on runtime.

FIELD (2012) describes how to transform point-based features into a visualization using hexagon-shaped polygon cells for multiscale web maps. He calls this process ‘data binning’. The steps he takes to transform the data correspond to the steps used in this thesis, except the fact that in his approach, the data is processed *before* publishing it into a web map using desktop GIS software. The resulting map is aesthetically pleasing, but again, this approach is a custom solution.

The investigation of related research indicated that most approaches focused mainly on the technical aspects of the cluttering problem. Not much research has been conducted on the field of map usability: Often, it remains unclear, how users actually perceive and interpret the clustered information. The eligibility of the clustering approaches should be also assessed on the aspect of usability.

A second key finding is that web maps are being used more and more on mobile devices. The type of construction of smartphones and tablet computers leads to several design issues: The limited screen size and screen resolution exacerbate the icon cluttering phenomenon, and demand for solutions to declutter the map. In addition, touchscreen interfaces demand for bolder menus and icons. Applied to the issue of web mapping, this fact demands for bigger point symbolizations which have to be arranged on a small screen estate. Consequently, this fact underlines the necessity to facilitate the data visualization on the map using methods of cartographic generalization.

2 WEB MAPPING

In this chapter, the fundamental basics, techniques and issues regarding web mapping and related topics will be addressed. As geospatial data can be found anywhere, and freely available tools make it easy to publish maps, some kind of web mapping is used in almost every web site today. With mobile devices gaining more and more popularity, location based services and mobile (web) mapping form an additional field of application.

To begin, it is necessary to understand the meaning of the term *web mapping*. In literature, web mapping is defined as '*the process of designing, implementing, generating, and delivering maps on the World Wide Web*' (NEUMANN 2012, 567). It is closely related to the terms *web cartography* and *web GIS*. While the former also includes the conceptual and theoretical background of web maps, a web GIS is a web mapping application which emphasizes on the analytical part and offers more advanced tools to the user. As a consequence, a modern *web map* can be defined as a digital, interactive map using raster- or vector-based data, whose main purpose is to communicate geospatial information requested by the user.

PETERSON (2003, 6) distinguishes web maps in three classes: static, interactive and animated. The first is meant to be the equivalent to a printed map, hence a static raster map. The importance of this class of map nowadays is negligible. Today, maps on the internet usually are highly interactive, and in some applications the user is able to perform tasks which are already related to a GIS. Hence, web GIS and web mapping are terms which are sometimes used synonymously, as the capabilities of both web mapping and web GIS are very similar. MITCHEL (2008, 9) states that interactive web maps actually are web based applications, due to the amount of features offered.

Nevertheless, web mapping also brings some restrictions, mainly due to the fact that the used medium is a screen of small size and bad resolution (in contrast to a printed map). Also, limited speed of the internet connection (or no connection at all) can influence the user experience in a bad way. As web maps often combine data from different, sometimes remote, sources, data integrity often cannot be guaranteed during the life-cycle of the map product. Permanent maintenance, that is, server maintenance, validation of data sources, adaption to new requirements, is costly, and therefore often hard to achieve.

In the chapters following, the concepts, standards and technologies around web mapping will be discussed. First, chapter 2.1.1 will shine a light on the history of web mapping, followed by an introduction to common used standards and technologies. Contemporary web maps usually consist of raster- and vector-based data. The characteristics and issues of these types of map sources will be discussed in chapter 2.2. The ubiquitous presence of web maps, the easy to use development tools, interfaces and the like not only offer unlimited possibilities of mapping data, these circumstances also lead to a series of obstacles and limitations, which will be subject of chapter 2.3.

An overview over the most common types of web maps is given in chapter 2.4. This section is concluded with a discussion of cartographic communication, data visualization, and map design and composition in chapter 2.5.

2.1 How Maps Are Provided on the Internet

In this chapter, an introduction to web mapping and to the required core concepts and technologies will be given. Starting with a brief summary of the development of web maps in chapter 2.1.1, this section then will focus on the concepts used today (chapter 2.1.2), giving further detailed insights into existing core technologies.

2.1.1 A Brief History of Web Mapping

The nature of web mapping has evolved from a static appearance to a highly interactive and dynamic nature since its advent in early 1990s, shortly after the establishment of the Hypertext Markup Language (*HTML*). PETERSON (2003, 1) states that there have been three major step stones in the development of maps in the internet: In the early 1990s, a web map was nothing else than a scanned and digitised paper map. These maps did not feature any kind of interactivity and no adoption to the display medium was performed. This situation changed with the second step, which according to PETERSON started in the time about 1997. Back then, maps have been produced already solely for digital purposes, meaning that the original map was composed in digital formats using digital data. The shift to the digital medium brought a series of benefits as it offered access to other hypermedia like images, videos or animations (MUEHLENHAUS 2013, 9), and accelerated the map making process. However, during this period of time, these maps have usually not been distributed by the internet but mainly by discrete media like CD-ROMs. Often, these maps were accompanying products of bigger software packages like the Microsoft Encyclopaedia.

The last step of PETERSON's classification describes the current situation, in which web maps can be seen as the result of continuing development and an evolving cartographic design process. The pervasive existence of the internet makes the web the core distribution medium and a vast amount of available data leads to a manifold collection of different maps and mapping applications. LIENERT et al. (2012, 23) summarize the current situation as follows:

'Internet maps are the major form of spatial information delivery, as the Internet is today the primary medium for the transmission and dissemination of maps.'

Maps in the internet are not limited to browsers anymore, as smartphones and tablets become more and more common, and thus, native apps offer mobile offline and online usage of mapping, navigation and location based services (MUEHLENHAUS 2013, 12).

Web maps how we know them today offer a variety of controls and functionalities which have been reserved exclusively for desktop GIS programs a few years ago. Over the last 20 years, these features like feature selection, control over layers, manipulation of the map styling, zooming, panning, and advanced calculations like measuring, buffering, or routing have been implemented step by step, as described in the following paragraphs. A graphical overview over selected milestones can be seen in Figure 4.

From a technical point of view, first web map services like the *Xerox PARC Map Viewer* (from 1993, see Figure 3) only provided static raster images representing maps without much interactivity. If the user wanted to change the extent or the zoom level of the map, or wanted to add or remove layers, or change the map projection, the whole web page had to be reloaded.

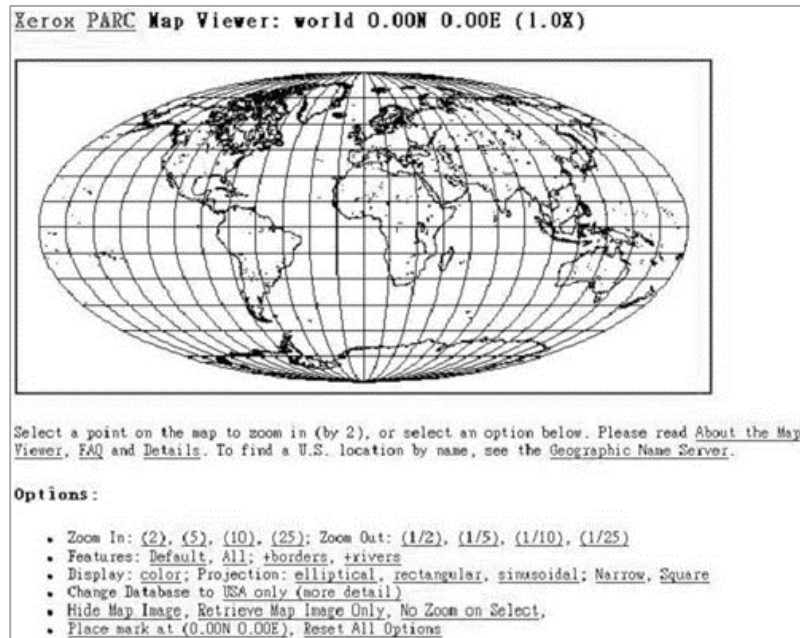


Figure 3: The Xerox PARC Map Viewer, one of the first interactive maps on the internet. (DETWILER 2014)

The next accomplishment was made in 1995 with the development of an application called *GrassLinks*. This web interface was developed by the University of California in Berkeley, and was built on top of the open source GIS *GRASS*. This product featured an interactive control which enabled the user to click anywhere on a map to *obtain information* about the underlying map feature.

In 1997, a server called *TIGER Map Server* was published by the U.S. Census Bureau, which allowed the access to the topographic dataset of the United States of America. The most important innovation here was the implementation of layers in the web interface. It was possible to toggle various geographic entities in the map, similar to desktop GIS applications (DETWILER 2014).

In 1996, a new web service called *MapQuest* was launched. This service was the first interactive mapping site for consumers, and introduced navigation and driving directions as a feature for web maps. In addition, the site offered access to business directories, aerial images and road maps (ROSENBERG n.d.). MapQuest became the most popular and successful mapping service, having up to 40 million users per month, until Google started to revolutionize the market with their own services, as described below.

With the establishment of the concept of Asynchronous JavaScript and XML (*AJAX*, cf. GARRETT 2005) in 2005 it was possible to dynamically change parts of the web page without the need of a whole page refresh. This situation contributed to an increased user experience, as the user was enabled to change the extent, zoom level and even content (with layers) of the

map on the fly, like in an ordinary desktop program. The term *slippy map* is often used to describe such a map with enabled pan and zoom capabilities (OpenStreetMap 2015a).

In 2005, Google started its online services Google Maps and Google Earth and radically changed the way we use geographic information on the internet (MUEHLENHAUS 2013, 10; YUN 2007, 1). Google used the concept of AJAX to provide that new look and feel to the broad audience of internet users. The company also offered an Application Programming Interface (*API*), which enabled users to create and spread individual maps on their websites using their own contents. This type of map is often referred to as a *mashup* (TURNER 2006, 2; see also chapter 2.4.1).

At about the same time, when Google started its successful launch of mapping services, another incisive project was founded: *OpenStreetMap* (OSM). In 2004, this project was initiated by Steve Coast in the United Kingdom as an attempt to map the country. Since then, OSM developed to an open, crowd-sourcing based *geospatial database* with global coverage, which is maintained and enhanced continuously. Thousands of volunteers around the globe contribute to this database, and, where available, open government data is used as well (OPENSTREETMAP 2015b). Today, OSM offers a series of base maps and its own API, but as the database is free to use, many other services, applications and maps are derived from this data source, as well.

OSM is subject to many research investigations. For instance, the data quality and accuracy was assessed by JOKAR ARSANJAN et al. (2015). The authors conclude that data quality heavily depends on the region: areas of higher interest are mapped often with more spatial and temporal accuracy and outperform the quality of commercial mapping providers, while other areas are represented only poorly.

Today, web maps do not have to be limited to the two dimensional space anymore. Recent technical developments (see chapter 2.2) allow a seamless switch between two dimensional map visualization, and three dimensional terrain models. Although three dimensional visualizations have been available before, the integration of both techniques into one product leads to a new user experience. Again, Google has been one of the first on the market, with integrating its Google Earth, and Google StreetView into the normal Google Maps application.

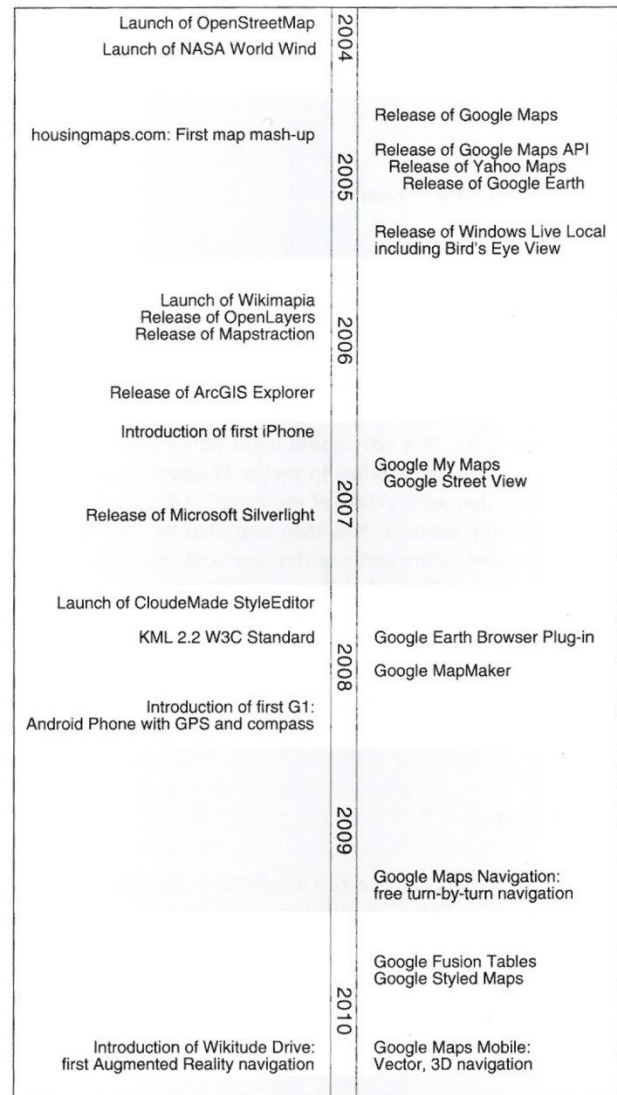


Figure 4: selected events in the history of web mapping services (PETERSON 2012, 14).

As already mentioned, recent developments have brought maps on mobile devices. With the release of the first Apple *iPhone* in 2007, mobile applications became popular, and many applications featuring Location Based Services (*LBS*), directions and navigation, geotagging and the like have been developed. Mobile mapping applications help to simplify the everyday life, as they provide the needed information ‘*specific to the current location of the mobile user*’ (PETERSON 2012, 11). Many applications are not even dependent on a persistent mobile internet connection, and offer offline storage or operation modes.

2.1.2 Standards and Technologies

Most web mapping applications today make use of the same protocols, standards, and even graphical user interfaces (*GUI*). Cartographers and map makers have several web mapping libraries and frameworks at hand, which offer a more or less customizable set of features accessible through their APIs.

The de-facto standard map *projection* used in web mapping at present is the *Web Mercator* map projection (EPSG: 3857), which is shown in Figure 8. This projection is a special derivative of the ordinary *Mercator* map projection. The Mercator map projection itself is a cylindrical, conformal map projection invented by Gerardus Mercator in early 1569 (BATTERSBY et al. 2014, 86). By the nature of a map projection, every approach to transform spherical or ellipsoidal coordinates into coordinates on a plane leads to some kind of distortion. The Mercator map projection as well cannot preserve angles and areas at the same time. Thus, it is not an equivalent projection, leading to massive scaling distortions in the polar regions. The Web Mercator map projection mentioned above, also sometimes referred to as *Spherical Mercator*, differs from the original one in the underlying mathematical model: it uses a simple sphere to model the earth (BATTERSBY et al. 2014, 87).

This map projection is not always a suitable way to visualize geodata (see chapter 2.3), but as most web mapping APIs are currently limited to the usage of prerendered raster tiles as base maps, and the majority of base maps are rendered in the EPSG: 3857 map projection, most applications cannot make use of other map projections.

As mentioned in chapter 2.1.1, one major milestone in the development of the web and web services was the introduction of AJAX in 2005. Today, all communication is usually performed using the AJAX communication model. In using an asynchronous communication method, data can be requested continuously in the background, without interrupting the application or locking up the browser. A scheme of this model can be seen in Figure 5. Only this model made it possible to create smooth reacting maps, and dynamically created and personalized content.

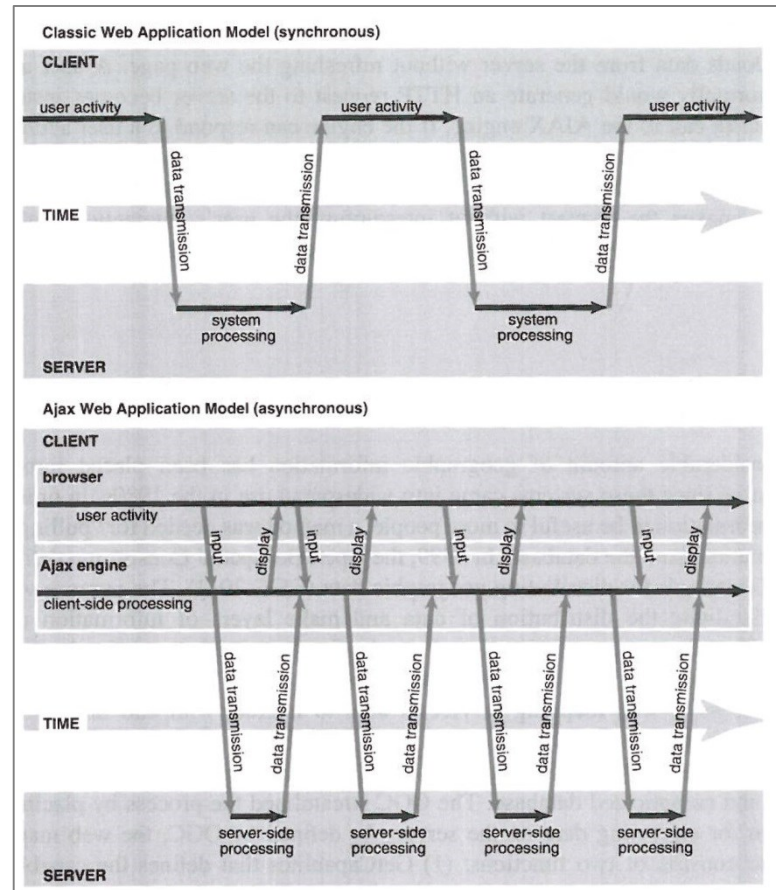


Figure 5: Comparison of synchronous and asynchronous client-server communication methods (PETERSON 2012, 9).

While first web pages used the synchronous model, as shown on the top of the illustration, every modern web 2.0 application is using the asynchronous model to handle dynamic content. Web mapping services make use of this model, as well. Geospatial data is usually transferred to the client using one of the following *standards* defined by the Open Geospatial Consortium (OGC). For raster-based data, a Web Map Service (WMS) or Web Map Tile Service (WMTS) is used to handle the communication between map client and server in a standardized way. The client therefore requests the data for a defined map extent in a specified map projection and optionally file format (such as PNG or JPEG).

For vector based data there exist several approaches: Either a Web Feature Service (WFS, also specified by the OGC) is used, meaning that a special geospatial server has to handle the requests, or the data is requested using (compressed) exchange formats like the JavaScript Object Notation (JSON), or the Extensible Markup Language (XML) respective a derivate of that like the Geography Markup Language (GML).

Most Web GIS and web mapping applications are programmed directly in *JavaScript* (or ECMAScript, which is the official, but unpopular term). JavaScript is a dynamic language, which supports multiple paradigms, like object-oriented approaches, but lacks some concepts and structures found in more advanced programming languages like C++ or JAVA. JavaScript is compiled by the web browser during runtime, the source code of a written piece of software is open and can be accessed through the browser. Modern web mapping applications make heavy use of sustainable, modular techniques like AJAX or Representational State Transfer (REST), and many frameworks are available which emulate more sophisticated fea-

tures like classes. Only few applications are still written in languages like Java or Adobe Flash, as those technologies require additional software plug-ins, and are often subject of security issues (cf. PEHNKE & SCHMID 2012).

Together with the newly released fifth version of the HTML standard, which introduced new elements and concepts like the canvas element or offline storage, JavaScript offers astonishing new possibilities, which are not limited to the two-dimensional visualization: high-performant three-dimensional graphics are possible as the canvas element allows the direct access to native functions of the Graphics Processing Unit (GPU) of a computer (see also chapter 2.2.3).

For the application of web maps, this new technology allows a new level of smooth and seamless map interaction: the map can be panned, rotated and zoomed with smooth animation, multiple layers can be faded or blended together using graphical transition techniques, and the transition between two-dimensional and (pseudo) three-dimensional visualizations can happen instantly.

2.2 State of the Art Web Mapping

This chapter is devoted to give an overview over recent developments in web mapping. The majority of maps which can be found on the internet today are built on top of a web mapping API. There exist a couple of popular frameworks, which will be introduced in chapter 2.2.1. The selection of a suitable API forms the first step when creating web mapping applications, and is depended on the kind of data, and the type of layers which will be used in the application. The subsequent subchapters will discuss the characteristics of raster-based and vector-based web mapping approaches.

Most web mapping applications today are using a combination of raster- and vector-based sources. The background data is usually displayed via prerendered static topographic maps or orthoimageries, and the overlay holds thematic data (BEREUTER & WEIBEL 2013, 1). Thus, the standard solution includes prerendered raster tiles with the interactivity given by overlaying vector layers. Often, these overlays contain point-based data: they form a combination of custom geodata with a standard base map. A typical application is illustrated in Figure 6. One or several possible base maps form the background of the map and are usually served using some kind of tile service. On the next tier, one or several raster based overlays can be added to the map. These are usually implemented using a Portable Network Graphics (*PNG*) raster file format which offers transparency. The resulting layers may represent topographic or thematic information like park areas (middle left) or a hill shade with contour lines (middle right).

Only on the last tier, vector-based graphics are used and thus, interactivity with the data representations can be offered. Layers on this level usually contain the thematic information, like POIs, which can be represented using marker- and icon-symbolizations. In addition, pop ups are can be implemented for the presentation of complementary information.

The interaction between the user and the map can be implemented using several methods. On classical computer systems like laptops or desktop computers, interaction with the user is provided via the keyboard and the mouse through the so called *WIMP* (windows, icons, menus, and pointer) model. The graphical interface is built on the usage of those input devices. Con-

trols for zoom, pan, feature selection and the like are implemented using common windows, dialogs and mouse gestures (MUEHLENHAUS, 2014, 21), and keyboard commands can be used as command shortcuts.

Although this model is applicable for desktop computers and laptops using traditional operation systems, mobile devices with touchscreen support demand for another type of user interface. On these systems, the user interacts with the system using multitouch gestures, or even eye-tracking. Additional sensors provided by the device can be used to rotate the map, or adapt the style of the map display to the current situation (day and night mode depending on the ambient light, navigation and browsing mode, depending on the GPS status, and the like).

Consequently, touch-optimized GUIs do not rely on pointers and windows anymore, as the limited screen size and touch accuracy on mobile devices in combination with small icons and menus would just lead to an unsatisfying user experience.

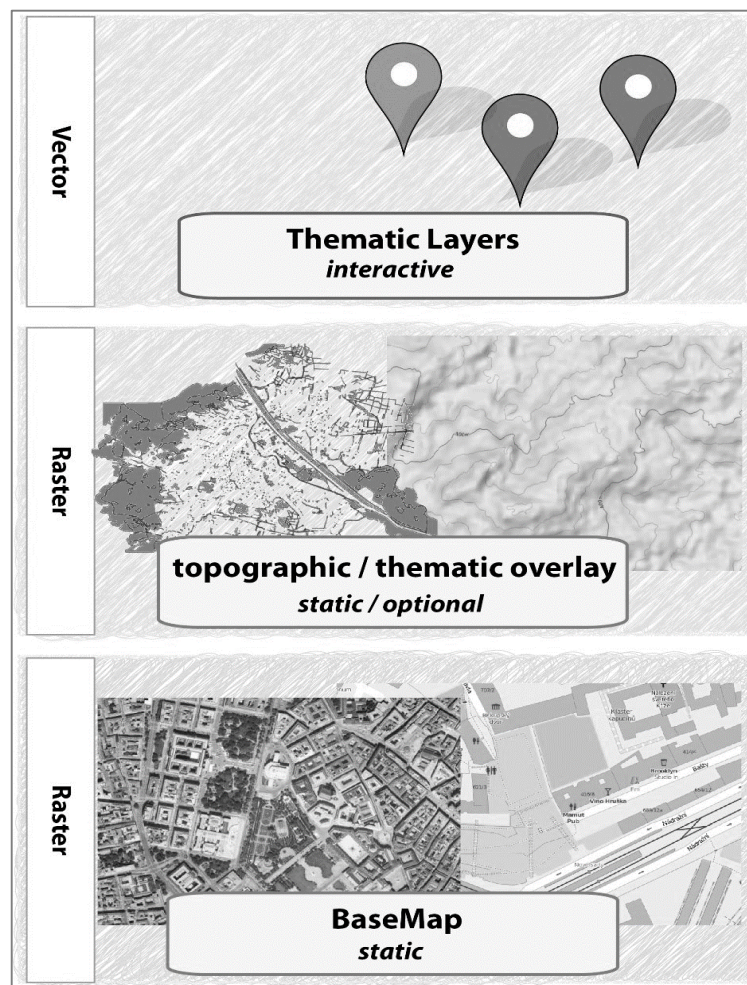


Figure 6: Composition of a typical web mapping application. Sources (clockwise from bottom left: OpenStreetMap / Stadt Wien, MA 41 / ASTER GDEM / Stadt Wien MA 22)

The communication between the client and the server is usually performed using standards defined by the *OGC* – that is in particular the *WMS* and the *WMTS* for the first two tiers in Figure 6, and *WFS* for the last one. As all these services require a special kind of server, which has to support geospatial operations, light weight mapping applications can also obtain the vector data using a transfer file format like *JSON* or *GML*. This approach comes in handy

in situations, where the implementation of a database would be oversized, compared to the scope of the used data.

Vector formats are slowly replacing raster-based maps, as the web clients get more and more performant, and transfer formats and concepts are getting smarter. For example, the *topoJSON* format (TOPOJSON-1), which is a subset of the *geoJSON* format, only needs a fraction of the latter's file size as it only encodes and stores topologies. Once the compressed file is received by the client, the original geometries are recalculated. Moreover, the concept of *tiling* (cf. chapter 2.2.2) is now also available for vector formats. Delivering maps in a vector-based fashion brings some advantages, such as an increased user interactivity and the ability to style the map individually. In chapter 2.2.3, vector-based web mapping will be discussed in more detail.

2.2.1 Web Mapping APIs

As already mentioned earlier, a typical implementation of a modern web mapping- or Web GIS-application uses one of the popular mapping APIs like:

- Google Maps API,
- Bing maps API,
- ESRI ArcGIS API for Flex,
- leaflet.js API and
- OpenLayers API.

These APIs can be distinguished into two categories: First, there are *frameworks*, which come as a complete package and provide everything to create a web map. That means, not only the application logic (that is, a JavaScript library) which handles the map interface, server communication and user interaction. These frameworks also offer prerendered base maps and orthoimages, which are ready to use. The first three entries from the API's mentioned above can be counted to this category.

The second category are *toolkits*, which consist only of the mapping library. In contrary to frameworks, these libraries are open source and therefore, free to use – even for commercial purposes. With the source code open for everyone, these libraries offer more flexibility as existing functions may be enhanced and new functionalities can be implemented in a rather easy way.

However, all of these toolkits and frameworks are written in JavaScript and are capable to handle both raster- and vector-based data sources to some extent. They vary in the complexity of available functions, and in the quality of documentation. Therefore, some APIs are easier to use, while others demand more background knowledge from the map creator.

Furthermore, there exist specialized toolkits, which can only handle vector-based data sources. At the time of writing, the most advanced ones are polymaps and Data Driven Documents (*d3.js*). It is also possible to combine a multi-purpose mapping library like OpenLayers or Leaflet.js with *d3.js*, and thus to get the most out of both libraries.

2.2.2 Raster-Based Web Mapping

The majority of web mapping applications are using raster-based background layers, also referred to as *base maps*. As mentioned in chapter 2.1.1, the predominant base map provider is Google, but there exist also products offered by MapQuest, Bing Maps, Nokia or ArcGIS Online, and various maps using data derived from OSM, just to name some.

A base map is usually a prerendered set of raster images, which are served dynamically to the mapping client depending on the zoom level and map extent. An inbuilt flaw of this concept is that the user (and the web map creator) has to accept the generalization and styling decisions made by the provider of the base map (MUEHLENHAUS 2013, 11). No individual design or styling is possible: the map projection, available scale steps, generalization rules, selected features, feature symbolizations and labels – all of these parameters are already pre-set by the provider.

A solution out of these limitations may be available with the rise of vector based mapping (see the next chapter), which will enable the map creator to only use the raw geodata and create an individual map representation on the fly.

For raster-based overlays which are delivered using the WMS standard, the styling of the map layer can be influenced using a special description format: a Styled Layer Descriptor (*SLD*), which was also developed by the OGC. Using this language, the WMS is extended to ‘*allow user-defined symbolization and coloring [sic!] of geographic feature and coverage data*’ (OGC 2007).

Two concepts are pervasive in raster-based web mapping: *Tiling* and *caching*. As base maps are stored in raster formats, every web map is actually just one huge image. Streaming the extent of the whole world in one image would overstrain typical internet connections. Furthermore, loading the whole image would be unnecessary: Typically, only a small extent of the world is requested at once.

Thus, the concept of *tiling* exists. A regular matrix is laid over the rasterized map image, which is then partitioned into equal sized, small parts called *tiles*. Typically, each tile has an extent of 256 times 256 pixels. This process is repeated for every zoom level, resulting in 2^n tiles for a given zoom level n . Every tile can be addressed by a unique combination of x , y and z values.

As popular web mapping services like Google Maps or OSM are frequently visited by millions of users (PETERSON 2008, 5), it would be unwise to generate the same raster tiles again and again. They are therefore *cached* and only re-rendered, when the underlying data has changed.

The *benefits* of raster-based sources are the quick loading times, as each map tile only needs a few kilobytes. Related to that, the effort to process and display the data is trivial, as only basic graphic tasks have to be executed. A potential bottleneck for raster-based web mapping can result from the server, which delivers the tiles. If the hardware is insufficiently dimensioned, or the internet connection does not deliver enough bandwidth, loading times will increase, leading to an unsatisfying user experience.

However, intelligent loading mechanisms might enhance the user experience as off-screen map tiles can be preloaded, and thus are available immediately upon request (MUEHLENHAUS 2014, 208).

2.2.3 Vector-Based Web Mapping

As the name already implies, vector-based mapping uses vector graphics to create a map image. All features are drawn by using geometrical primitives, such as points, lines, curves or polygons. In doing so, the application is enabled to manipulate all objects in mathematical ways (e.g. transformations, rotations and the like). This approach forms the most notable *difference* to raster-based mapping: the efforts of processing the raw data to the final, (appealing) map image is shifted away from server-side pre-processing. Vector-based web mapping requires real time calculations performed directly on the client.

There exist various implementations of vector engines in modern browsers. Some come as plug-ins, such as *Adobe Flash*, *Microsoft Silverlight* or *JavaFX*, and some are supported natively by the browsers. One of the most popular technologies is the Scaled Vector Graphics (*SVG*, W3C 2011) format, which is drawn directly in the Document Object Model (*DOM*) offered by the browser. Another, quite new technology is the *Canvas* element, which was introduced as a part of the new HTML 5 standard. A canvas element offers the possibility to draw any shape on a two dimensional pane using JavaScript (W3C 2014). In addition, the *WebGL* technology (KHRONOS 2011), which is a derivate of the OpenGL ES 2.0 language developed by the Khronos Group, offers low level access to the graphics card of a computer. It is also capable of rendering three dimensional objects, and therefore a promising technology for three-dimensional data visualizations and cartography.

Vector-based formats can be used in web mapping in two ways: First, as a standalone technology, which means that all map elements are rendered in a vector format. And second as an enhancing technology for raster-based web mapping. In this case, the base map is rendered using a raster-based technology like a *WMTS*, and the vector-based overlay is used for interactivity. This is a common use approach for most map mashups (cf. chapter 2.4.1) or marker maps, as shown in Figure 7.

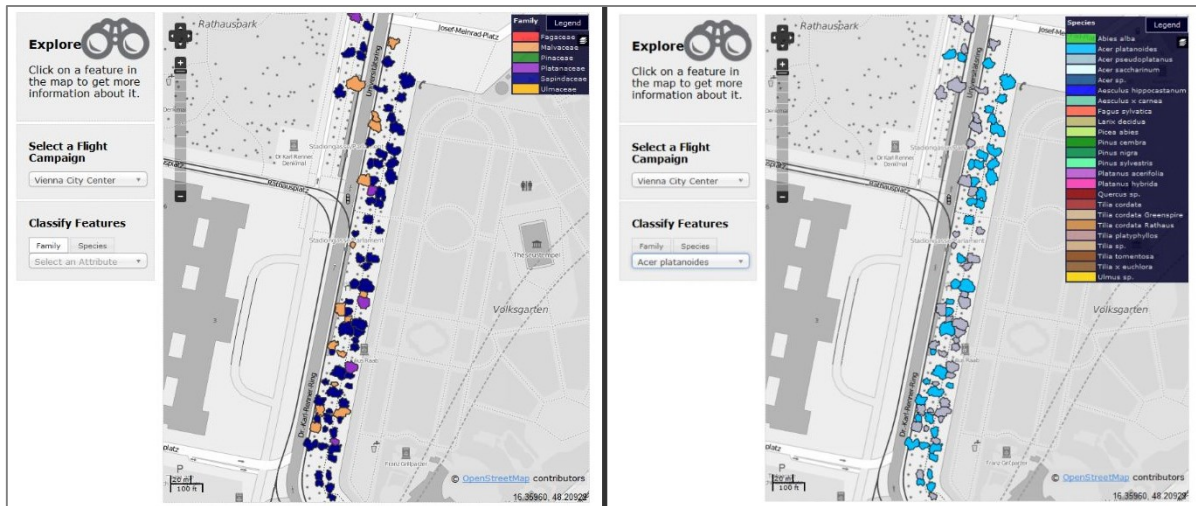


Figure 7: Combination of raster- and vector-based mapping technologies. The vector layer representing tree objects can be modified on the fly. Left: classification by family membership; right: filter for a specific tree species (LVISA-1).

In contrast to raster-based mapping, vector formats offer a series of *benefits* for interactive mapping. According to LIENERT et al. (2012, 24), those are:

- looseness scalability (no graphical artefacts),
- on the fly adjustable symbolization (as shown in Figure 7),
- ability to animate the geometry or symbolization of features,
- or to hide or show them without a forced map reload,
- the ability to store and modify attributes for each feature as well as
- the possibility to change the map projection on the fly.

Related to that, vector-based mapping allows the dynamic application of generalization operators (cf. chapter 3) on map elements during runtime, depending on the zoom level, screen resolution and requirements of the current situation.

Vector-based data are usually stored and transferred over the internet using GML, KML or geoJSON formats, depending on the type and amount of data. The geometry and attributes of geospatial data stored in a vector format are encoded using numbers and strings, resulting in an extensive amount of information, which is required to describe the data properly. As a consequence, data stored in a vector format can result in huge file sizes. Several approaches like file compression, simplification of the geometry, or the implementation of local coordinate systems have been developed, to minimize the amount of data.

Today, vector-based maps cannot replace rasterized base maps yet, as the complexity of a base map is just too high. The amount of data needed to render a decent background layer is huge, and the styling rules needed for symbolizations, labels and the like are too complex to allow a smooth user experience (cf. GAFFURI 2012). Other unsolved problems concern enhanced cartographic illustration techniques like hill shading, and special cartographic issues like rock depiction, although the first was already addressed by AUER (2012).

2.3 Obstacles and Limitations

Modern web mapping toolkits, a variety of free-to-use base maps and orthoimagines, as well as various data sources which are accessibly effortless using standardised APIs, make it easy and convenient to create and share spatial data visualizations on the internet. Map makers may choose from a variety of available options, and to some extent customize the appearance and styling of the map. In addition, the vast amount of data which is available to the public today offers almost endless topics to create maps about. However, this situation also leads to some obstacles, and the available technologies are limited. In the following sections, these issues will be explored.

2.3.1 Quality Issues

Without doubt, the success of Google Maps and other online mapping services brought web maps on almost every web site. Unfortunately, this development also brought up the side effect of quality issues.

First, as mentioned before, a rather cartographical illiterate audience was enabled to create maps with just a few clicks or lines of code. Second, as Google Maps was used almost everywhere, other projects had to provide the same look and feel in order to attract the attention of users. Improvements in the user interface were hindered. PETERSON (2008, 8) calls this ‘*the Google Maps Effect*’ which describes, that a user expects the user interface, map behaviour and handling of the map to be similar to the solutions, which Google implemented.

Furthermore, the predominant Web Mercator projection (see chapter 2.1.2) which is used today by all mayor base map providers (Google, Bing, OSM, and the like) often leads to a distorted projection of the world. Not only can the predominant use of that projection influence the way users picture the shape of certain regions – it is also unsuitable to provide a background layer for thematic applications using medium- or small-scale maps (MUEHLENHAUS 2013, 11 & 145; BATTERSBY et al. 2014, 88). Figure 8 shows the distortion of the polar regions caused by the Mercator projection.

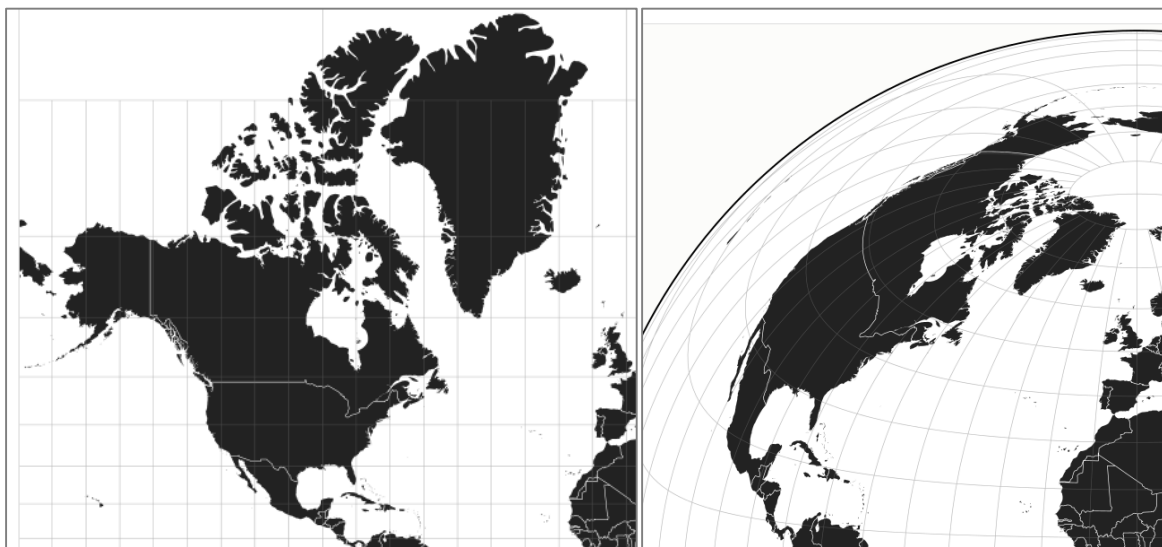


Figure 8: Left: Mercator projection as commonly used in base maps. (BOSTOCK, M. 2012a) Right: Lambert Azimuthal Equal-Area (BOSTOCK, M. 2012b). Note how the polar regions are distorted on the left.

2.3.2 Icon Cluttering

Besides the quality issues which followed the triumphant success of Google Maps, another issue arose with the massive availability of geodata like POIs. According to BURIGAT & CHITARRO (2008, 14), unprocessed displaying of too many features can lead to a dramatic degradation of performance, high transmission load, and more importantly, it can lead to an unwanted behaviour: the so called *icon cluttering* problem. This phenomenon occurs, whenever too many icons are placed on a too small map space. As a consequence, symbols are touching and overlapping each other. Important map features are hidden, leading to a low level of map readability.

The rise of mobile devices with their limited screen sizes, relatively coarse screen resolutions and touch-based input mechanisms even increased the problematic situation: now, even larger symbols are required to represent map features like POIs. In Figure 9, the impacts of point size, scale denominator and distance have been composed. The use of larger symbols leads to a more likely overlapping of the symbols, especially on small scale levels of the map.

This effect further emphasizes the need for generalization steps, and thus, for a *clustering strategy* on cluttered maps, as it becomes increasingly difficult for the user to precisely select one specific marker, when the neighbouring icons are too close to each other.

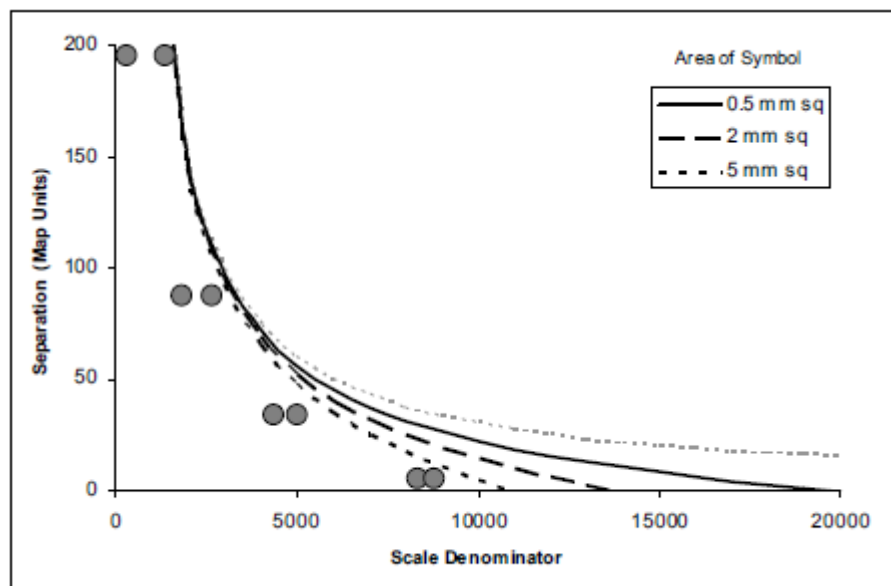


Figure 9: Scaling of point symbols on various scale levels (EDWARDES et al. 2005, 14)

2.4 Common Types of Web Maps

‘It is not possible to get an overview of an area in any way other than by consulting a map’

(KRAAK & ORMELING 2003, 33).

This statement is still valid, even in modern times, when almost any information can instantly be obtained from almost anywhere in the world over the internet. Maps always serve a certain purpose: they are either designed to convey geospatial relationships, or to analyse information about relationships graphically. The most important function of maps in general is providing navigation or orientation (KRAAK & ORMELING 2003, 44). This role also applies to maps on the internet. Furthermore, besides this basic functionality, web maps can serve other purposes, as well.

There exist various approaches to classify the vast amount of available web mapping products. KRAAK (2001, 3) was one of the first authors, who created a classification of web maps. He distinguished web maps according to their nature into static and dynamic. Each of those two types then were further divided into view-only and interactive maps. While this first approach was eligible at the time of creation, today almost every web mapping product is of a dynamic nature. Hence, this classification can be seen as a very generic, first attempt to classify the variety of map types. Besides the classification according to the *appearance* of the map, it is further possible to classify web maps according to their *purpose*. CARTWRIGHT (2003, 25f.) created six categories of web maps:

- Maps and image collections
- Downloadable data storages
- Information services with maps
- Online map generation services
- Web-atlases
- Hybrid products

In the era of the Web 2.0, different types of maps are used on websites. The most common type of a web map is a map mashup, which will be described in detail in chapter 2.4.1. A second application of web maps lies in thematic mapping, which will be discussed in chapter 2.4.2. As many real life phenomena are not of a static, but a rather dynamic, developing nature, animated web maps are a good means to communicate the relevant information to the target audience. This type of map will be introduced in chapter 2.4.3.

The following illustrations cannot depict the full amount of possible web maps. Combinations of the mentioned types are possible – and likely. Many thematic web maps are based on mashups. They might even be animated, if the data source is appropriate. However, for the purpose of developing a real time clustering algorithm based on grid structures, these types of web maps form the most worthwhile applications, therefore they have been selected for this thesis.

2.4.1 Map Mashups

One of the accompaniments of the Web 2.0 is the rise of the so called *mashups*. The incredibly high amount of content and data available through the web is not only being used to be rendered directly to the browser of a user, these data can also be accessed, processed and re-used by machines. WILDE (2006, 3) summarized, that ‘*mashups are a way to reuse the increasing amount of information being available on the Web; as well as [being] a trend for supporting tools and technologies to make application development in this environment easy*’.

While the term ‘*mashup*’ is only loosely defined, the minimal requirement for a web mashup seems to be the combination of a minimum of two different information sources using some sort of *API* and standardised exchange formats like XML (cf. WILDE 2006, WOOD et al. 2007, LERNER 2006).

Map mashups, or geo mashups are a derivate of the general mashup: they are a combination of content which has a geospatial context with a base map, usually using one of the public available mapping APIs, thus forming a (simple) web mapping application. As almost every topic has some kind of geospatial context, map mashups can be found literally anywhere on the web: in social networks, store locators, real estate platforms, weather forecasts, and many more.

The possibility to create mashups without much effort can lead to the same quality issues, which have been described earlier in chapter 2.3.1: web designers are usually rather unaware of the cartographic principles, and the result too often is a standard Google base map with too many markers on it.

2.4.2 Thematic Web Maps

Another common type of web maps are thematic web maps. As implied by the name, the focus lies on communicating ‘*information in a convincing manner about a topic or several topics*’ (MUEHLENHAUS 2014, 64). The thematic visualization stands in the foreground, thus the design of the base map should be simplified. The background layer only has to communicate the geographic context and does not have to provide detailed information.

Thematic representations can be implemented in various forms, such as cartograms, choropleth maps, dot maps or heat maps, just to name some examples (see Figure 10, right side). The choice of the right map form of course depends on the *nature* of the phenomena that have to be mapped, and how the data were collected. The spatial properties of real life phenomena can vary to some extent, which MACEACHREN classified as shown in Figure 10 on the left: The mapped phenomena can occur in a discrete manner, only covering certain places, or in a more continuous nature. Another way to describe phenomena is to classify it into abrupt or smooth (MACEACHREN, 1995, 303).

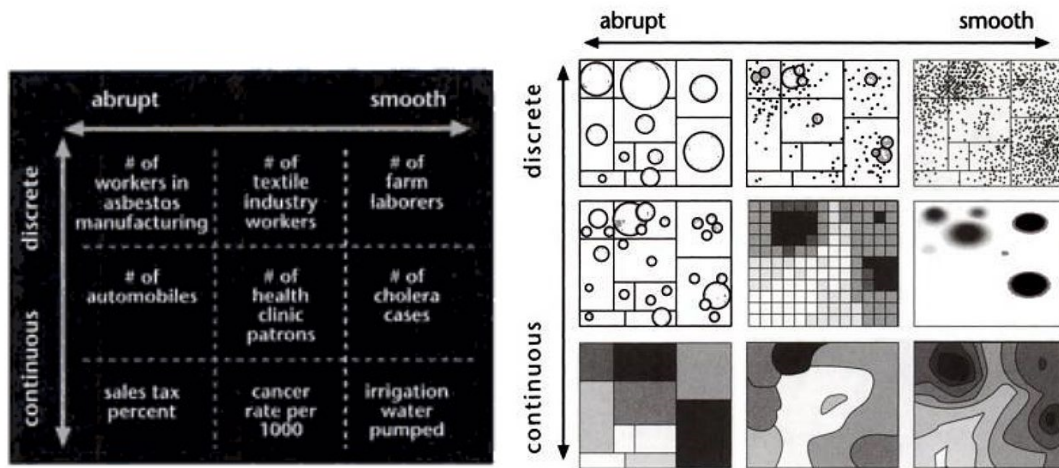


Figure 10: Classification of phenomena characteristics, and possible map forms (MACEACHREN 1995, 303 & 304)

However, the method chosen to sample the phenomena might lead to different characteristics of the resulting dataset. A continuous, smooth phenomenon might be represented by discrete, abrupt data, depending on the way the data has been collected.

Choosing the wrong thematic representation for a map can lead to unsatisfying results, as the reader of the map either will not understand the intended message, or the nature of the data might be presented in a misleading way.

From a technical point of view, thematic web maps are usually a combination of a raster-based background layer, and a vector-driven thematic layer, which provides the interactivity. The decision to use an existing web mapping APIs might allow a convenient workflow to publish the map, but it comes with the cost of limited flexibility. The map creator will have to choose from an available thematic representation, which might not be appropriate for the thematic data he wants to map. More exotic forms of data representation are usually not available, and hence the type of thematic maps more often is determined by the available technology and not driven by the idea to choose the method which represents the phenomena in the most accurate way (MUEHLENHAUS 2014, 145).

Another possibility to map data in a more insightful way is to only use vector-based content and a vector-derived mapping framework. In doing so, more flexibility in the styling of the layers, the type of thematic representation and map projection can be achieved. However, it has to be noted that the possible cost of a higher demand in data transfers and computing time has to be taken into account. Furthermore, the deployment of vector-based thematic web maps demands higher programming skills from the map creator, and more time for preprocessing has to be invested. More details about the characteristics of vector-based mapping and map composition can be obtained in chapters 2.2.3 and 2.5.

2.4.3 Animated Web Maps

The visualization of temporal processes is another typical application of web maps. Analysing data snapshots can be restricting, and oftentimes the derived results are not very adequate. Especially for continuously recorded data, a much more revealing approach would be to un-

derstand and analyse the *process*, thus to identify typical patterns in space over time (see also chapter 2.5.1).

For this purpose, animated maps can be an efficient means, as they allow to visually emphasise the temporal component of geospatial data (cf. KRAAK & ORMELING 2003, 54). It has to be noted, that animated web maps can also mean the animation of the map extent, orientation or zoom. While these animations can help to improve the usability, or at least the look and feel of a web map, they are not subject of this chapter.

Animated maps have been used in cartography since the 1930s, first as cartoons, later as computer maps. However, this type of map only emerged and established when digital maps could be distributed easily through the internet (MUEHLENHAUS 2014, 173).

The type of animation used for the visualization of dynamic data on thematic maps is called *cartographic animation*. OGAO states, that cartographic animation ‘*is about [...] change of geospatial data components of location, thematic attribute and time. Their real power lies in showing the **interrelationship** between these three components*’ (OGAO 2002, 19).

As a consequence, when composing an animated web map, additional map elements are required, as shown in Table 1: First, the user should be offered an *interface* allowing him to control the state of the animation. That is to start, stop or pause it, or to select certain time stamps. Furthermore, this interface can offer controls to change the interval length between each animation frame (ANDRIENKO et al. 2005, 207). Second, an additional *temporal legend* helps the map reader to understand, how quickly the data is animated, and which temporal unit is used. Last but not least, this legend furthermore enables the reader to understand, when the presented data occurred (MUEHLENHAUS 2014, 182). Temporal legends might take the form of a scale bar, a gauge or a clock. MUEHLENHAUS (2014, 184) states, that the placement of a temporal legend is of vital importance to keep the attention of the reader focused on the map. If the legend is positioned far away from the mapped area, the reader will be distracted and thus is likely to miss out information.

Similar to ordinary thematic web maps, the source of the geospatial data as well as the rendering technique is usually vector-driven. The base map should only show generalized information, as the visual focus clearly lies on the animated thematic data. Former approaches to create animated web maps used plug-ins or proprietary software like Adobe Flash, Apple QuickTime, Macromedia Shockwave File (SWF) or the Virtual Reality Modelling Language (VRML). While the first is still being used, most thematic web mapping applications nowadays are built directly on SVG or HTML 5 elements.

2.5 Map Design and Composition

Modern web maps consist of several elements, some of them are obligatory, some vary depending on the purpose of the map, or the type of the map. A typical composition of a web map was already shown in Figure 6. These elements are ordered in visual hierarchy levels, as shown in Table 1, below. Besides the map layers, which hold different types of content, a web map needs supplemental elements, for map interactivity, as well as labels, attributions and a title. The map legend is often omitted in general purpose web maps, which is unfortunate, as the symbolization of a base map is not always self-explanatory.

Table 1: Visual hierarchies for map elements of web maps. (Modified from MUEHLENHAUS 2014, 64)

Visual Hierarchy Levels for Web Map Design					
General Web Maps, Mashups		Thematic Web Maps		Animated Web Maps	
Level 1	Title, Map Symbology, Key Reference Data, Legend	Level 1	Title, Thematic Visualization, Legend	Level 1	Title, Animation Symbology, Map Symbology, Temporal Legend or Interface
Level 2	Base Map, Labels	Level 2	Generalized Base Map, Charts, Info Windows	Level 2	Base Map, Legend, Info Windows
Level 3	Map Interactivity Controls (Pan/Zoom/Rotate)	Level 3	Base Map Labels, Map Interactivity Controls (Pan/Zoom/Rotate)	Level 3	Base Map Labels, Map Interactivity Controls (Pan/Zoom/Rotate/Play/Pause)
Level 4	Locator Maps, Additional Multimedia	Level 4	Locator Maps, Additional Multimedia	Level 4	Locator Maps, Additional Multimedia
Level 5	Supplemental Information, Attribution, Neatlines	Level 5	Locator Maps, Additional Multimedia	Level 5	Locator Maps, Additional Multimedia

Thus, the creation of a *good* web map requires the inclusion of several visual levels, meaningful base maps and intuitive controls for map interaction into a coherent layout. As shown in the preceding chapter, the first step to create an effective, cartographically aesthetic map is to choose the right type of data visualization, especially for thematic purposes.

Applied to the topic of this thesis, when adding a clustering algorithm to a map, the choice of the cluster representations, as well as the type of clustering algorithm will have serious impacts on the insights the reader will gather from the map, as the transported message strongly depends on the way the data is presented.

In the next chapters, the nature of cartographic communication and data visualization will be touched on, considerable design principles for web maps will be discussed, and eligible types of cluster symbolizations will be explored.

2.5.1 Cartographic Communication and Data Visualization

Cartographic communication, or communication in general, can be defined as the process of exchanging information with the user, while the first has an emphasis on spatial structures (HAKE et al. 2002, 18). The major purpose of making maps can usually be seen in the effort to communicate this information in an efficient, understandable way to a target audience by using cartographic methods of generalization and visualization (cf. KRIZ 2009, 3).

As already mentioned earlier, in this data-rich time we are living in, the visual data exploration is the key to the discovery of knowledge and information. Usually, this process of exploration happens in three steps: (1) overview, (2) zoom and filter, and (3) details-on-demand. This approach has been termed the ‘*information seeking mantra*’ (SHNEIDERMAN, 1996). According to KEIM et al. (2005, 25), these three steps look like this:

The user gathers an overview of the data in a first step. Here, he identifies interesting patterns or clusters in the data and sets his focus on one or several of them. In the next steps, as the user wants to further analyse this selection, he has to drill down and access details of the data of interest. KEIM et al. further state that ‘*visualization technology may be used for all three steps of the data exploration process.*’ (KEIM et al. 2005, 25). Applied to digital maps, and web mapping in particular, the user is enabled to perform the selection process described above in an efficient way by using the zoom- and pan-interactivity, as well as popup functionalities offered by the mapping application.

However, these technical developments alone cannot prevent cognitive overload, in contrary, when applying them without proper knowledge, the result might be a confusing, unreadable map. As a consequence, it is the responsibility of a cartographer to design and compose a (web) map in a way that the spatial information is communicated in a digestible, understandable manner.

As already mentioned earlier, *data visualization* is the key to knowledge discovery, and when performed properly, leads to a deep understanding of the characteristics and relationships of the shown phenomena. In order to analyse and map this data in a correct way, it is essential to understand the nature of the data.

Spatio-temporal data consists basically of three components: space, time and objects. Data can be classified using the *type of spatial distribution* (see chapter 2.4.2), or according to their *temporal properties*. BLOK (2000, 25 ff.) distinguishes these properties into existential changes like the disappearing or appearing of an object, changes of spatial properties, like changes in size or form, and changes of thematic properties, which means changes in attribute values.

Depending on the nature of the data which has to be communicated, different forms of visualization may be considered to be benefiting, or as ANDRIENKO et al. (2005, 201) put it: ‘*It is commonly recognized that techniques used for graphical representation of data must correspond to characteristics of the data*’. Displaying data on a map may not always be the best solution, depending on the scope and complexity of the topic. However, if a map is chosen as the graphical representation, different forms of data visualization are available, as mentioned in chapter 2.4.2.

2.5.2 Design Principles

Maps should always transfer a clear message. Depending on the scope and the target audience, different levels of complexity may be appropriate to do so. For instance, thematic maps using multiple layers with different types of detailed symbols and representations might be used in printed atlases, but might lead to cognitive overload on a typical computer screen, and in all probability make the map unreadable on a small screen of a smartphone. Hence, the design of a web map depends on technical limitations like the targeted screen size and resolution, and on the target audience. Luckily, several techniques exist to create clear, simplified maps, and to emphasize certain characteristics of the presented data.

In 1983, BERTIN developed a set of *visual variables*, which are used in mapmaking. He identified six core variables, which allow the graphical presentation of information in an efficient, intuitive way. Those are: shape, orientation, texture, size, and the hue and value of colours.

Since then, several cartographers investigated the field of visual variables, and altered or enhanced the original set defined by BERTIN, as shown in Figure 11. For instance, KRYGIER & WOOD added a variable for colour intensity. The illustration also shows, that some visual variables are more eligible to show qualitative data, while others are best used for the presentation of quantitative data.

	Points	Lines	Areas	Best to Show
Shape		Possible, but too Weird to Show	Cartogram	Qualitative Differences
Size			Cartogram	Quantitative Differences
Color Hue				Qualitative Differences
Color Value				Quantitative Differences
Color Intensity				Qualitative Differences
Texture				Qualitative & Quantitative Differences

Figure 11: Visual variables, after KRYGIER & WOOD (KRYGIER & WOOD 2011, 177).

Intuitive symbols and data representations however, will more likely result from the thoughtful *combination* of several of the presented variables. The proper use and manipulation of these visual variables is crucial for the correct communication of thematic data, and to create a clear and intuitive map.

As shown above, *colours* play a big role in the theory of visual variables. Our strongest sense as human beings is the visual perception of objects. It is only natural to use this fact to emphasize objects and guide the attention of the map reader in the right direction. On the other side, applying colours in an unintuitive way will most likely lead to misinterpretation and confusion.

2.5.3 Symbolizations

A typical map mashup makes use of the default marker symbols, also known as push-pins, or balloons, which are provided by the chosen web mapping API (cf. chapter 2.2.1). These generic icons are rather bulky, meaning that the size of this symbols in combination with too many features on too little map space oftentimes lead to icon cluttering on the map and to cognitive overload. In addition, these default markers are not descriptive, meaning they do not indicate which kind of feature they represent on the map. Recalling the cartographic communication, a requirement for map symbolization should be to use intuitively comprehensible

symbols. This is especially the case when a clustering algorithm is applied to simplify the data visualization, as the placeholder symbol has to communicate several information to the user, such as:

- The *location* of the cluster
- Additionally, the *area* covered by the cluster
- The *count* of the aggregated features

When thinking about possible symbolizations, two geometrical forms for the representation of a cluster intuitively come to mind: Both a *point-based* symbol or a *polygon-shaped* representation seem to be adequate to represent a cluster on a map, as those two forms offer the possibility to visualize the above mentioned information.

According to the theories about visual variables (see chapter 2.5.2), the thoughtful manipulation of one or several of those variables leads to insightful data visualizations. This should be kept in mind, when designing a symbolization for clusters.

Point-based symbols are used already in commonly known implementations of distance-based clustering algorithms (see chapter 3.2 for further details). The default symbolization provided by those tools is a colour-coded circle, with a label holding the amount of features, as shown in Figure 15. The *colour hue* of the symbol is the only the visual variable which is manipulated to indicate different amounts of features within a given cluster. A typical colour set for the classification of the quantity of features often is green for a small amount of features, yellow for medium and red for a high count of features (see Figure 43 on page 81 for a composition of typical implementations).

From a cartographic point of view, this implementation is actually inappropriate, as it uses a *qualitative* visual variable to show *quantitative* data (cf. MUEHLENHAUS 2014, 132 and chapter 2.5.2). The choice of a multi-hue colour set might be unfortunate, as users will associate the colour hues with a traffic light: green is good, red is bad. When using colour variables to manipulate the appearance of the symbols, only the *saturation* or the *value* of the colour should be manipulated.

Alternatively, the quantity of features within a cluster could as well be represented by manipulating the size of the symbol. In this case, the colour of the symbols would not be altered, and the increasing size of the circle indicates the quantity of features within a cluster intuitively, as illustrated in Figure 12.



Figure 12: Cluster symbols indicating amount of contained features with varying icon size.

However, when manipulating the size of the symbol for the purpose of data classification, the resulting icons might touch each other, depending on the threshold defined for cluster generation, and the maximum amount of pixels, an icon may take. As a consequence, the maximum allowable icon size has to be kept smaller than the maximum cluster coverage. For an imple-

mentation, this consequence has to be kept in mind, and logical queries have to be built in the application logic to prevent unwanted icon cluttering.

The second possibility to represent clusters on a map is to use the shape of the cluster, hence laminar representations. The benefit of this approach is that the reader immediately understands the coverage of each cluster, while point-based symbols give no information about that.

To facilitate the perception of the amount of contained features, the manipulation of the colour value seems to be legitimate, if the size of the cluster area is exactly the same for all clusters. When clustering the features in areas of the same size, the resulting quantities are already *standardised*, and hence, comparable. If the clustering method leads to different sized clusters, this visualization might lead to misinterpretations, as larger areas appear more dominant on the screen, although the density of the contained features is actually lower.

Furthermore, using colour-coded polygons to represent the amount of covered features might lead to cognitive overload in situations when the base map already provides a lot of information, using colourful representations itself.

In summary, polygon-shaped symbolizations for clusters should only be used, if the information gain outweighs the added graphical complexity to the map, especially when manipulating the colour value of the symbols. For many situations, a dynamic combination of both the geometric forms might lead to a more satisfying data visualization. For instance, polygon-based symbolization might be used in small scale overviews to indicate the coverage of each cluster, and point-based icons might be useful for medium and large scale views, as their lightweight footprint does not further complicate detailed map scenes.

3 CARTOGRAPHIC GENERALIZATION

Whenever the situation occurs that a map extent is occupied with too many features, resulting in a too high information density, it becomes difficult to extract the relevant information. The map ends up to be too cluttered, as all the markers cover the underlying map elements (compare Figure 1). In this situation, a generalization mechanism needs to be applied.

In cartography, *generalization* is defined as ‘the process of reducing the amount of detail in a map in a meaningful way [...]’ (KRAAK & ORMELING 2003, 75). To achieve this, several generalization steps can be applied to the object when recording the data (e.g. only include trees that have a defined minimum height). This kind of generalization is called *object generalization*. The other possibility is to apply procedures when processing the recorded data to a cartographic product. Thus, this procedure is called *cartographic generalization*.
















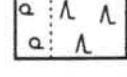





	Procedure	Visualization		
		1:25 000	1:50 000	1:100 000
1.	Simplification			
2.	Enlargement			
3.	Displacement			
4.	Aggregation			
5.	Selection			
6.	Classification			
7.	Exaggeration			

Figure 13: Procedures of cartographic generalization. (Edited; HAKE et al. 2003, 169)

In modern cartography, cartographic generalization can be applied in two ways: first, methods of generalization can be applied when editing the digital model of an object and thus, altering its semantic and geometric resolution (this can also be part of the object generalization). Second, generalization can be applied as a step in the cartographic visualisation process (HAKE et al. 2002, 168).

According to HAKE et al. (2002, 169), there exist seven generalization operators, namely: Simplification, Enlargement, Displacement, *Aggregation*, Selection, Classification and Exaggeration (see Figure 13).

In order to address the problem of map cluttering, the aggregation of features is the most important generalization procedure needed for this research. The algorithms described to solve the issue of overlapping icons on a map use the concept of *clustering*. In traditional cartography, marker clustering is nothing else then the generalization procedure of *aggregation*. It is part of *model generalization*: When zooming out, some detailed concepts (a tree, for instance) disappear to be replaced by less detailed aggregated forms (e.g. a forest).

The following chapters will describe in detail, which methods exist, that declutter point-based map mashups. Basically, three approaches are eligible to fulfil this task: filtering, point aggregation and point displacement. As the focus of this thesis is set on the exploration of clustering strategies, only the aggregative methods will be investigated. In chapters 3.2 and 3.3, two selected clustering methods will be introduced, namely: distance-based clustering and density-based clustering. Subsequently, a comparison will reveal the benefits and restrictions of each of them.

This chapter is concluded with an introduction to the topics of data classification and grid structures, since the understanding of both subjects is crucial to develop a prototype implementation of a grid-based clustering algorithm.

3.1 Decluttering of Point Markers

As described earlier, the vast amount of available data which can be displayed in maps often leads to severe problems. Cognitive overload, bad application performance like a sluggish response to map interactions or long loading times can lead to a frustrating, unsatisfying user experience. HUANG & GARNTER (2012) identified three solutions to address one or several of the above stated issues:

1. Intelligent filtering of irrelevant content (context-based)
2. Icon displacement
3. Icon aggregation (build clusters).

Intelligent filtering methods can be of use in many situations, as the amount of POIs can be reduced dramatically, and the user is enabled to find the desired results in a more efficient and faster way. For example, suppose to have a web portal displaying accommodation in Paris (compare Figure 14). The unfiltered map view shows a lot of unnecessary POIs, as unavailable hotels, or hotels which do not meet specific search criteria like pricing or ratings pollute the map space.



Figure 14: The result for a hotel search in Paris on a booking portal. Unnecessary POIs increase the map complexity.
(Source: BOOKING-I)

The application of sematic filtering would be benefiting. Depending on the situation, this could clear up the view to a varying level. Still, the situation might occur where multiple POIs fall together, leading to overlapping icons. In this case, the location of the icons either need to be rearranged (solution No. 2 above), or the POIs need to be aggregated and replaced by a placeholder symbol.

HUANG & GARTNER (2012, 168) note, that although the icon displacement methods are of use, they might change the location of a specific POI, which clearly cannot be advantageous as the exact location of the marker on the map is essential for a POI to be useful.

Especially on small screens, typically on mobile mapping applications performed on devices like smartphones or tablets, the cluttering of icons forms a serious issue that needs to be addressed (cf. MENG et al., 2005).

BURIGAT & CHITTARO (2008, 22) define the process of *icon aggregation* as follows: ‘*identify clusters of mutually overlapping icons and replace them with special aggregator icons [...]*’. According to the authors, this approach helps to declutter the map, and hence to improve map legibility in freeing the map space without a loss of information. Therefore, the aggregation of data points might be the solution of choice, as it can also be used in combination with filtering methods. In the following chapters, the topic of clustering, and the possible kinds of clustering algorithms will be discussed.

3.2 Distance-based Clustering

Distance-based clustering algorithms are quite common in modern web mapping. There exist various implementations for every major web mapping API, like OpenLayers, Google Maps, or Leaflet.js. The basic algorithm behind this kind of clustering is the so called *agglomerative hierarchical clustering*. According to GAN et al. (2007, 109), the algorithm works as follows:

‘Agglomerative hierarchical clustering starts with every single object in a single cluster. Then it repeats merging the closest pair of clusters according to some similarity criteria until all of the data are in one cluster.’

For the purpose of decluttering marker icons in web mapping applications, the complexity of this algorithm was reduced. The similarity criteria mentioned in the quotation above is set to the distance to a neighbouring point feature, usually measured in pixels. If this threshold is met, the affected features will be added to the corresponding cluster.

Consider a map with many markers, called P_A to P_X (as shown in Figure 15). The clustering algorithm starts with the point feature P_A . As there exist no clusters yet, P_A is set as the location of a new cluster C_A . The same happens to all other point features $P_B... P_F$, as long as they do not touch the distance threshold of C_A . If so, they are added to the cluster which is located closest to them. Following, the location of the cluster centre is recalculated, considering the added feature(s). If a cluster C_X contains only one feature, the symbolisation will not change. Instead, the original marker remains on the map (P_F in Figure 15).

For performance reasons, usually only the markers which are currently visible (that means whose location lies within the visible map bounds), are being processed. The same procedure is repeated after every change of the map zoom, or map extent.

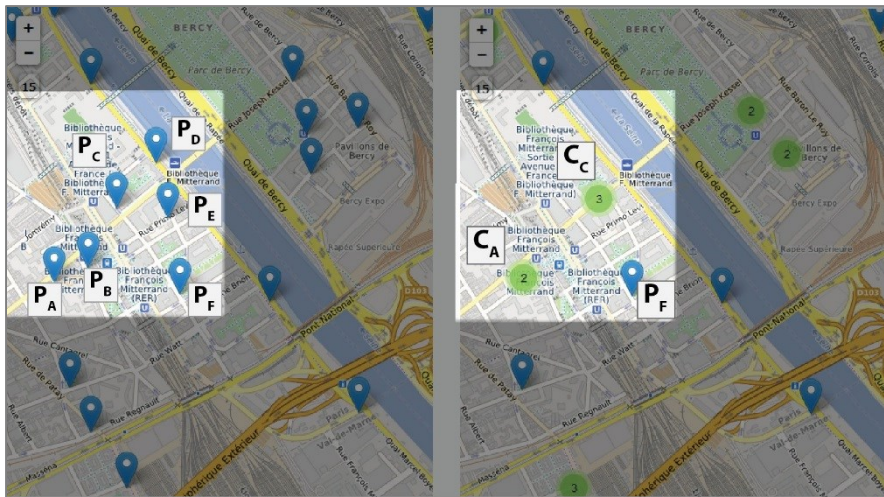


Figure 15: Illustration of the distance-based algorithm (using the Leaflet.js Markercluster plugin).
Base map: © OpenStreetMap contributors)

Every cluster C_X is represented by a special symbolization, which usually holds a label showing the number of contained features. As mentioned above, the location of the icon representing the cluster is moved to the geographical centre of the points contained (like the point C_C in Figure 15) in most implementations. Another common feature is the automatic colour coding of the cluster icon according to the amount of features.

Furthermore, the cluster icons are interactive, meaning that the user can click on them to gather more information about the contained features. Typically, the map zoom is set to the extent of the cluster, showing all markers (or smaller clusters, depending on the density of features).

HUANG & GARTNER (2012, 170) list a few implementations for the Google Maps API, and point out that ‘*the clustering algorithms in them are simply based on the visual overlapping of icons*’ (HUANG & GARTNER 2012, *ibid.*), with varying performance.

One of the **benefits** of a distanced clustering algorithm is the resulting clear map picture, as the amount of features is reduced drastically. Furthermore, most implementations are computing very fast. For example, the implementation for Leaflet.js (*Leaflet.markercluster*) is capable of processing up to 50 000 markers in a decent amount of time.

One of the major **drawbacks** of this clustering method is the behaviour of the cluster icons for each zoom level, as the location of the cluster icons changes unpredictable for the user. As TREVES (2011) states, this can lead to confusion, as many users would actually use the cluster points as landmarks. Without a fixed position, the user is unable to use those points as orientation.

Usage restrictions further apply when trying to visualize temporal developments of point-based features. As the clusters are created solely based on the local distance to neighbouring points, there exists no reference structure, which may allow the comparison of different time stamps.

3.3 Density- or Grid-based Clustering

Point-shaped features can also be clustered into equal sized areas of interest. This approach is referred to as *grid-based clustering* or *density-based clustering* (GAN et al. 2007, 209). The general processing steps are described by GAN et al. (2007, *ibid.*) as follows:

1. *Creation of a grid structure on the data space,*
2. *Calculation of the cell density for each cell,*
3. *Cell sorting based on their densities,*
4. *Identifying cluster centres,*
5. *Traversal of neighbour centres.*

In cartographic applications, the algorithm is altered in a way that steps number three and four can be omitted. Concerning step one, ARNBERGER (1977, 141) states that the shape of the grid cells may only be of an area-wide, equilateral polygon kind. More specifically, that includes only triangles, squares or hexagons. While hexagon-shaped cells may improve the aesthetics of the data visualization, square-shaped cells are easier and hence, faster to compute.

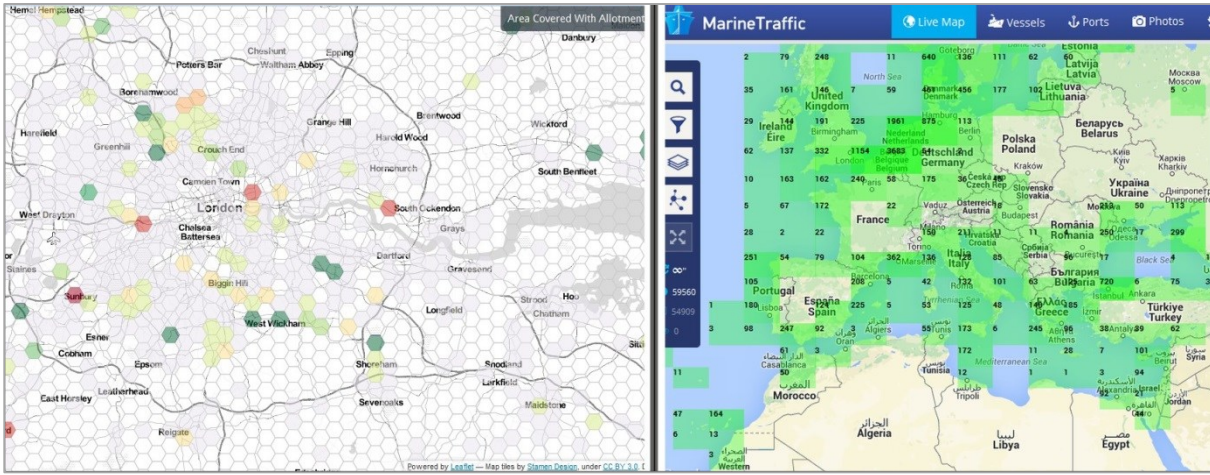


Figure 16: Examples of a hexagonal-shaped grid (left) and a square-shaped grid (right). (OSMATRIX-1 & MARINETRAFFIC-1)

After laying the grid over the map, the point features are allocated to the corresponding cells (second step above). The cluster cells can be represented using either the shape of the cell (as shown in Figure 16), or using point-based cluster symbolizations, similar to the ones described in chapter 3.2. The location of the cluster icon may be the centre of each cell. Alternatively, the location can be calculated to reflect the allocation of the features within the cluster cell using for example the *arithmetic mean*. Given a set of features a_1, \dots, a_n , the formula for the arithmetic mean is:

$$A = \frac{1}{n} * \sum_{i=1}^n a_i$$

(To receive the mean location of the cluster icon, this calculation has to be done both for the latitude and longitude part of the coordinates).

When using the cell polygons as cluster representations, the binning of the cluster values can help to improve the data legibility, as shown on the left in Figure 16 (this step corresponds to step three in the algorithm definition stated above). More information about symbolizations can be found in chapter 2.5.3.

For each zoom level, or scale level, there is a defined grid size. In an ideal case, the cell size is scaled up and down by the factor two, similar to the tile size of raster tile layers. In order to improve the computing performance, only the currently visible features should be considered.

One **advantage** of the grid-based algorithms is the ‘*significant reduction of the computational complexity, especially for clustering very large data sets*’ (GAN et al. 2012, 209). However, as most map mashups applications do not contain more than a few thousand points, this might not be the most important feature for web mapping applications.

A more valuable gain may be seen in the existence of the rule-based alignment of the cluster symbolizations. Through binning all point features in equal-sized cells, the spatial distribution of the data can be perceived more efficiently, especially when the zoom of the map is altered. The impact of this behaviour is investigated in the survey, see chapter 8.

The grid-based algorithm is highly eligible for the application in thematic mapping (e.g. for choropleth maps), as the data displayed in standardised cell areas is strictly comparable. How-

ever, the map projection should be equal-area, in order to provide non-distorted cell representations. Otherwise, the map might be misleading to the reader, and the data will be misinterpreted (cf. MUEHLENHAUS 2014, 147).

Related to that, the regular grid structure offers the possibility to visualize temporal changes in data in a comprehensible way, or to compare the spatial allocation of the data independent of (more or less arbitrary) administrative boundaries.

One major **drawback** of the grid-based clustering approach is, that spatially concentrated feature agglomerations may not be accurately described by the clustered visualization as the bounds of the grid cells are not aligned in a *semantically meaningful way*, as shown in Figure 17 on the left. Depending on the chosen cell size and type of symbolization, the representation of the data may be misleading (cf. MAHE & BROADFOOT 2010).

A solution to this issue might be to calculate the position of the label for each cell using the arithmetic mean of the contained features, as shown on the right side of the illustration, and described above. The comparability of the grid structure is still given, but now the spatial concentration of the features within the cell is indicated by the position of the labels.

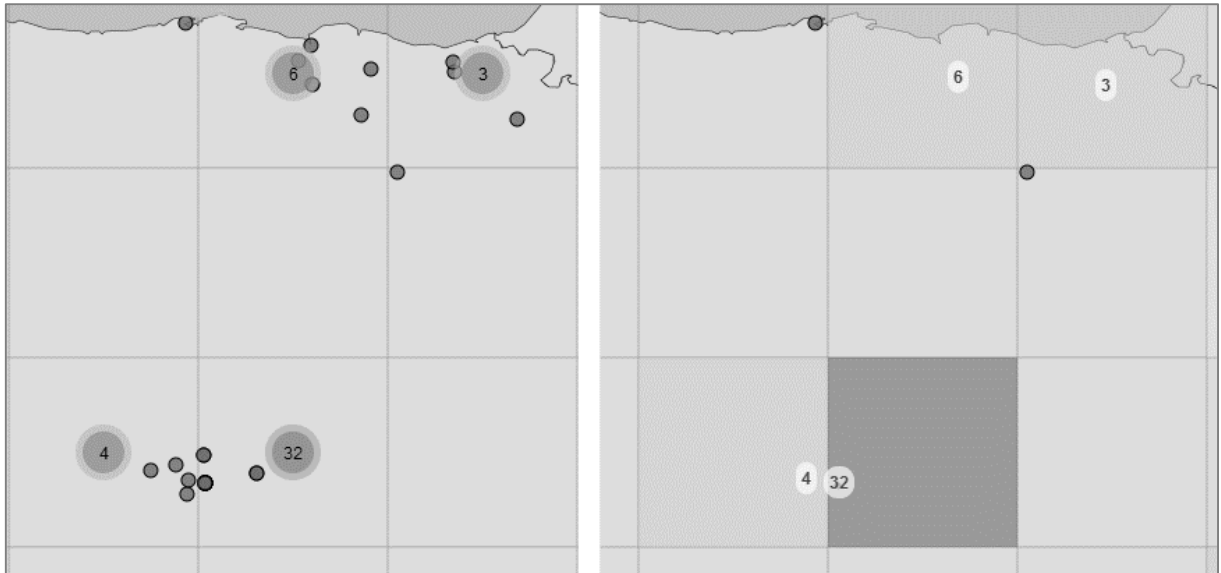


Figure 17: Grid-based cluster visualizations. Left: icons positioned in the centre of the cell, right: adapted location based on distribution of the underlying features.

Another serious drawback is caused by the Web Mercator projection used by popular base maps: the geometry of the cells is distorted in the North-South direction, depending on the location (the more distant from the equator, the higher the distortion). This issue can be addressed in two ways. Either a scale factor, implemented into the algorithm, can be used to flatten down the north-south distortion, or custom made base maps with an equal-area map projection can avoid the distortion. Especially for thematic mapping purposes, the latter solution should always be preferred (cf. MUEHLENHAUS 2014, 146).

3.4 Comparison

After the exploration of both the distance- and the density-based clustering approaches, these two methods will now be compared and discussed in the following section.

When comparing both clustering methods, one major drawback becomes apparent: The disadvantage of both clustering strategies is the *lack of contextual awareness*. The algorithms are completely unaware of the situation shown on the underlying map – they simply alter the location of the POIs according to the implemented rules. Thus a semantically meaningful clustering (for example, to aggregate features close to public train stations, or to weight certain POIs according to the distance to a relevant feature) cannot be achieved. The root of this problem seems to be the ‘dumbness’ of the static base map, which provides no additional information besides the RGB values of the pixels.

Again, major improvements could be accomplished with further developments of vector-based base maps or through context-based extra calculation steps performed server-side, if the geospatial data of the base map is available in a suitable format.

Apart from that, the strength of the distance-based clustering method lies in the application to POI driven map mashups, as it is fast and easy to integrate – which might be one major factor for mashups that are usually not created by experts. It furthermore solves the icon cluttering problem in any situation, as the radius of the clusters cannot be smaller than the size of the cluster symbolizations.

In contrast to that, the distance-based approach seems to be unsuitable for the usage in animated maps, as it lacks a regular structure. The dynamic calculation of the cluster locations hinders the comparison of temporal developments.

On the other hand, the application of a grid-based clustering method can be of advantage in exactly those scenarios: The regular grid structure allows the comparison in the means of space and time (see also chapter 3.6). Another advantage is the flexibility given by the grid form: either a point-shaped cluster representation can be chosen, or the cells may be used as cluster symbolizations, allowing the visual highlighting of the feature allocation by applying a colour coded classification (see chapter 3.5).

The downside of this approach might be a lack of accuracy, in terms of feature allocation. Thus, the grid cell size is crucial for a meaningful result. If it is chosen too big, the cluster areas will not indicate the actual distribution of the features. If it is set too small, the resulting map could be cluttered up with cluster symbolizations.

To conclude, it lies in the responsibility of the map creator to choose the right agglomeration technique, depending on the topic and situation he wants to communicate to the reader, and on the type of map he chooses for this purpose. The evaluation of the survey will show, how the questioned audience perceived and valued each approach during the simulation of real world web mapping situations (see chapter 8.3).

3.5 Data Classification

In literature, *data classification* is defined as ‘*the process of analysing the distribution of values in a dataset and grouping them based on their location in a histogram*’ (MUEHLENHAUS 2014, 148). The main purpose of data classification is to conveniently organize the data before displaying them, thus creating a clearer map image (KRAAK & ORMELING 2003, 116). It is therefore a means of generalization, as listed in Figure 13.

Before data can be classified, it has to be known of which type the data is, so that necessary definitions and statistical measurement scales can be applied. KRAAK & ORMELING further note that setting class definitions and object descriptions is not a trivial task, as the definition of an object can be ambiguous in some cases, and unclear in practical use. For instance, the definition of a car seems to be straightforward, but the definition of a traffic jam already leaves more room for interpretation.

Furthermore, the data might be of continuous or discrete nature, it might be nominal, ordinal, or interval-based. Depending on the data, the operational processing steps, the appropriate type of map, and the kind of symbolization differ. Interval data, for instance, is best visualized using choropleth or isoline maps (KRAAK & ORMELING 2003, 116), and classified using statistical approaches like natural breaks, quantiles or equal intervals. Whichever method is chosen, after the classification, the data groups are assigned to predefined colour codes which will then be used to style the data representation on the map.

Research has shown that human beings can hardly process more than seven discrete information units, otherwise cognitive overload will hinder the user to perceive the theme of the map on a glance (KRAAK & ORMELING 2003, 116). As a consequence, maps should not show more than five classes (MUEHLENHAUS 2014, 148), as a higher number of classes, and therefore colours, cannot be distinguished anymore.

Especially for thematic mapping purposes, data classification, or *binning*, is essential to communicate the raw data to the user in an efficient manner. The cartographic generalization procedures used for the processing of *point-based* raw data are a combination of feature aggregation and classification, as shown in Figure 13.

3.6 Grid Structures

In this chapter, the nature of grid structures, which are used in the density-based clustering algorithm, will be explored. The purpose of a grid is not restricted to just aggregate POIs into equally sized cluster cells. This attribute of providing regular cells can serve for thematic mapping purposes, as well. ARNBERGER (1977, 141) notes that administrative units, which are often used to classify thematic data, are rather unsuitable due to their heterogeneity in area, form and structure. Using those units, an objective comparison of regions is hindered. Also, temporal developments cannot be shown, as administrative units are often altered (for example united, split or erased), leading to different geometric structures over time.

Consequently, a statistic raster structure can provide comparability in means of space and time. In laying an equal-sized grid over the map, a homogenous structure is created, allowing a regional and temporal comparison of the cell-based contents. ARNBERGER (1977, *ibid.*) further lists the following advantages of this approach:

- The regular divisibility of the grid cells allow automated generalization for various map scales
- Data can be compared on an international level using transnational grid systems
- Particularly applicability for cartographic automation.

Having a map with a high load of point-shaped information, the transformation of that information to a more abstract, generalized level can improve map legibility. The use of regular grid structures will remove the noise caused by the point-shaped features, and thus reduce the complexity of the map. The perception of relevant information is eased and the cartographic communication process is more efficient.

4 EVALUATION

Whether a map, or parts of it, are working as expected, or not, can only be determined by asking potential users. This feedback is part of the cartographic communication cycle (chapter 2.5.1), and should be used to improve the quality of cartographic products. During a design cycle, usability tests should be performed as often and as early as possible (KRUG 2006, 134ff.). By doing so, unpredictable, unwanted and unexpected behaviour can be detected and thus be eliminated on an early stage of the development process. Usability tests are performed using *empirical methods*.

In the implementation level of this thesis, the feasibility of grid-based clustering solutions will be evaluated using several methods. While the theoretical part was completed by analysing related projects, best practices and cartographic design principles, the empirical part will be covered by usability inspections and a survey. The eligibility will basically be investigated using two empirical methods: *Heuristic evaluation* and *deductive tests*. While the first is part of usability inspections, thus requires the input of experts, the latter can be used in usability testing, hence the survey. In the following subchapters, these two methods will be discussed in detail.

4.1 Heuristic Evaluation

The idea of heuristic evaluation is to ‘*look[ing] at an interface and try[ing] to come up with an opinion what is good and bad about the interface*’ (NIELSEN 1994, *ibid.*). Heuristic evaluation therefore is a *formative* method to examine the usability of a user interface, using the opinion of experts rather than the experiences of real users. The author further states that a minimum of five participants is needed to detect at least 75 % of possible usability problems. The advantage of this method lies in the ability to integrate usability testing into the development process. Problematic software behaviour or unclear designs can be ruled out already before showing the product to real users.

For this thesis, a modified version of heuristic evaluation has been used. During early design stages of the prototype implementation, three experts of cartography and geography have been asked to give their feedback to the prototype. By doing so, several design flaws could be detected, and the usability test performed with ‘real’ users could be performed with an *already optimized version* of the prototype.

4.2 Deductive Tests

According to SARODNICK & BRAU (2011, 163), deductive tests are used to monitor the performance of a single system or to compare several systems with each other. Applied to web map design, the feasibility of different visualization techniques can be tested. In using deductive tests, insights for potential improvements can be gained. Deductive tests belong to the technique of usability testing. NIELSEN (1994, 165) describes them as a crucial part of the investigation, as they allow the direct input of real users, and hence offer insight how the system is used in practice, and how user experience the interface and visualization. As cartographic communication is a circular process, this kind of assessment offers the valuable possibility to integrate feedback into improved versions of the mapping product.

4.3 Scenarios and Use Cases

‘Scenarios are arguably the starting point for all modelling and design’

(SUTCLIFFE 2003, 323).

No matter if in software development, requirements engineering or in the general design of systems, scenarios are used to start the development process of a product, as they are usually derived from real world problems. A scenario describes real world experiences or stories, and breaks them down to models and specifications, thus making them easier to handle (SUTCLIFFE 2003, 320). In doing so, two functions are covered by scenarios: first, the real world issue which has to be modelled, is recorded and written down, and the problem is sketched. And second, specifications, models and use cases can be derived, enabling to put the issue on a more abstract level. However, a scenario might also serve as inspiration for the design process when using development methods based on prototyping. Related to that, scenarios can be used to describe tests in evaluation methods (SUTCLIFFE 2003, 320).

In summary, scenarios can be used in many different ways, and their content, scope and role in the design process of a product varies. For the scope of this thesis, scenarios have been used to create test situations for the performance evaluation in chapter 7.3, and for the identification of requirements and components of the prototype, as shown in the chapters 7.1 and 7.2.

A *use case*, in contrast, is a sequential description of possible events and actions, carried out by a role and a system. The role is often called stakeholder, or actor and can be human, time or an external system. COCKBURN (2001, 15) defines a use case like that:

‘A use case captures a contract between the stakeholders of a system about its behavior [sic!]. The use case describes the system’s behavior [sic!] under various conditions as it responds to a request from one of the stakeholders, called the primary actor’.

The differentiation between a use case and a scenario is not always clear, as both terms and concepts are closely related. As SUTCLIFE (2003, 320) states, use cases can be derived from scenarios, as they usually form a sequence of actions. Use cases can be defined using narrative texts, which describe the situation, actors and the system behaviour in a detailed form. Furthermore, use cases might also be described on a graphical level, using a modelling language, like the Unified Modelling Language (*UML*).

However, COCKBURN (2001, 224) states that choosing UML to express a use case certainly will add more complexity to the process. The author therefore recommends to use clear, understandable text to picture a use case. In doing so, room for misinterpretations is eliminated, resulting in a more efficient development process. The use cases derived for this thesis are written in a narrative text form.

5 LEAFLET.JS

This chapter is dedicated to giving an introduction to the web mapping library *Leaflet.js*, which is used for the implementation of a grid-based clustering algorithm in this thesis. The main characteristics of the library as well as the plugins, which are crucial for this research, are illustrated.

5.1 The Core Library

The web mapping library *Leaflet.js*, which is written in the programming language JavaScript, is one of the most popular tools used for building web map applications. The initial release of the library was in 2011. Since then, several releases followed, as it was - and still is - consequently developed by an active community of volunteer developers.

The core advantages of Leaflet.js are its straight forward usage, its light footprint and the modern user interface. In addition, it is also published as *open-source* and therefore easy to extend. The library can be downloaded on the project website www.leafletjs.com. The core library itself offers only a *basic amount of features* out of the box, but the functionality can be extended by using plugins, which are provided by a vivid community of developers.

The key features of this web mapping tool are the *native support* for various raster and vector layer formats, such as WMS and (geo) JSON, as well as the standard vector forms point, line, polyline, polygon and circle. It is therefore easy to mix multiple data sources in several layers on the same map.

One difference to contemporary desktop GIS and other web mapping libraries like OpenLayers is the way, how vector features and layers are defined in Leaflet.js. Traditionally, a vector-layer represents a *set of features* which have similar attributes or semantics, like a road network or boundaries or point markers. The hierarchy therefore traditionally looks like this: in the map object, there exists a layer object, which covers all corresponding single features.

In Leaflet.js, every single vector feature is represented as a *separate* layer object. However, these objects can be grouped into a Feature Group or a Layer Group and can then be accessed as one object.

5.2 Proj4Leaflet Plugin

Leaflet.js natively supports only three map projections: Spherical Mercator (equal to the *Web Mercator*), Mercator and a Plate Carree projection (LEAFLETJS 2014a). It can also handle various projections on the fly, when using the *Proj4Leaflet* plugin. Therefore, the projection definition has to be set in the source code of the map. The definitions of common projections which have an EPSG code can be found on web services like <http://epsg.io>.

This feature is relevant for the research of this thesis as the predominant Web Mercator projection is not always the best choice when medium-scaled maps are used for thematic mapping (as shown in chapter 2.3), as well as for the data visualization using the grid-based clustering (see chapter 3.3).

5.3 MarkerCluster Plugin

The Leaflet.MarkerCluster plugin is an implementation of the distance-based clustering algorithm introduced in chapter 3.2. It offers dynamic clustering of features, which means that features can be added or removed to the map space during runtime. The plugin furthermore offers interactive cluster symbolizations. It is possible to display the coverage of each cluster, as well as to adjust the radius of each cluster. Furthermore, the symbolization can be customized using Cascading Style Sheets (CSS) rules.

This plugin is used in the survey (chapter 8), to create a competing visualization technique for the use cases designed in chapter 6.

6 USE CASES

Reconsidering the definition of use cases and scenarios given in chapter 4.3, the purpose of this chapter is to identify possible real world scenarios, and transform them into a sequence of actions, hence to create use cases, which will serve to test the eligibility of point clustering approaches. Chapter 6.1 is devoted to the identification of plausible use cases. To anticipate, two use cases could be identified. They will be worked out in the following chapters.

6.1 Identification of Possible Use Cases

In this section, possible use cases for a meaningful application of the grid-based clustering method will be discussed. The use cases will be applied in the survey, which is presented in the next chapter. According to SARODNICK & BRAU (2011, 164), use cases used in usability evaluations have to be as close to reality as possible, in order to identify issues in the application later. As a consequence, there have been two use cases identified, which can be considered to be common in modern web mapping applications.

Furthermore, it is essential to understand which aspects of map usage are important for typical users. The focus of the research will lie on the obstacles and flaws in navigating on a map filled with a huge amount of point-shaped features. Plausible questions to ask might be:

- How difficult is it for the user to understand the concentration of features and find the relevant information they are looking for?
- Are they using the cluster markers as landmarks as they manipulate the map extend or zoom? If so, cluster points which change their location in an unpredictable way during zooming could lead to more confusion.
- Is it easier to navigate the map when colourized cluster-polygon features are used instead of markers?

The key findings of the enquiry of existing research (discussed in chapter 1.5) appear to be *the cluttering problem of icons or markers*. In many papers, the considered scenario was similar:

Typically, a user navigates on a map mashup featured website, in search for POIs related to a specific topic. Conceivable topics might be restaurants, hotels, touristic places or bike sta-

tions. As a result, he receives a map showing the unfiltered POIs within an area of interest (*AOI*). Depending on the amount of features and the amount of space (that is, map scale, screen resolution, marker size and map size) available, the resulting map legibility is either good or bad. If the service is designed smart enough, filtering options may solve the cluttering of icons (cf. HUANG & GARTNER, 2012). In any other cases, or in addition to the filtering, a clustering algorithm should be applied on the thematic layer. Thus, this situation will form the first use case to be examined by this thesis. It will be developed in chapter 6.2.

Another rewarding application of the grid-based clustering strategy can be found in thematic web mapping. Features are usually classified using administrative boundaries which often do not represent the real situation or characteristic of the data. The allocation of certain features like mobile network coverage or tweets created by twitter users doesn't end at some artificial boundaries, thus their characteristic distribution may not be mapped correctly. Additionally, when dealing with point-shaped data, the representation of point data distributions using unfiltered point symbolizations might impede the ability of the map user to identify patterns and relevant areas.

Using an equally sized grid, data originated from different regions, be it administrative divisions, or any other territorial entity, can be compared with each other easily and furthermore, even temporal developments can be explored in an efficient manner. The second use case will reflect these ideas, as described in chapter 6.3.

6.2 Use Case 1: Locating a Bike Station in Paris

In this use case, the following scenario is being simulated: A tourist visits the city of Paris, France. He plans to use the public bicycle sharing service, which is offered in this city. He wants to take a tour through the city, starting at the Metro Station 'Gare d'Austerlitz' in the centre of the city. Therefore, he tries to figure out where the bike stations are located, so he can pick up and later return the bike. To get this information, he visits the official website of the bike rental (<http://en.velib.paris.fr/>). Unfortunately, the stations there are all displayed as single markers, so as soon as he zooms to a medium scale of the map, to get an overview, the markers cover up most of the underlying map space. This is rather confusing (see Figure 1 on page 4).

Reflecting the theory of the *information seeking mantra* (overview, zoom and filter, details-on-demand) which was introduced in chapter 2.5.1, the sequence of actions might look like this: The user starts with an overview over Paris on a small map scale. This map should use a generalization technique (clustering approach) to present the distribution of the stations in a visually optimized way.

After figuring out, where the desired location is originated, the user drills down to medium and larger map scale levels to progressively find the information needed, while the clusters are reclassified on each zoom step. Lastly, the detailed distribution of bike stations using the original markers should only be displayed on a big map scale.

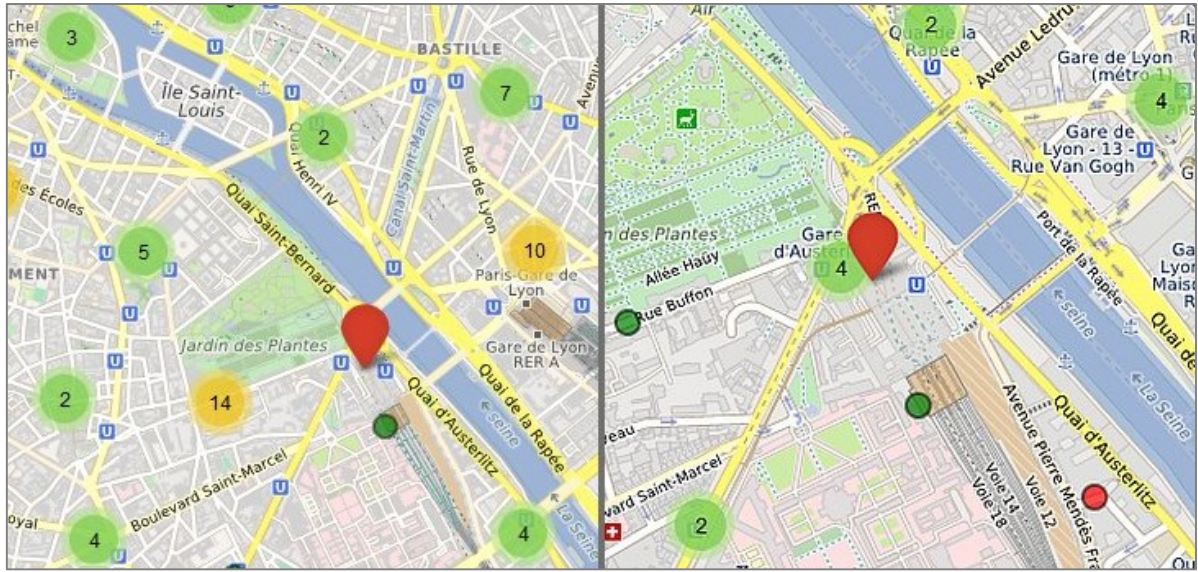


Figure 18: Results of the Markercluster algorithm. Left: Zoom Level 14, Right: Zoom Level 15. (Map source © OSM contributors).

In a first attempt to declutter the map, the same scenario is being simulated using a map with the distance-based clustering technique offered as a Leaflet.js plugin (*Leaflet.Markercluster*). Using the map with a medium scale (zoom level 14), the cluster algorithm shows 14 stations close to the ‘Gare d’Austerlitz’ station, but none directly there. If the map is zoomed in once, the situation completely changes: now there are four stations exactly where the user needs them. This can be rather confusing, as well (see Figure 18).

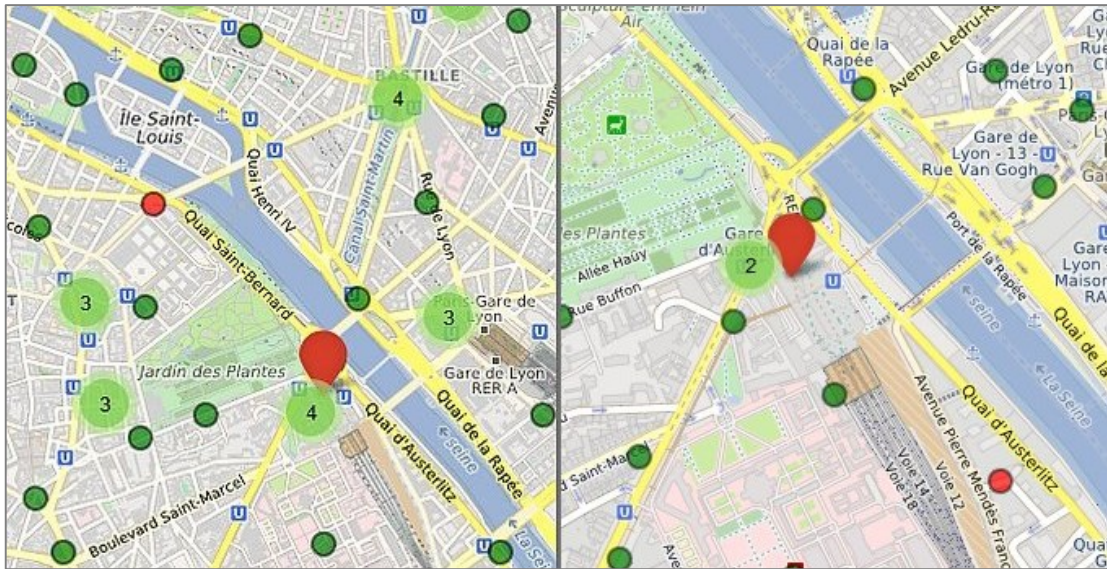


Figure 19: Result of the grid-based algorithm using dynamic cluster symbol positioning based on the mean location of contained features. Left: Zoom 14, Right: Zoom 15 (map source: © OSM contributors)

In the second attempt, the same scenario again is being simulated using the grid-based cluster algorithm developed in this thesis (see chapter 7). At zoom level 14, there are four bike stations exactly at the ‘Gare d’Austerlitz’ station. When zooming in, they also change their location, but not as much as in the second attempt (compare Figure 19). Nevertheless, it has to be noted that the result of the clustering strongly varies, depending solely on the location of the

original POIs, and on the chosen cluster radius or cell size. For another location, the clustering algorithms could lead to different results.

Lastly, the grid-based algorithm is again used for a third attempt. In this setting, the cells of the cluster were chosen to form the representation of the clustering results. Hence, a laminar symbolization was chosen. The clusters have been classified by the amount of features they contain, and the cells have been assigned corresponding colours, as shown in Figure 20.

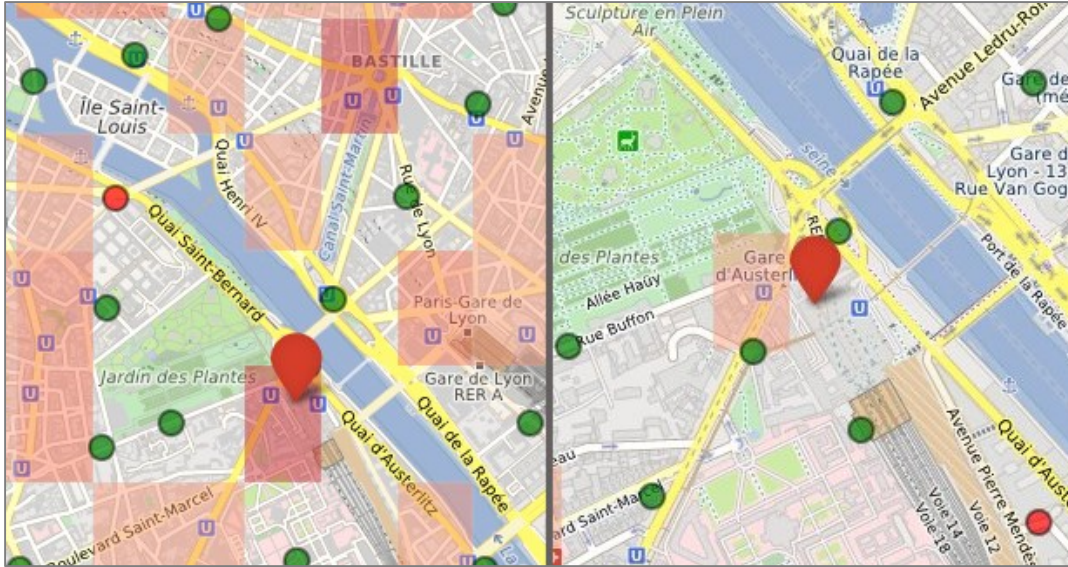


Figure 20: Result of the grid-based algorithm using laminar symbolizations. Left: Zoom 14, Right: Zoom 15 (map source: © OSM contributors)

In case of the grid-based algorithm, the size of the grid cells determines the accuracy of the result. If the chosen cell size is too large, the resulting cluster areas will not represent the situation adequately. However, a too small cell size would lead to marker cluttering, which should be avoided as well. As a consequence, a trade-off has to be found.

This use case is deployed in the survey (chapter 8) and is built up like follows: the base map is derived from OpenStreetMap data, the tiles are rendered in the Web Mercator projection. The thematic data (the bike stations) are accessible using an API (CITYBIKES-1). The API returns a JSON encoded list of all available stations with detailed information, which then is processed using the Leaflet geoJSON-Layer. The distance-based implementation was used with the default settings. The grid-based solution was used to create the second and third try. All three attempts have been recorded with a screen recording software, starting with zoom level 12 (medium scale), and increasingly zooming to level 17 (detailed view of the map). Each zoom change was delayed for approximately five seconds, to allow the user to perceive the new situation. The evaluation of this use case can be found in chapter 8.3.2.

6.3 Use Case 2: Visualization of Temporal Change of Data

While the first use case simulated a map mashup with POIs, this use case is meant to address the field of thematic mapping. In particular, temporal changes of point-based geodata will be simulated. No matter how data is collected, it always represents a snapshot of the observed reality, while the sampled phenomena usually change in a more or less dynamic way over time (for instance traffic flows, earthquakes or volcano eruptions). The mapping of these temporal changes in point-based phenomena therefore forms an interesting and realistic scenario for the application of a clustering strategy.

In this second use case, the temporal development of fictive incidents throughout Europe is being monitored for a time series from 2001 to 2010. The dataset was created, edited and converted into the geoJSON format using the GIS software *Q GIS 2.4.0*. A total of 19 884 features were created using the *random points* tool offered in Q GIS. The created point features have been modified by hand to simulate a meaningful distribution close to populated areas. In addition, a year attribute value ranging from 2001 to 2010 was assigned to each feature. The focus of the time development was set to Central Europe. The scenario now will be built up as follows:

The map interface contains the map view, with a vector layer forming the base map and the dataset loaded into an *L.geojson* vector layer. A control button is provided to start or stop the animation of the temporal development. The dataset for the base map was taken from *naturalearthdata.com*, which offers free to use geodata. As this map does not contain any raster-based layers, the map projection could be set to the equal-area *Plate Carree*, resulting in a square-based grid for the grid-based approach. By doing so, the visualization follows the best practices described earlier in chapter 2.3.

The aim of this use case is to explore, which clustering technique leads to an easier understanding of the data distribution. Therefore, the grid-based approach will be provided twice: One version will visualize the clusters using the well-known point markers symbolization, the alternative version will feature a colour coded visualization using the geometry of the cells themselves.

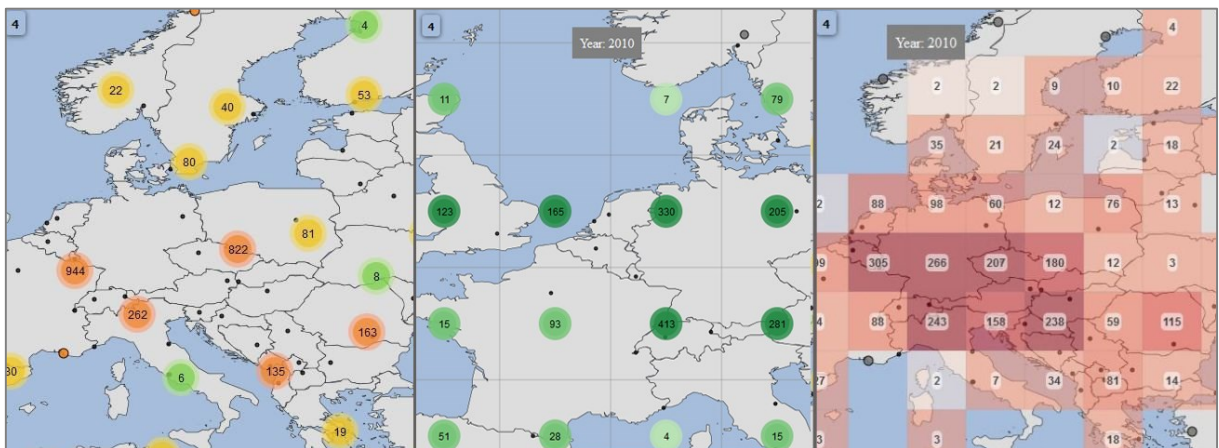


Figure 21: From left to right: MarkerCluster, grid-based markers, grid-based polygons.

A snapshot of the animation using the three different data visualizations is illustrated in Figure 21. It is quite obvious, that the grid-based approaches lead to a more structured representation of the data, compared to the distance-based method shown on the left.

Similar to the first use case, the animation of the temporal development was recorded using a screen-recording software. The resulting video clips are about 30 seconds long, as each year was shown for about four seconds. The method of animation chosen for this use case was the so called ‘stop-frame animation’, which means that the animation is built up using only individual frames, without the application of tweening between the frames. Due to technical restrictions of the used framework, and as this type of animation is typically the option chosen for time-series raster data (MUEHLENHAUS 2014, 174), this technique was taken for the use case.

This use case was deployed in the survey as well, the results of the evaluation are documented in chapter 8.3.3.

7 IMPLEMENTATION OF THE PROTOTYPE

In this section of the thesis, the insights gained from the previous chapters will be used to build the prototype implementation of a grid-based clustering algorithm in a JavaScript web mapping library. Using the developed use cases, a set of requirements for this implementation will be worked out. Following, the design and implementation of the prototype will be explained. This chapter will be concluded with a section about performance testing: As the first part of the evaluation of this approach, the algorithm will be assessed from a technical point of view. In the next step, the algorithm will be used in a small online survey to evaluate the performance and the feasibility of the grid-based clustering approach using the opinions of users.

The algorithm used to implement the clustering strategy is based on the principles described in chapter 3.2. The web mapping library Leaflet.js (see chapter 5) in the latest release version in the time of writing (0.73) will be used as the source library. The class *L.FeatureGroup* will be extended to form the new grid-based clustering extension. The source code is available online on GitHub and can be accessed using the link <https://github.com/andy-kay/Leaflet.GridCluster>. The plugin is published under the MIT license.

7.1 Requirements and Settings

The use cases defined in the previous chapter, as well as the examination of existing projects, research and cartographic design principles lead to a set of requirements, which the implementation of a real time grid-based clustering algorithm should fulfil.

Without doubt, the user experience heavily depends on the performance of an application. If lags or long waiting times during the usage occur, and the application does not provide a feeling of responsiveness, users will most likely dismiss the service and look for a substitute. However, it is not easy to define the critical response-time, upon which users will accept to use a software or service. According to MEIER et al. (2007, 121), poor performance is a subjective impression, which is based on the individual experiences and preferences of every user. For instance, if a person was using a competing software product or service, he will expect the new approach to respond in a similar way, and will probably not accept significant longer loading or processing times.

For the development of an alternative clustering solution, this demands a resulting prototype that is capable to respond in similar time spans, as already existing solutions do. Thus, as a precondition, the clustering has to be done within a reasonably short time. In an ideal case, the performance will meet the already existing distance-based cluster strategies. The responsiveness can be measured with the developer tools of a modern browser in various use cases. The evaluation of the prototype will be discussed in chapter 7.3.

Besides the requirement for a certain computing performance, other technical aspects have to be considered: The implementation has to be compatible with the Leaflet.js core library, and needs to be able to work with various map projections, at least with the set of map projections which are offered natively by Leaflet.js (Web Mercator and Plate Carree). As leaflet internally always computes latitude and longitude, this is a rather trivial requirement.

The `L.FeatureGroup` class represents a group of features, which means in Leaflet.js, map layers. Every single marker is a layer. This class already offers inbuilt methods to add, remove and retrieve single features, or a set of features. These functions are needed in the prototype as well, as the implementation needs to offer methods to add and remove raw features to the cluster layer. As a consequence, the inbuilt methods of the base class have to be extended or overwritten by custom program code, as a JavaScript object will be needed to keep track of the unclustered features.

The newly written class has to be able to process the typical point-based layer types in Leaflet.js, namely: `Markers`, `CircleMarkers` and `Circles`. Only features which are visible on the current map extent should be processed, off-screen features shall be ignored. Every change of the map zoom or a panning of the map will trigger another clustering circle.

The grid has to be defined using a given extent, the size of the cells, and alignment. Therefore, a method is needed, which gathers the current map bounds and enhances them according to the cell size and the alignment. Furthermore, the cell size needs to be alterable. It has to be recalculated on map zoom changes.

The symbolization of the cluster cells should follow the best practises discussed in chapters 2.5 and 3.5. Both point-based and areal symbolizations have to be supported, and in order to classify the data and colour-code the symbols, the possibility define classes and colour sets needs to be implemented.

The implementation should feature a set of customization options that allow the adaption of the algorithm to specific project needs and add flexibility to the prototype. These options can be distinguished into functional settings and styling settings. The first group consists of options to set the initial cell size, as well as the zoom factor, which should be applied for every zoom change. Additionally, the maximum zoom level, until which the clustering should be performed, has to be specified, as well as the minimum amount of features, from which the cluster cell will be created.

For the latter group, options to control the appearance of the cluster representations as well as the grid have to be implemented. An overview over the available options to customize the prototype is shown in Listing 3.

7.2 Design

After gathering the requirements for the prototype, this chapter is meant to document the design and implementation of the prototype. Recalling the definition of a grid-based algorithm given in chapter 3.3, the implementation has to perform the following steps, every time a new clustering cycle is requested by an event like a zoom change or map pan (see Figure 22):

In the **first step**, the grid is created over the visible map space, according to the options specified during the initialization. In particular, the size of the grid will be calculated based on the scale factor, the current zoom level and the initial grid cell size.

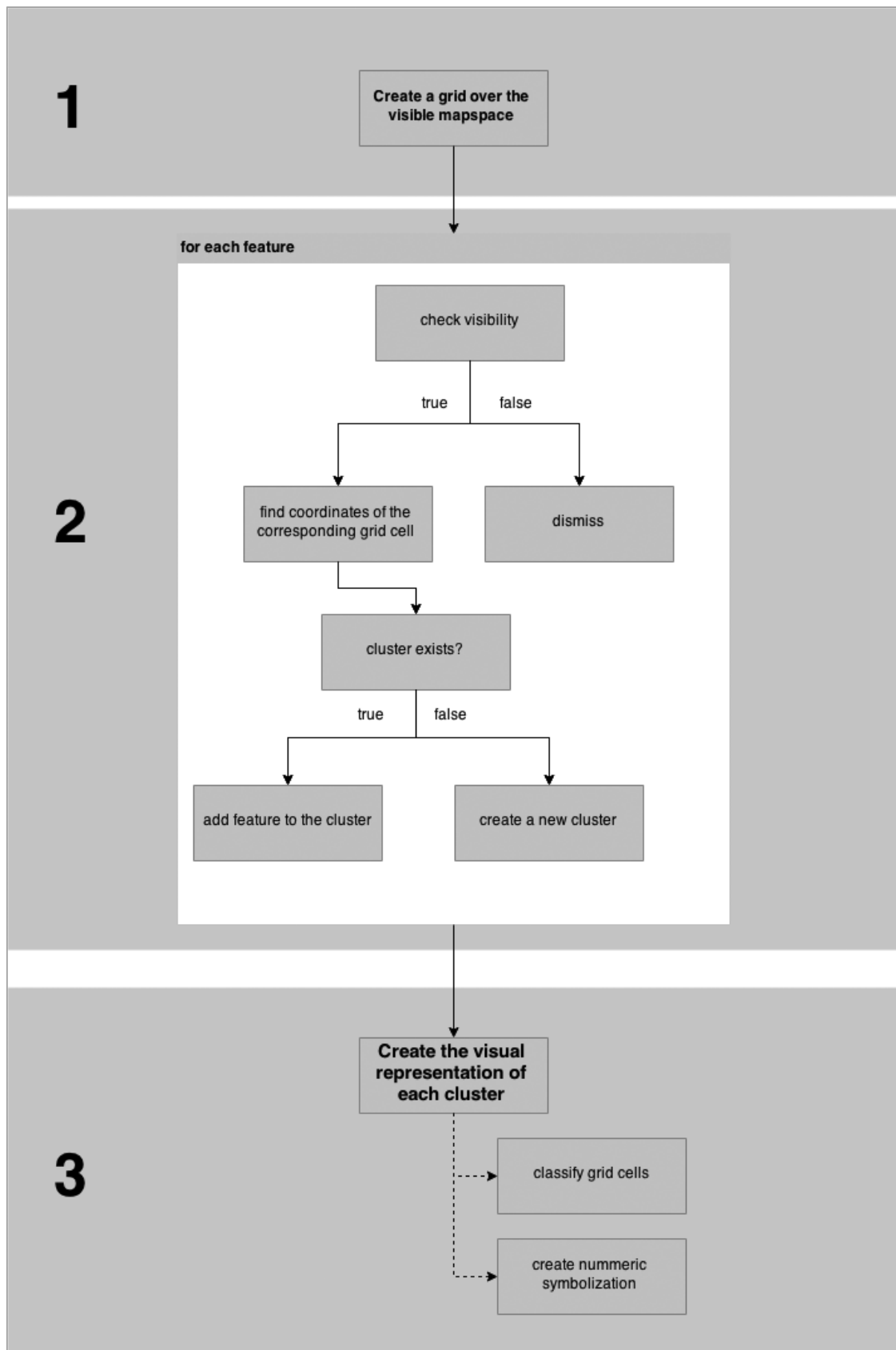


Figure 22: Workflow of the grid-based clustering algorithm

Whenever features are added to or removed from the layer, the **second step** of the routine is initiated and a method is called to check for each of the features in the group a set of conditions. The first task is to exclude features out of sight. Only visible features will be considered, thus the next step is to test if the feature lies within the currently visible bounds of the map space. If so, the algorithm will incrementally try to find the latitude and longitude of the grid cell that contains that particular feature. On success, the feature will either be added to an existing cluster, or if no cluster exists yet, the routine will create a new one and add the feature to it. Each cluster is represented by a JavaScript object featuring a unique identification number, which is created using the longitude and latitude of the cell centroid, as shown in

```
clusters[clusterID] = {  
    count : 1,  
    latLng : centroid,  
    features : [feature],  
    polygon : polygon  
};
```

Listing 1: Definition of a cluster cell object

Listing 1. The cluster object further holds an array with the original features, a count attribute, which will be updated on every change of the number of features, the centroid coordinates and a polygon geometry representing the shape of the cell.

In the **third step**, the visual representation of each cluster cell is created, according to the options given by the map creator. The visual output can be performed in the form of point-shaped cluster symbolizations, or in using the shape of the grid cells as laminar symbolizations. The grid cells can be classified optionally, using an array of colours. The length of the provided colour array determines the amount of classes which will be created. As a future improvement, more advanced methods to calculate the class thresholds, like equal intervals, quantiles and the like could be implemented, as well.

Furthermore, a numeric symbolization of the amount of features covered by the cell may be added. The labels can be positioned within the cell using either the centroid of the cell, or the calculated mean location of the contained features. The algorithm to calculate the mean location is described in chapter 3.3.

The styling of the marker symbolizations can be customized and controlled using CSS rules, as depicted in Listing 2. The default colour scheme uses a set of three different greens, varying only the value of the colour. The thresholds to distinguish the cluster symbols into three classes are ten features and 100 features. These numbers are currently hard coded in the source code. In a later stage of development, further flexibility in terms of custom symbolization might be implemented.

```

.marker-cluster-small {
    background-color: rgba(186,228,179, 0.8);
}
.marker-cluster-small div {
    background-color: rgba(186,228,179, 0.8);
    height: 30px;
    width: 30px;
}
.marker-cluster-medium {
    background-color: rgba(116,196,118, 0.8); /* greens*/
}
.marker-cluster-medium div {
    background-color: rgba(116,196,118, 0.8); /* greens*/
}
.marker-cluster-large {
    background-color: rgba(35,139,69, 0.8); /*greens */
}
.marker-cluster-large div {
    background-color: rgba(35,139,69, 0.8); /*greens */
}

```

Listing 2: Styling of the Cluster Symbolizations using CSS rules.

The type of representation may be changed during run time. It is possible to switch between point-based and polygon symbolizations as well as to alter the calculation method of the labels. Additionally, a method was implemented offering the possibility to draw the grid on the map. Again, the styling of that symbolization can be customised.

Listing 3 illustrates all available options to customize the algorithm. Two options have been implemented to control the limits of the clustering process. First, a threshold determining the maximum zoom level can be set, which will limit the cluster visualization to that given level. On any zoom level higher than this threshold the features will be drawn using their original representations. Second, the minimum amount of features, which are necessary to create a grid cell may be set. If the number of features in a given cell is lower than this threshold, the original feature representations will be drawn on the map. This option might be of use in situations, in which only few features lie within a given cell. In this case, no visual overlapping has to be expected and hence, the representation of single features using their original locations will most likely increase the accuracy of the map.

```

options : {
  gridSize : 1,
  zoomFactor : 2,
  minFeaturesToCluster: 1,
  colors : ['rgb(254,178,76)', 'rgb(253,141,60)', 'rgb(252,78,42)',
'rgb(227,26,28)', 'rgb(177,0,38)'],
  maxZoom : 16,
  showGrid : true,
  showCells : true,
  showCentroids : true,
  weightedCentroids: false,
  cellStyle : {
    color : 'gray',
    opacity : 0.1,
    fillOpacity : 0.5
  },
  gridStyle : {
    color : 'gray',
    weight : 1,
    opactiy : 0.8
  }
}

```

Listing 3: List of all available options to customize the clustering algorithm.

The prototype algorithm was designed to be as convenient as possible to use. In a minimal setup, only four lines of code are required to add and activate the grid-based algorithm on a web map (as illustrated in Listing 4). Still, most applications will require the application of custom options, which can be defined as simple JavaScript objects.

The design and implementation of the grid-based clustering prototype was accomplished within the course of two months. The final version of the prototype consists of roughly 500 lines of code. Considerable improvements in performance and code efficiency might be achieved with more effort, but for the scope of this research, the prototype is suitable to fulfil its task in the next chapters, as is shown in the following chapter.

```

var map = L.map('map', {});
var gridCluster = L.gridCluster().addTo(map);

var thematicLayer = L.geoJson(earthquakes, {});

gridCluster.addLayers(thematicLayer);

```

Listing 4: Minimal setup of the grid-based clustering algorithm.

7.3 Evaluation of the Performance

In this chapter, the computing performance of the prototype will be assessed. As mentioned in chapter 7.1, the algorithm has to compute fast enough to allow a smooth and responsive user experience in real life situations. Hence, it has to be tested, until which amount of features the prototype is able to deliver satisfying results.

Therefore, four test cases were built using the demonstration pages of the Leaflet.MarkerCluster plugin as a template (MARKERCLUSTER-2). By doing so, the computing performance of these two algorithms can be compared, and it can be evaluated, if the performance of the prototype is able to compete with the calculation speed of the existing clustering plugin, which is already used in productive environments.

Both the implementations were tested on three different, typical systems: A laptop, a tablet and a smartphone. The laptop was powered by an Intel Core i5 460M CPU (Central Processing Unit) with 2 x 2.53 GHZ, 6 GB of RAM and a SSD hard drive and was running on Windows 7. The tablet was powered by an Intel Atom Z3740 CPU with 4 x 1.33GHZ, 2 GB of RAM and a SSD hard drive with Windows 8.1 installed on it. The smartphone was powered by a MTK6577 2 x 1.0 GHZ SOC (System on a Chip), 1 GB RAM and was running Android 4.2.2.

The tests have been conducted on the latest version of the browser software Google Chrome. Several performance tests have been run, testing the rendering and scripting time on various scenarios and with varying amounts of point-shaped features. All tests have been performed locally, meaning that all relevant script files and vector data files have been accessed from the local hard drive, as loading times over the internet have not been the focus of this evaluation. Only the tiles of the base map have been accessed over the internet.

In total, *four scenarios* have been prepared and evaluated, as composed in Table 2. In the first scenarios, it was measured how much time was elapsed until the page was fully loaded the first time. Here it was assessed, how efficient the algorithms can process big chunks of raw input data. The scope of the third and fourth scenario was to evaluate the processing time of the algorithm when changing to a smaller zoom level. These two scenarios emulated the real life situation, when a user is altering the map zoom while browsing the map. When interacting with the map, the application should respond immediately, thus the faster the clusters are recalculated, the smoother the resulting user experience will be.

Table 2: Composition of scenarios for performance evaluations.

Scenario	Amount of Features	Assessed Event	Zoom Level
Scenario 1	10 000	Time of first page load	13
Scenario 2	50 000	Time of first page load	13
Scenario 3	10 000	Time elapsed on zoom level change	13 to 12
Scenario 4	50 000	Time elapsed on zoom level change	13 to 12

For each scenario, five passes were conducted, using the Leaflet.Markercluster plugin and four variations of the grid-based prototype, namely: (1) marker symbolizations, (2) cell symbolizations, (3) weighted marker symbolizations, and (4) weighted cell symbolizations. These four variations have been chosen to examine, if, and to which extent, the type of symbolization or the method of calculating the label positions have a perceptible influence on the performance of the algorithm.

Each of the tests was repeated three times, and the average of the results was calculated afterwards. In total, 180 test cases have been assessed (60 on each system).

In the following paragraphs, the key results will be explained. First it has to be noted, that the amount of time needed to load and render the page is negligibly short, compared to the time necessary to execute the program code.

Scenario 1:

In the first scenario, the MarkerCluster algorithm needed more time than the prototype to process the code. When processing 10 000 data points, it was computing up to 64 % slower (on the tablet, 50 % on the laptop, 46 % on the smartphone) than the fastest grid-based algorithm, as illustrated in Table 3.

Table 3: Time needed to compute 10 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 1).

Device	(1)	(2)	(3)	(4)
Tablet	-56%	-56%	-52%	-64%
Laptop	-49%	-45%	-45%	-50%
Smartphone	-42%	-36%	-45%	-46%

For the performance of the grid-based approach, the type of visualization and method of position calculation did not have a noteworthy impact on the computing times on any of the tested devices.

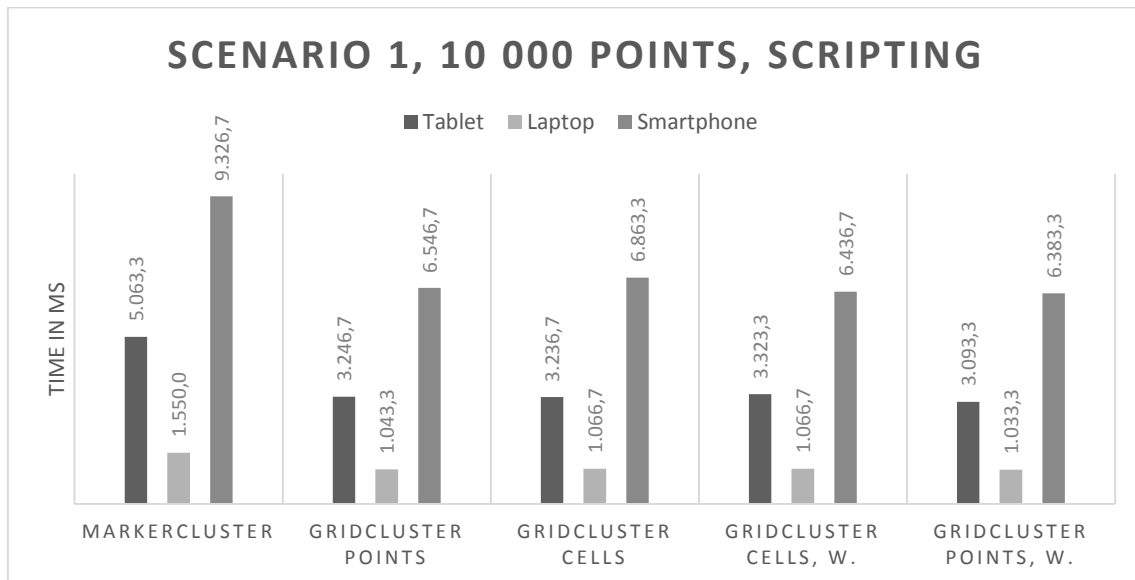


Figure 23: Average time needed for code execution in the first test scenario (10 000 features).

As illustrated in Figure 23, the absolute time needed to process the code varied heavily, depending on the used system. More than nine seconds waiting time for the MarkerCluster algorithm, and up to seven seconds for the grid-based method demand quite some patience from the user, when the page is loaded for the first time.

Scenario 2:

When processing 50 000 features, the MarkerCluster algorithm performed up to 19 % quicker than the fastest version of the grid-based algorithm, as shown in Table 4. The results for pass (4) on the tablet and on the smartphone fall out of scope compared to the other approaches. It remains unclear, why the algorithm performs so much better on these systems (see also Figure 24), as the calculation time for the cluster cells and for the geometry of the symbolizations remains the same on all four variations.

Table 4: Time needed to compute 50 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 2).

Device	(1)	(2)	(3)	(4)
Tablet	+7%	+7%	+7%	-3%
Laptop	-6%	+13%	+14%	+14%
Smartphone	+12%	+17%	+19%	-15%

However, even with three reruns per test, inaccuracies cannot be eliminated completely, and random occurring background tasks of the operating system might have an influence on the computing results.

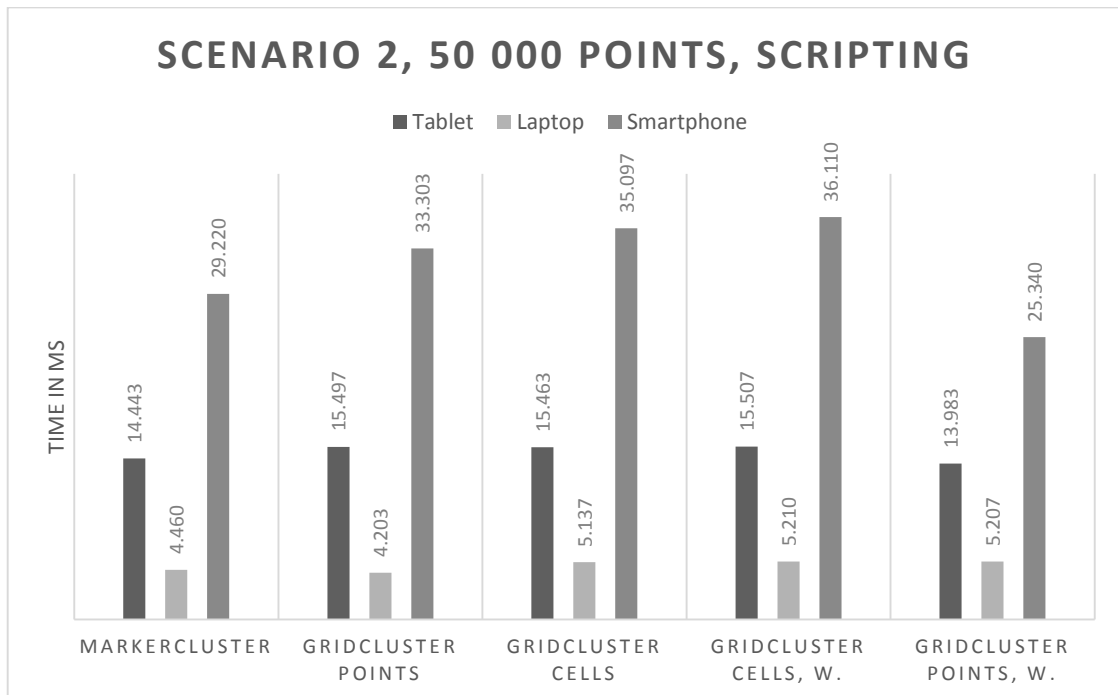


Figure 24: Average time needed for code execution in the second test scenario (50 000 features).

Regarding total numbers, the performance of both algorithms for processing 50 000 points when executed on an average tablet can be described to be not very pleasing, while the results on the smartphone are not acceptable for productive environments at all. Processing times of up to 36 seconds on initial page load are quite long, and demand patience from the user. Not

to mention, that in a real life application, the data points would have to be transferred through a slow internet connection (compared to local file access), increasing the waiting time even more.

However, when embedded into a bigger application, a smart design of the website could help to conceal the long loading times, through presenting additional content that could keep the user busy until the map application would be ready to use.

Another interesting discovery was the behaviour of the grid-based algorithm on the tablet. Comparing the computing times for pass (2) and pass (4), the calculation of the average location of the contained features was performed faster than the calculation of the geometric centre of the cluster cell.

Scenario 3:

In the third scenario, the responsiveness of the map was tested when triggering a zoom change. The setup of the first scenario was used for this simulation, meaning that 10 000 features have been loaded into the application. In contrast to the first two scenarios, the values of the time measures in this scenario lead to a different picture. Here, the MarkerCluster computed fastest, meaning that the map interaction is smoothest using this algorithm. While the grid-based algorithms are almost on a par with the MarkerCluster algorithm on the laptop, the grid-based approach performs notably slower on the mobile devices, needing up to 68 % more time to compute on the smartphone, as shown in Table 5 below.

Table 5: Time needed to compute 50 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 3).

Device	(1)	(2)	(3)	(4)
Tablet	26%	44%	47%	28%
Laptop	-5%	26%	29%	1%
Smartphone	68%	61%	58%	38%

Speaking in absolute processing times, the subjective user experience can be described to be smooth with all tested clustering approaches, except for the smartphone, where small lags of about one second occur when using the prototype. On the laptop and the tablet, the recalculation of the clusters is performed well under half a second, as illustrated in Figure 25. A noteworthy difference in calculation time could be observed between the cluster point visualizations and the cluster cell visualizations: Pass (4) was computing on average 29 % faster, and pass (3) needed an average of 10 % less time to process.

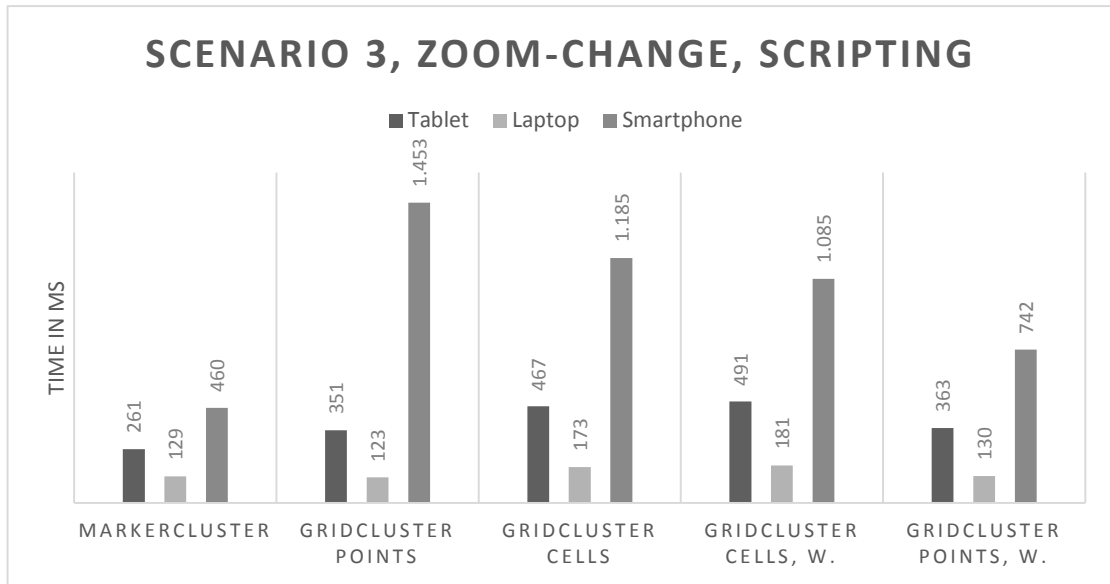


Figure 25: Average time needed for code execution in the third test scenario (10 000 features).

Scenario 4:

In the fourth scenario, the MarkerCluster algorithm recalculated the clusters almost instantly, as well. The prototype needed up to 89 % more time for the fourth scenario (on the smartphone), as shown in Table 6. Although those differences appear to be huge, all algorithms recalculated the clusters well under the time of one second, except for the smartphone. As illustrated in Figure 26, up to two seconds were needed in that case. Especially on the laptop, estimated waiting times for the user are not dramatic at all.

Surprisingly, the overall calculation times for the scenario using 50 000 features were about 50 % shorter than the ones achieved in the third scenario, although five times more features have been used. The explanation might be found in the templates used for the scenarios: the dataset including 10 000 features differs from the one using 50 000 features in the covered map extent. As a consequence, the ratios of visible parts of the datasets are not exactly the same.

Table 6: Time needed to compute 50 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 4).

Device	(1)	(2)	(3)	(4)
Tablet	77%	80%	81%	80%
Laptop	74%	76%	76%	74%
Smartphone	79%	89%	80%	85%

It is evident that the prototype struggles on the tablet – and even more on the smartphone – when processing the calculation of 50 000 features. Further code improvements could solve this problem.

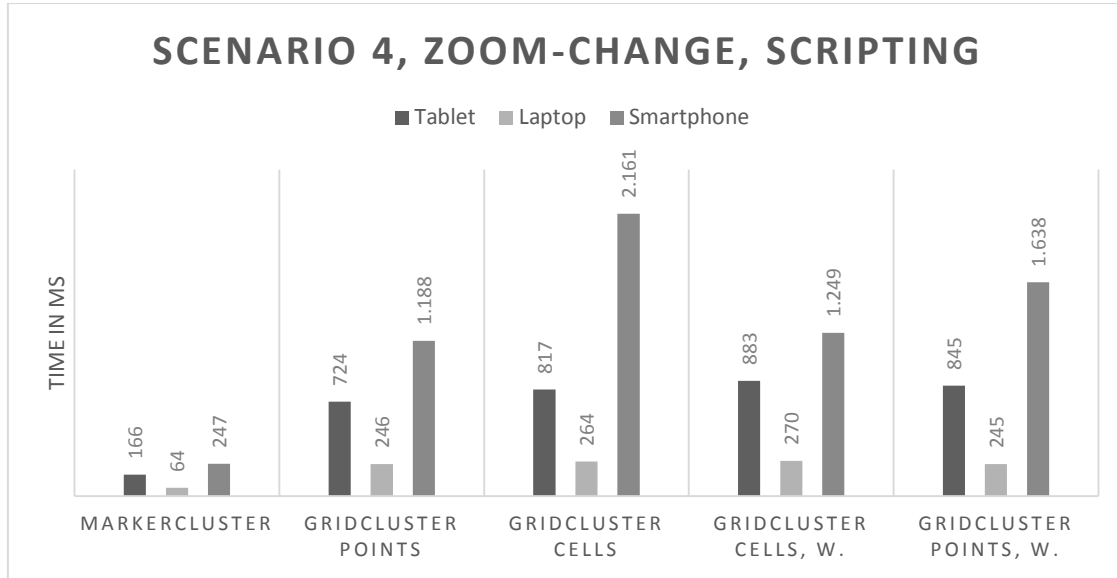


Figure 26: Average time needed for code execution in the fourth test scenario (50 000 features).

Conclusion:

To conclude this chapter, the evaluation of the performance has shown that a satisfying, reasonable computing speed can be achieved with datasets containing up to 50 000 points, depending on the performance of the used device. Only when exceeding this number, the prototype needs too much time to process the input data. As a consequence, the map behaviour becomes sluggish, leading to an unsatisfying user experience. It became clear that the limits on mobile devices are reached much earlier than on desktop computers, as the processing power of mobile processors is still limited. In any case, the time needed to transfer the data over the internet has to be added to the measures results. Depending on the connection speed, and the design of the application, this can be an additional limiting factor.

It was shown that the prototype has a small advantage concerning the processing times on initial page loadings, while the MarkerCluster algorithm can score with almost instant calculation times on zoom changes. The type of visualization and the method used to determine the position of the labels does not have a perceptible effect on the processing speed of the grid-based prototype.

For the scope of this research, the processing speed of the prototype seems to be reasonably fast. Furthermore, on typical desktop computers the algorithm might already be used in productive environments.

8 SURVEY

With the implementation of the grid-based algorithm and the definition of suitable use cases ready, a small online survey was set up to test the working hypothesis. As a short reminder, in chapter 1.2 the working hypothesis was defined as follows:

*Grid-based cluster approaches can be used in web mapping to **improve map readability** on web maps with a high, point-based information density, compared to algorithms used in popular mapping libraries.*

In the following chapters, the methods which were used in the questionnaire will be described, as well as the design of the survey, the target audience and last but not least, the answers of the participants will be presented and analysed.

8.1 Goal of the Survey

The online survey aims to clarify, which experiences users make while using a web map which features a real time clustering algorithm. It further shall be examined under which conditions a grid-based clustering strategy can be benefiting, and in which situations users would prefer a map with a density-based clustering technique. In the end, a cartographer wants to use the most efficient method to communicate data to the user, so the participants were asked to rate their experiences and state, which visualization they preferred in a given situation. The evaluation of that feedback will help to answer the research questions stated in chapter 1.2.

8.2 Survey Design

In this section, the structure and the implementation of the survey will be described. To examine the feasibility of a grid-based clustering strategy during runtime, the preceded use cases (chapter 6) were prepared and shown to the participants. Both use cases were presented using the distance-based and the new developed grid-based clustering methods. Following, the participants were asked to rate the visualizations under certain aspects, which will be described below.

The use cases have been programmed using Leaflet.js and the corresponding clustering plugins and were performed locally, while being recorded using a screen recording software. Thus, the participants were presented short video sequences (each about 30 seconds of length) showing an exactly defined map behaviour. In choosing to show pre-recorded video clips, a series of interferences could be eliminated: These might be varying performance of the hardware used by the participants, a slow internet connection or browser incompatibilities. In consequence, the examples were exactly the same to all participants (cf. SARODNICK & BRAU 2011, 164), only a slow internet connection could slow down the streaming of the video clips.

Both use cases were shown in the three variations described in chapter 6: first, the distance-based clustering algorithm was presented, followed by the same scenario using grid-based cluster marker symbolizations. Finally, the scenario was repeated using colour-coded cells (laminar symbolizations) of the grid structure. The decision to show the distance-based ap-

proach first was made, as this visualization method is already available and therefore, probably well-known by the participant. In doing so, it was attempted to reduce the bias of the participant when showing the same scenario three times.

In this survey, the method of deductive testing was used, as described in chapter 4.2. By showing three variations of the same use case, the participants were able to compare the different approaches and thus, to form an opinion about what they liked and disliked in each variant. After each series of visualizations, the participants were asked, which visualization technique they liked most, and to rank the visualizations. In addition, an open question offered the opportunity to write the impressions the participants had during the presentation of the scenarios.

The survey was concluded with a series of questions determining the internet experience, possession of electronic devices, and demographic facts of the participants. These information were used to classify the participants into different levels of experience.

The survey started on 15.01.2015 and was active for 48 days, until 24.02.2015. During that time, it was promoted on social networks, forums and user groups. In addition, friends, colleagues and family members were also asked to participate. The survey was designed using the open source software package ‘limesurvey’ (LIMESURVEY-1), which was installed on the personal web space of the author. The questionnaire was offered in an English and German version, each consisting of a total of 18 questions. The questions have been grouped thematically: for both use cases, each visualization method was presented as one group. Every use case was concluded with a request to rank the visualization methods. The English version of the questionnaire is printed in the appendix at the end of this thesis.

According to MENG et al. (2008), insights gained through questionnaires and scenarios may vary in the level of candour, so as a consequence, the answers given in the open question may only be evaluated using the previous knowledge of the survey creator.

The *target audience* of this survey is not easy to define, as every internet user is a potential participant. Thus, only a random sample can be taken into account. Therefore, the survey cannot be seen as statistically significant, as the number of participants was too low. However, a subjective profile of different opinions could be collected, and as stated above, issues in the design of the visualizations could be determined. These results already can help to indicate design issues and narrow down possible applications of a grid-based clustering approach.

8.2.1 Use Case 1

The first group of questions was created to present the first use case to the participants. In this use case, the performance of the density- and grid-based clustering algorithms in a map mashup scenario was assessed. First, the scenario was explained to the participant, and the symbolizations used in the map as well as the task the user has to fulfil, were laid out. The first version of the use case was presented to the participant using the density-based method. Following up, he was asked to rate several aspects of the presentation. These questions were implemented using arrays, ranging from ‘no – disagree’ to ‘yes – agree’ in five steps. The questions can be found in the appendix.

The same set of questions was reused to present and assess the first use case using alternative clustering methods, with the grid-based approach using marker symbolizations being the second version, and finally, with the grid-based approach using laminar symbolizations.

This first question group was concluded with the task to create a ranking of the three visualization techniques. The participant was asked to rank them in ascending order. In an open question, the participant could write his impressions, and state what he liked or disliked about the particular visualizations.

8.2.2 Use Case 2

Similar to the first use case, one goal of the examination of the second use case was to determine, which visualization technique the participants preferred. Furthermore, the questions were designed with the intention to understand, with which method the participants could identify the patterns of temporal change in the most accurate way. After the presentation of the video sequence showing the animation with the respective clustering method, the participants were asked to indicate the areas in which they believed to have seen the strongest developments. Therefore, the map extent of the example was divided into equal sized, consecutively numbered cells. In addition, they were asked to rate, how clear it was for them to understand which region on the map was represented by each symbolization.

Again, the second use case was concluded with the request to create a ranking of the three visualization techniques. An open question enabled the participants to state their impressions and comments.

8.3 Survey Results

In the 48 days the survey was open, a total of 111 users followed the link to the survey and opened the questionnaire. Most of the users closed the survey website without answering any questions. Unfortunately, 71 users did not finish the questionnaire, from which eleven aborted the poll at some point in the middle. Consequently, a total of 40 participants completed the survey, resulting in a success rate of 36 %. The incomplete responses have not been included in the evaluation of the survey.

The average time needed to complete the survey was about 49 minutes, resulting from a participant who left the survey open for over 22 hours. The more accurate median time was about fourteen minutes. In the following sections, the interview audience will be explored, and the key results of the first and the second use cases will be presented and analysed.

8.3.1 Survey Audience

In order to interpret the results of this survey as correctly as possible, it is benefiting to begin the evaluation of the survey with an overview over the interviewed audience. The details about the experiences and the demographic structure of the participants will be revealed in the following paragraphs.

The evaluation of the personal questions showed that most participants have had an above mediocre level of internet experience. Most users stated to frequently use online services like internet banking (82 %), online shopping (56 %) or social networks (56 %), as illustrated in Figure 27. The absolute majority of the probands said that they frequently use online mapping services (90 %) and that they consult the internet for research purposes (97 %). It can therefore be anticipated that the surveyed audience was familiar with map mashups and cluster representations of POIs.

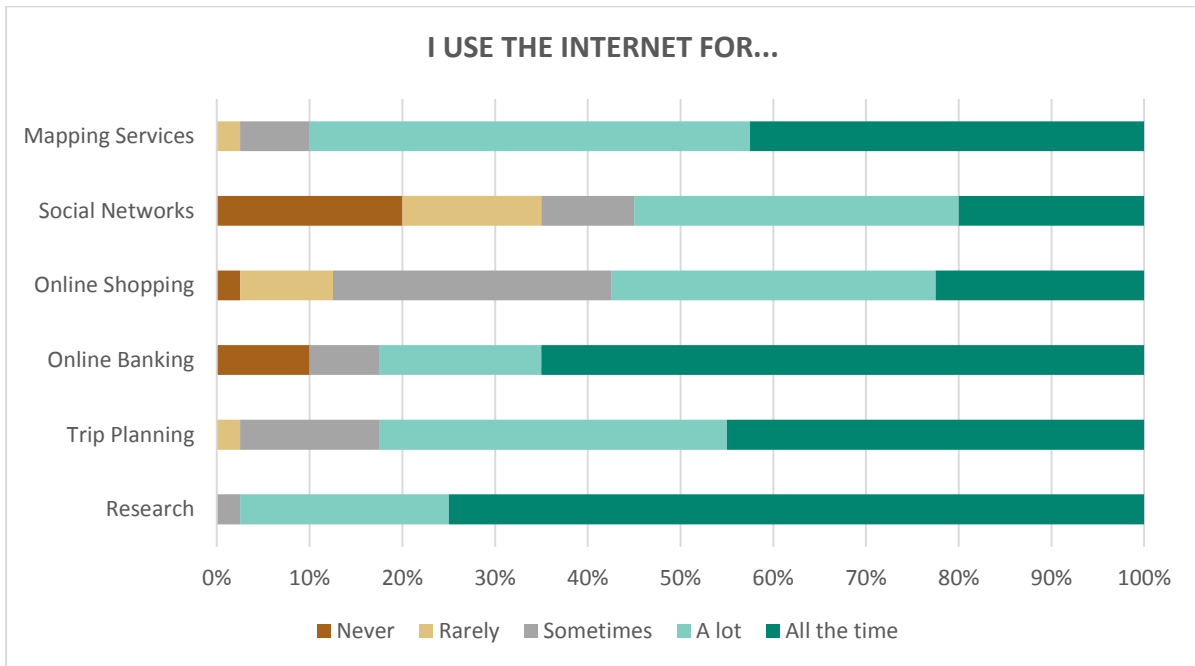


Figure 27: Internet experience of the interviewed audience (N=40).

When asked, which devices they call their own, all participants stated to own a computer or laptop, and 90 % claimed to possess a smartphone, while only one fourth of all participants stated to own a tablet device. About half of the surveyed audience owned a proprietary navigation system, or used navigation apps on their mobile devices (compare Figure 28). These results seem to be interesting, as the *mobile use of mapping applications* even increases the icon cluttering problem (as shown in chapter 2.3.2), and therefore emphasises the need for efficient decluttering solutions and intelligent data visualization techniques.

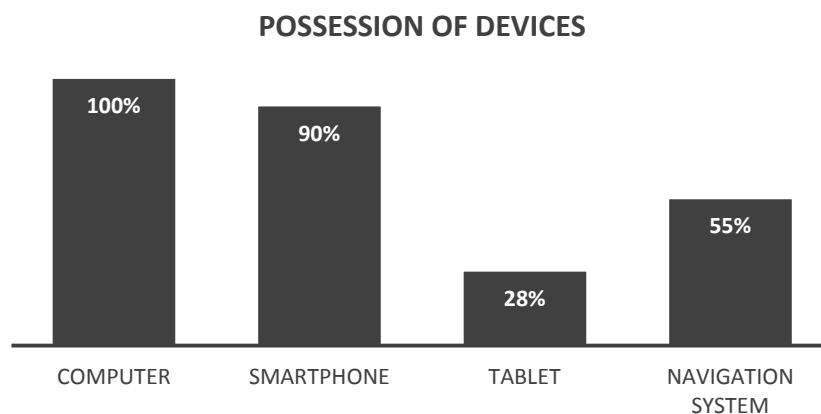


Figure 28: Possession of digital devices. (N=40)

The average age of the survey participant was 34.3 years. The youngest user claimed to be 20 years old, while the oldest participant was 81. Both genders have been represented in an almost equal share, with 43 % of the users being female, and 57 % stating their gender to be male. To conclude, most participants (61 %) used the German version of the survey.

8.3.2 Use Case 1

In this section, the results of the participants concerning the first use case will be evaluated. As a reminder, the scenario was repeated using first the distance-based, then the grid-based visualization (both using marker symbolizations) and lastly, the grid-based approach using colour-coded cells as cluster representations.

After presenting the short video sequence, the participants were asked, whether it was easy for them to figure out the bike stations close to their location, or not. The answers are illustrated in Figure 29. According to the results, the clustering method using grid-based marker symbolizations was perceived to be the technique which delivered the clearest visualization. This method received 91 % approving votes, followed by the distance-based (MarkerCluster) visualization with 76 % of the participants agreeing with the asked statement.

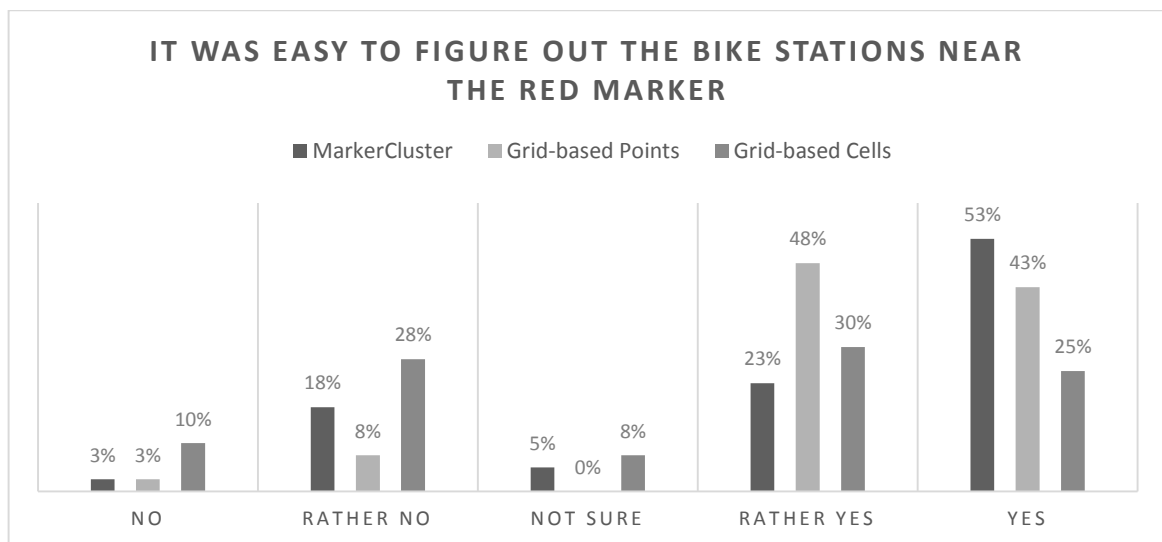


Figure 29: Results of the question, how easy the location of the bike stations could be determined (N=40).

With only 55 % positive votes, the method using colour-coded cell symbolizations seemed to be unattractive to the participants, which was also reflected in the results of the next question (see Figure 30).

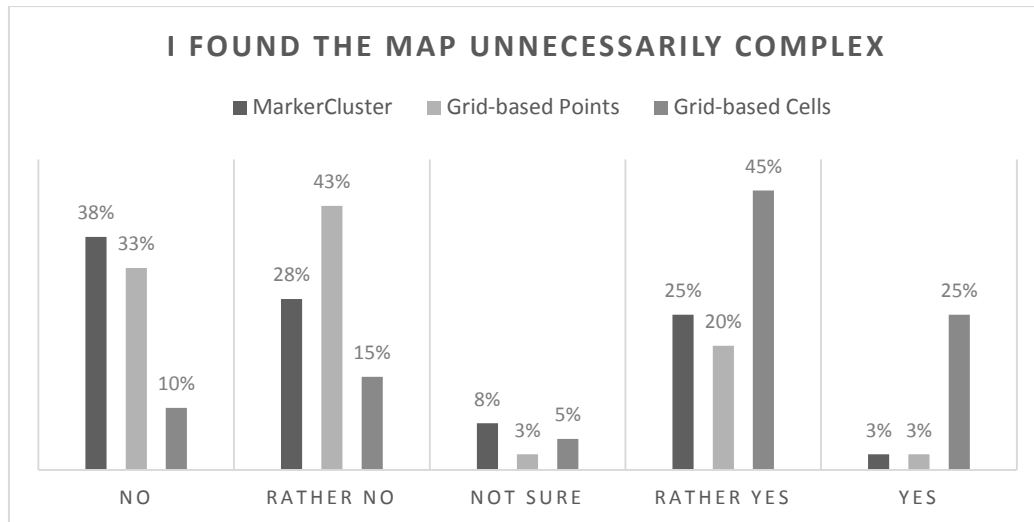


Figure 30: Results of the ratings for unnecessary map complexity (N=40).

When asked, if the representation made the map appear more complex than necessary, the version using colour-coded cells produced unfavourable results, as over 70 % of the users found the map to be unnecessarily complex. In contrast to that, the grid-based method and the distance-based method have been voted to be similarly eligible to keep map complexity on an acceptable level, with the grid-based method having a slight lead.

Recalling the definition of use cases and user behaviour given in chapter 6.1, the next question asked if users would use cluster symbolizations as orientation points (landmarks). The evaluation of the survey revealed that over 60 % of the participants actually used the cluster symbols as landmarks on the map in the first two visualizations (63 % for the first visualization, 66 % for the second). As it can be seen in Figure 31, less participants (only 33 %) stated to use it in the third case, when colour-coded cells have been used as cluster representations.

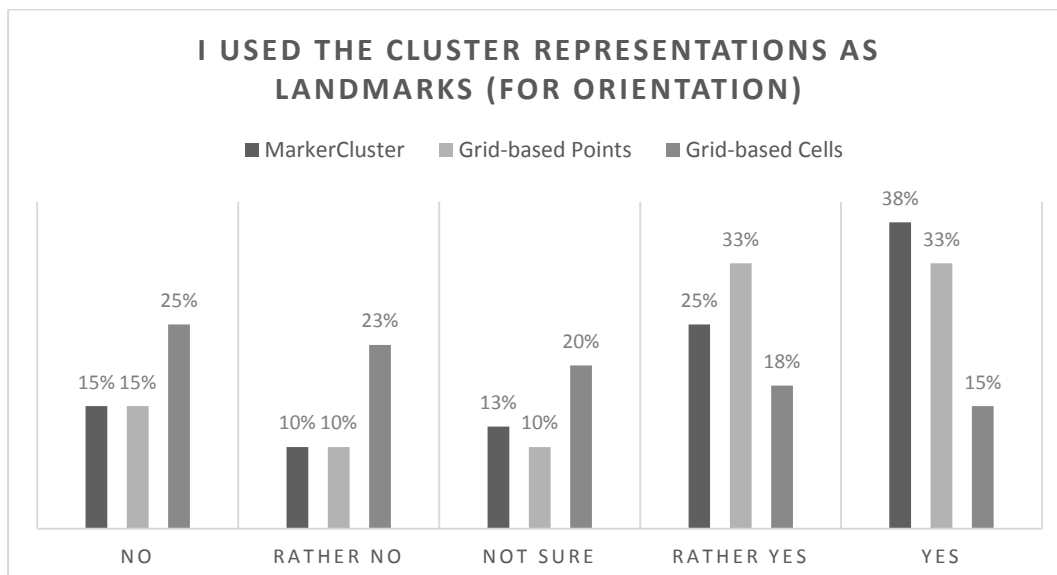


Figure 31: Usage of cluster representations as landmarks (N=40).

Related to that, the participants were asked, if they found the change of marker's locations to be irritating, while the map zoom was altered. Figure 32 illustrates the results for the distance-based and the grid-based approaches. Although most users did not perceive the location

change of the cluster representations to be irritating, the grid-based approach appeared to be less confusing with 71 % of the votes saying ‘(rather) no’, compared to the 53 % for the MarkerCluster visualization.

This question was not asked on the third try, as the location change of the laminar symbolizations is less dramatic, due to the broader visual extent of the representations.

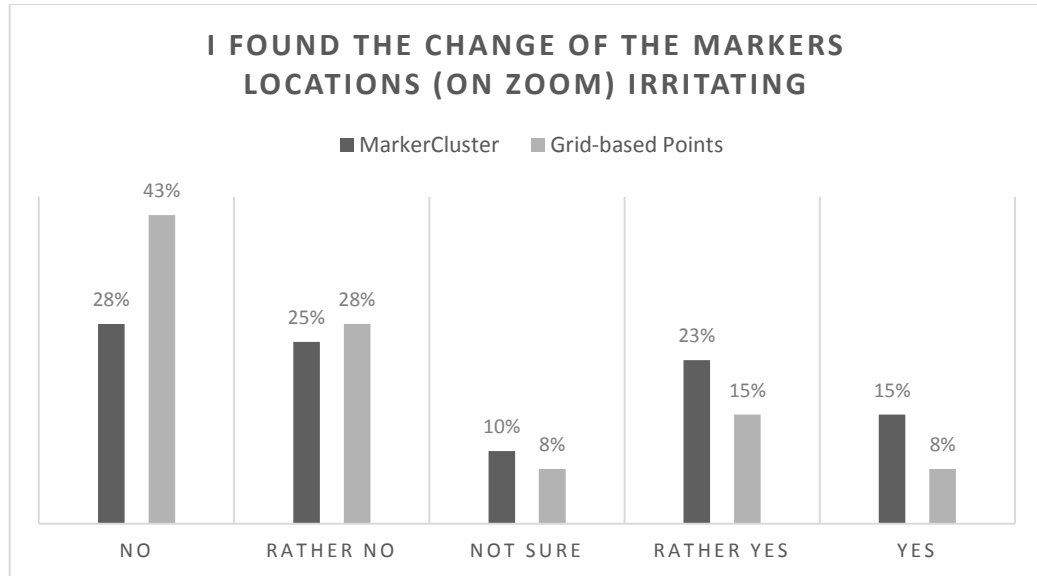


Figure 32: Perception of marker behaviour on zoom change (N=40).

Instead, the participants were asked in the third run of the use case, if that type of visualization was intuitive to them. As shown in Figure 33, about 46 % stated, that it was somehow intuitive for them, while a slight majority voted the approach to be not intuitive.

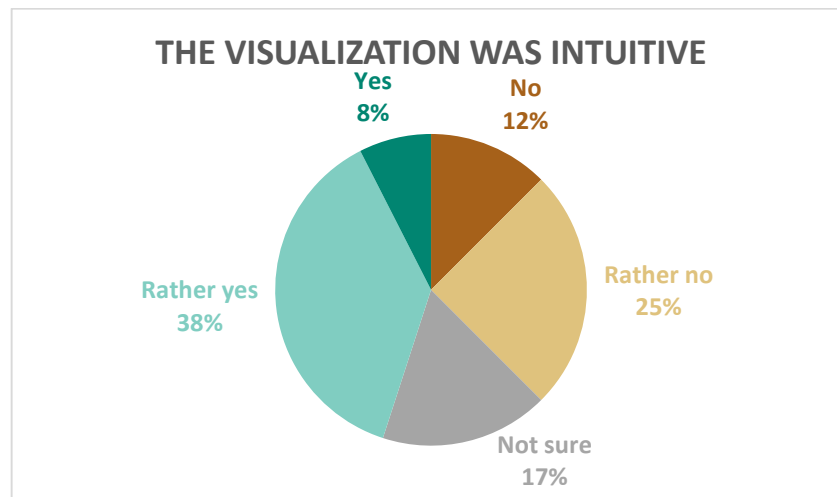


Figure 33: How intuitive was the grid-based clustering algorithm using colour-coded cells as representations to the participant? (N=40).

The last sub question asked the participants to rate, to which extent the visualization technique was suitable to help them fulfil the task of locating bike stations near their position. The approach to visualize the bike stations by using grid-based cells was accepted least, while the first two approaches have been voted to be almost on par, as illustrated in Figure 34. Again,

the grid-based approach using marker symbolizations received the highest amount of concurring votes (83 % versus 75% for the distance-based approach).

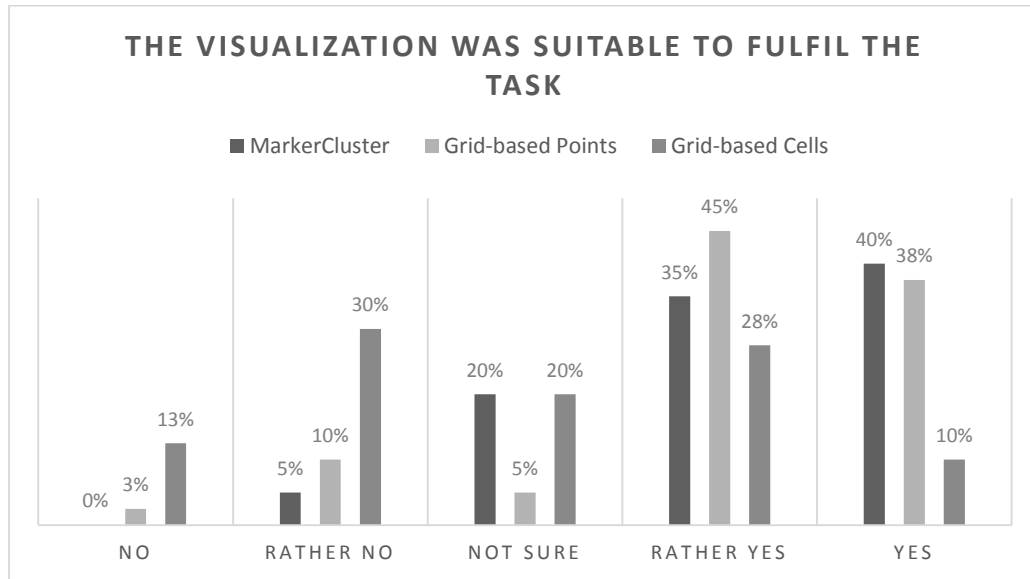


Figure 34: Eligibility to fulfil the task of locating bike stations (N=40).

When questioned, which visualization technique they liked most, the participants voted the distance-based approach (the *Leaflet.Markercluster*) on first place, followed by the grid-based approach using marker symbolizations. Consequently, the third approach was the least popular, which goes hand in hand with the results of the preceding questions. The results are illustrated in Figure 35.

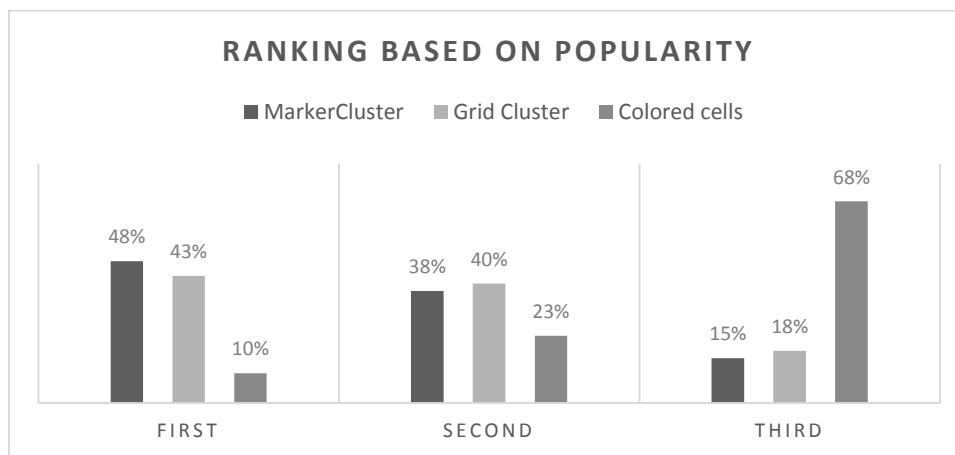


Figure 35: Ranking of the visualizations based on popularity (N=40).

Related to the questions about using the cluster representations as landmarks, the participants were asked if they used the grid lines of the grid-based visualization to orientate themselves on the map. As displayed in Figure 36, almost 60 % answered with yes. This result leads to an interesting insight, as grid lines could be added to any map mashup or mapping application, to aid the users to find orientation, regardless of the type of clustering algorithm.

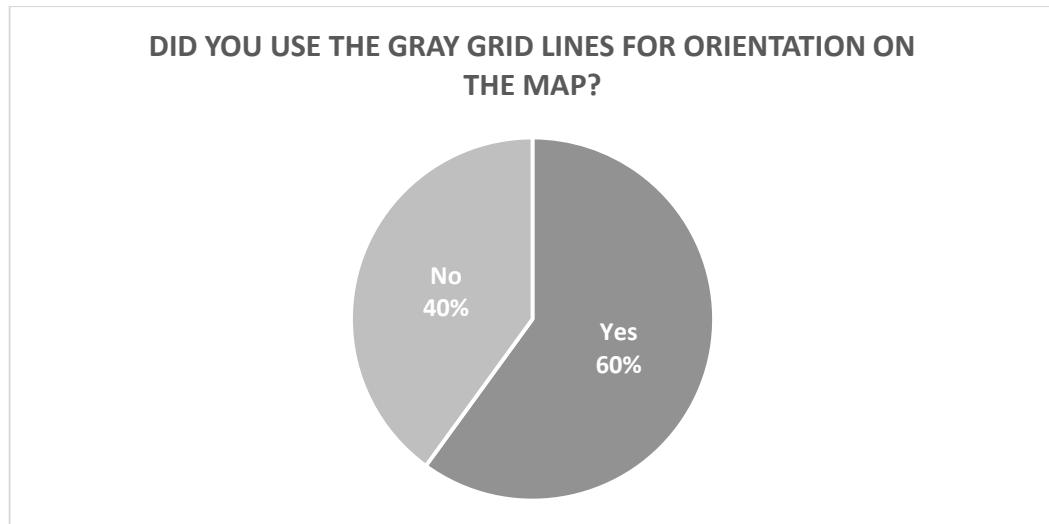


Figure 36: Results indicate that grid lines may serve as orientation support on web maps ($N=40$).

The evaluation of the open question, in which the participants could provide their feedback and thoughts about their ranking revealed several details, which will be discussed in the following paragraphs.

One user stated that he was *colour blind* – this is an interesting comment as colour blindness is often unthought-of when designing a map. Especially when using colour-coded symbolizations to classify data, this fact should be kept in mind when choosing a set of colours.

However, there have been *contradicting* opinions. Some users criticized that the grid-based algorithm would split the map into random pieces, rather than dividing it into ‘*public known areas*’. While this point certainly is valid, it might as well be true for the distance-based approach. As already mentioned earlier, no available aggregation technique is intelligent enough to take the underlying base map into account.

Many participants emphasized the fact, that the distance-based approach gave them *no indication* about the size and shape of the area covered by each cluster symbol. In contrast, they liked the *clearness* of the equal sized grid cells (with marker symbols for the clusters), as it gave them an idea, which area was taken into account for each symbol.

Opposing the preceding results of this chapter, some users wrote in their comments that the distance-based algorithm left them in false believe about the real allocation of the stations, and that they were irritated by the jumping of the cluster locations. In contrast, they emphasized the structure provided by the grid-based clustering approach. Other participants even liked the visualization using the cells. Especially the classification of the amount of feature using colour-coding was mentioned several times to be useful.

In summary, the evaluation of all answers to this open question indicated a trend that confirms the results of the previously presented questions: Most users favour the distance-based approach, followed by the grid-based method using marker representations. The majority of users dismissed the third approach for the first use case, it seems to be unsuitable for this task.

8.3.3 Use Case 2

In the second use case, the evaluation of the answers lead to different picture than in the preceding use case. The answers and results will be presented in the following paragraphs.

After the presentation of the video sequence, the participants were asked to select those areas, where they believed to have seen the strongest developments throughout the animation. Counting clockwise from the top left, the results derived from the Markercluster approach, grid-based approach using marker symbolizations, and the grid-based approach using cells are illustrated below in Figure 37. All three visualization approaches lead to similar results: The majority of participants identified the strongest developments in west- and central-Europe. Regions with less changes were identified better with the grid-based clustering solutions.

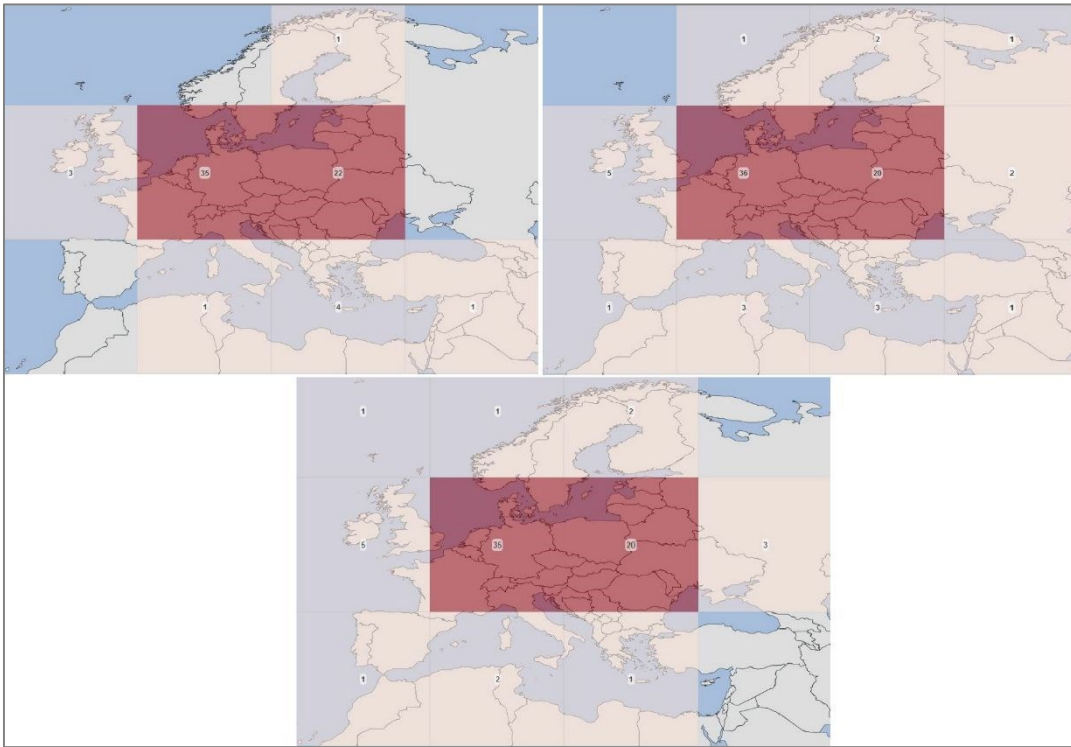


Figure 37: Regions with the strongest developments, according to the participants.

However, these results have to be interpreted with caution: It has to be considered, that the participants were getting familiar with the shown scenario after the second and third approach, and thus have been able to recognize more regions in the later repetitions of the use case.

The next question asked, how easy it was for the participant to understand the allocation of incidents shown on the map. The strongest results can be credited to the grid-based visualization using the cells as symbolizations: for more than 90 % of the surveyed users, this method was eligible to understand the feature allocation, while only 41 % found the distance-based approach helpful. The grid-based approach using marker symbolizations was voted similar helpful as the one using the cells, as 88 % of the participants agreed to the question statement, as shown in Figure 38.

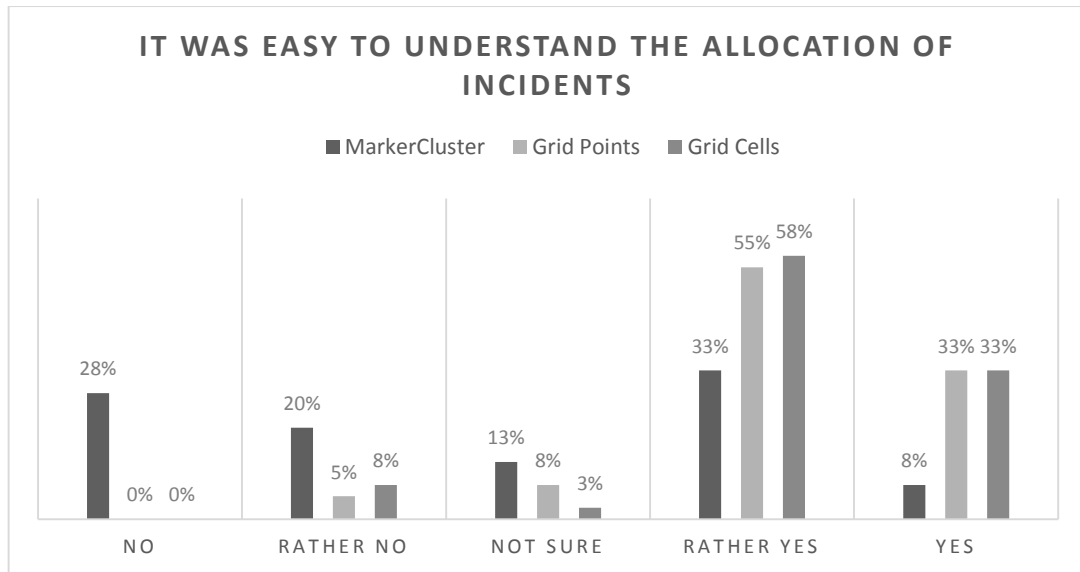


Figure 38: Understanding of Incident-Allocations (N=40).

The answers given to the question, if the participant found it easy to understand the shown map, draw a similar result (compare Figure 39): the third visualization was voted in total to be understandable in the easiest way, with 86 % of the participants voting for (rather) yes. The method using grid-based cluster markers received 86 % positive votes as well, with 58 % of the participants selecting ‘rather yes’. The MarkerCluster method was declined as only 51 % agreed to the statement, and 38 % found the map to be cumbersome to understand.

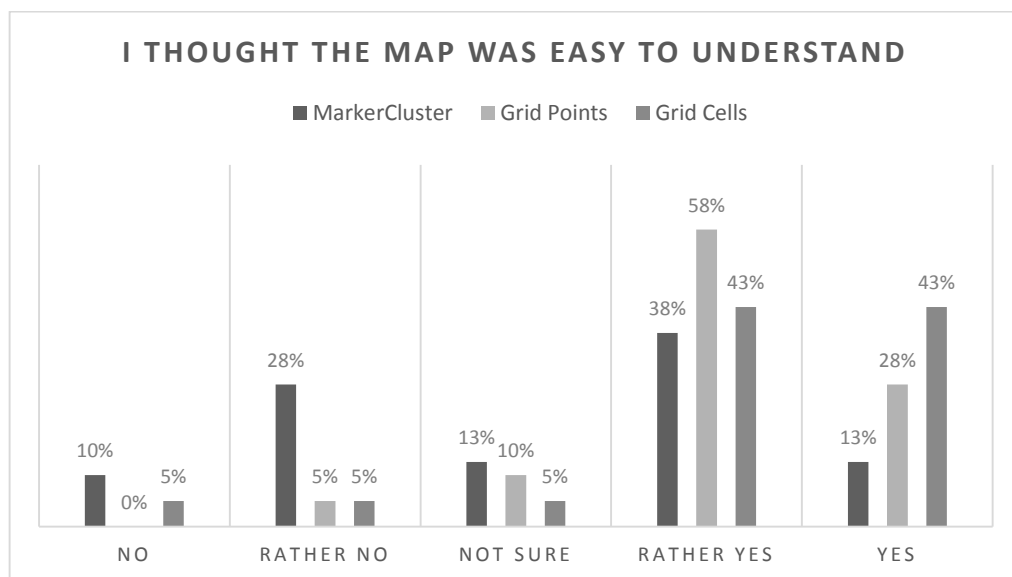


Figure 39: Understanding of the Map (N=40).

The performance of the distance-based approach seemed to be poor, as acceptance by the participants was low. This picture was continued in the next question. The users were asked, if the visualization made it clear, which region each cluster symbolization represented. In Figure 40, the results are illustrated. Again, the grid-based methods have been accepted by most participants, with 88 % voting for the grid-based markers and 85 % favouring the grid-based cells. The distance-based method only received 40 % positive votes.

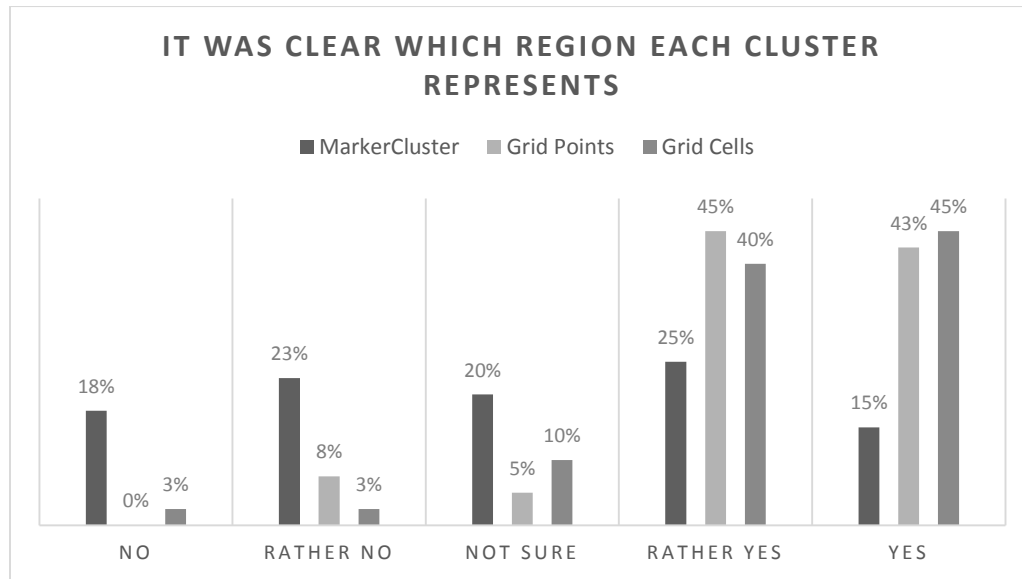


Figure 40: Representation of regions by cluster symbolizations (N=40).

Consequently, this method was elected to be unsuitable to communicate the data, with only 33 % of the users voting for the MarkerCluster visualization. The strongest support received the second visualization method with 80 % of the votes being positive. The method using grid-based cell symbolizations was perceived to be similarly suitable to communicate the thematic data, and received 68 % positive votes (see Figure 41).

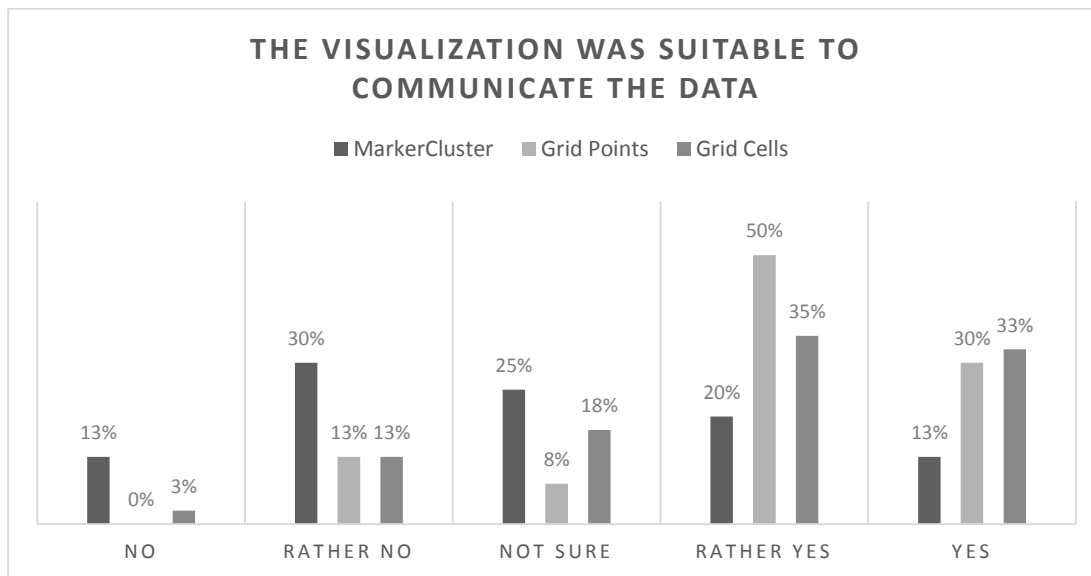


Figure 41: The visualization was suitable to communicate the data (N=40).

As a consequence, the visualization technique using grid-based cells for symbolization was the most popular one for the second use case, with 53 % of the users voting it on the first place. The grid-based approach using marker symbolizations was voted on second place, while the Markercluster approach was clearly dismissed on the third rank (compare Figure 42).

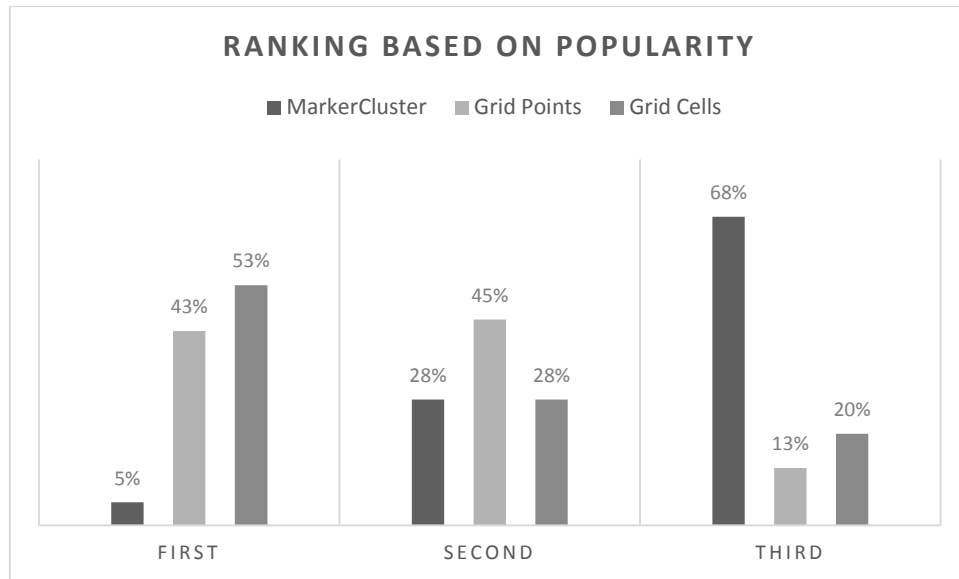


Figure 42: Ranking based on popularity for the second use case (N=40).

The evaluation of the open question lead to some interesting insights, which will be described in the following paragraphs. In total, 34 participants stated a comment; only six users did not fill out the form.

Most participants stated that the distance-based method was least understandable for them, they were irritated by the *jumping* of the cluster points and were left unclear, where the concentrations of incidents could have been. According to most answers, the distance-based approach leads to an unsteady map picture with too much change. As a consequence, the temporal change of the incidents could not be perceived anymore. In this case, the thematic information is not communicated in an efficient manner.

Hence, most users wrote that the visualization using the cells was most convenient and clear for them. Some of the users criticized that the laminar symbolizations added too much complexity to the map. In contrast, others emphasized how easy the colour-coding of the cells made the interpretation of the data for them, as they did not have to read the labels of each cell. Obviously, there exist controversial opinions about the usability of this kind of visualization.

However, most participants stated that the grid-based approaches was perceived superior to the MarkerCluster, as the understanding of the temporal changes was made easier. Many users wrote that they used the *static grid structure* as landmarks for orientation, thus allowing them to focus solely on the changes of the thematic data. The map was filled with less visual overload and so this visualization enabled them to interpret the developments in a more convenient way.

8.4 Limitations

After evaluating and analysing the results of the online survey, two major conclusions could be drawn, which will be laid out in the following section:

However, the methods chosen for this survey may have influenced the answers of the participants. For instance, it could be possible that the participants have been influenced by the previous visualization techniques, meaning that when watching the same scene for the second or third time, they have been already biased by the preceding visualizations. It is arguable, if the presentation of different map locations would have led to different results. On the other side, presenting exactly the same situation made the approaches strictly comparable.

The limited time which was available throughout this thesis hindered the interrogation of a larger amount of users, which could have led to a different outcome. Nevertheless, the majority of participants voted in a similar way, and the statements given in the open questions mentioned the key issues. It can be expected, that a larger amount of participants would not have affected the overall picture in a drastic way.

9 KEY FINDINGS AND RECOMMENDATIONS

In this chapter, the key findings of the presented research will be discussed. The evaluation of related work, the survey, and the examination of the clustering algorithms indicated several benefits for the usage of point-based clustering algorithms. However, also restrictions and pitfalls became evident. In the following sub-chapters, these chances and issues will be addressed.

9.1 Chances and Advantages

To get back to the research objectives of this thesis, one of the research questions from chapter 1.2 asked: ‘*Where can the application of a grid-based clustering strategy be advantageous?*’

After the evaluation of the online survey, the results and the answers given by the participants indicate that most users preferred the grid-based approach for thematic mapping purposes, while the application in map mashups seemed to be perceived controversially.

Especially the grid-based approach using laminar symbolizations performed worse. Only few probands liked this method of clustering, most users criticized the increased graphical complexity which is caused by this kind of symbolization. On the other hand, using areas as cluster representations brings the advantage of giving instant information about the coverage of each cluster to the map reader.

Depending on the targeted audience and the topic, the application of a grid-based clustering strategy may be of benefit in a number of situations: The survey clearly indicated that for *thematic mapping purposes*, especially for the animation of temporal developments, the grid-based approaches have performed very well. According to the answers of the participants, this

approach is very eligible to communicate thematic data in an efficient way, making it easy to understand the distribution of data on the map.

The regular nature of the grid was eligible to help users to orient themselves on the map. Furthermore, the rule-based increasing or decreasing of the grid cell size lead to a steady, subtle change of the locations of the cluster symbolizations. This method was perceived to be less disturbing – probably due to the predictable behaviour of the symbolizations.

As a consequence, the grid-based clustering algorithm might be of advantage in complex, dynamic mapping situations, as it does not complicate the visualization any further when using marker symbolizations. For thematic mapping purposes, the grid-based approach is of use in all situations which require the comparability of data originating from different regions or time stamps. It further has been proved to be feasible for the animation of temporal changes.

The possibility to colour-code the data clusters when using laminar cluster symbolizations proved to be another advantage of the grid-based approach, as it can facilitate the perception of the situation: The user is not required to read the labels of each cluster, as the colours of the cell give a clear visual indication about the data distribution.

With the availability of a client sided implementation, this approach might be used easily by map makers and cartographers, and can serve as an alternative to the existing distance-based algorithms. As map makers do not have to implement custom, server-side solutions, it might help to overcome the inhibition to use a grid-based approach in web maps.

9.2 Restrictions for the Usage

The evaluation of the performance tests, state of the art web mapping practices, and the online survey revealed several limitations to the usage of a grid-based clustering method. They can be divided into technical and thematic restrictions.

From a cartographic point of view, a major limitation can be seen in the map projection: At least for the application of a grid-based clustering method in typical thematic web mapping, the predominant Pseudo-Mercator projection is unsuitable to display grid cells on small map scales, as this is not an equal-area map projection.

Therefore, with growing distance from the equator, regions get more and more distorted by scale, as shown in Figure 8. This forms a serious issue for the application of a grid-based cluster algorithm, as the resulting grid cells will hardly ever be of a square form, if their geometry is not altered programmatically by calculating a scale factor. This kind of visualization can thus result in misleading interpretations by the user, as the cell areas are not strictly comparable with each other in a visual way.

The feasibility of a grid-based clustering algorithm for map mashups featuring POIs has to be seen critically, as the accuracy of the clustering result heavily depends on the chosen cell size. If the cell areas are too large, the resulting clusters will be too imprecise, leaving the user alone to guess, how the actual allocation of POIs on the map looks like.

On the other hand, a cell size chosen too small will lead to a fragmented map picture, and thus defeating the purpose of a clustering strategy. Conceivable solutions might be to either modify the scaling factor for the cell size in a progressive way, depending on the map scale, or to

calculate the location of the label (if using the cell geometry as symbolization) or the icon according to the distribution of features. However, the latter solution again leads to a less steady map picture, as the labels of the cluster cells are shifted in an unpredictable way after every change of the zoom level.

The evaluation of the performance has revealed another limitation for the application of real time clustering solutions in general: If the amount of (POI) data that has to be transferred to the client is too high, the resulting map behaviour will become sluggish and unresponsive. The critical amount of data heavily depends on the computing performance of the user device and the available bandwidth of the internet connection. Unfortunately, both factors can hardly be anticipated when designing a web mapping application. If the scenario demands only a few hundred points, typical loading times will be within an acceptable range. However, if thousands of points have to be transferred to the client, the loading and parsing of the data might consume too much time. If the implementation is programmed poorly, the parsing of the data might even lock up the browser.

In such situations, a server-side processing of the data might be advisable. Alternatively, the filtering mechanisms discussed in chapter 3 may help to reduce (visual) overload, resulting in a smooth and satisfying map behaviour.

Recalling the results of the online survey, almost every participant (90 %) stated to own a smartphone. Furthermore, almost 90 % of the surveyed persons answered to use online mapping services frequently. The chance that a user will access a map mashup with a mobile device can be considered to be very likely. As the processing power of mobile devices is limited, compared to typical multi-core desktop computer processors, the usage of a real time clustering approach can lead to unsatisfying results, if the amount of features is too high, as shown in chapter 7.3. Consequently, the implementation of a real time clustering approach on the client might not always be the best solution.

9.3 Recommendations

The purpose of cluster symbols on maps is to indicate the location of a cluster, and the amount of features contained within it. Recalling the definition of data types, in this case, the nature of the displayed information is quantitative. Irrespective of the type of clustering algorithm chosen, a key recommendation evolved from the investigations performed during this thesis is:

Whenever a map maker decides to use a clustering plugin, he should make use of possibility to customize the style of the cluster symbolizations, and follow the best practises described in the chapters 2.5.2 and 2.5.3. That is to use and manipulate the *proper visual variables* for displaying quantitative data: *size* and *colour value*. Unfortunately, almost every available implementation by default uses unsuitable variables, which might confuse the map users and lead to misinterpretations. In the following Figure 43, screenshots of existing point cluster algorithms have been collected to demonstrate this unfortunate situation.

10 CONCLUSIONS AND OUTLOOK

The goal of this thesis was to investigate the feasibility of a grid-based clustering approach for real time web mapping solutions. Therefore, a prototype implementation of a real time grid-based clustering approach was designed and developed, and the prototype was tested and compared in means of computing performance with existing methods of point agglomeration. Besides the technical analysis, the acceptance by typical internet users was investigated, as well. A survey helped to analyse the performance of this alternative approach, and valuable feedback and insights could be gained from the surveyed participants.

The analysis has shown that methods for point clustering in map mashups are still in an early stage of development. Expertise from computer science and cartography should be seamlessly integrated to provide better solutions, in technical and cartographical terms. Furthermore, it became clear that the cluster solution has to be chosen for every web mapping project individually, as requirements and conditions strongly vary depending on the scope of the map and the nature of the mapped data.

A second major insight gained throughout this thesis is that in today's web mapping, many maps but also technical frameworks are designed and developed by a non-cartographic audience. As a consequence, developers and map makers are unaware of certain design principles which can lead to negative effects on the quality of the created maps. For the adequate representation of thematic phenomena on a map, a crucial step lies in the selection of the right map projection, and the lack of variety in today's base maps hinders quality improvements. This thesis has also shown that the feasibility of grid-based clustering approaches strongly depends on the chosen map projection. Only with equal-area projections, the cells are visually comparable, distortions in scale lead to confusion.

Improvements can be expected with further development in vector based mapping technologies. Depending on the scope of the map, already today maps can be created without the usage of prerendered base maps, giving the possibility to choose an adequate, equal area map projection. In addition, more advanced and intelligent clustering algorithms could be developed when using the additional feature information from vector derived base maps.

The key to better maps and data visualizations on the web lies in the improvement of the existing popular web mapping frameworks. Only when these APIs will be enhanced, more cartographically pleasing results will be seen in online map mashups, and mapping applications.

Concerning the performance of the developed algorithm, the integration of a spatial quadtree, as done by BEREUTER & WEIBEL (2013) could increase the computing performance for the grid-based clustering algorithm. Moreover, the set of generalization operators could be enhanced, as well.

Further investigation may be needed to explore the feasibility of different cell-geometries. The application of hexagons might result in a smoother, visually more attractive map picture. However, the calculation and feature aggregation for this case is not trivial. It would have to be evaluated, if the performance of a typical (mobile) client is already sufficient to calculate those geometries in real time.

The evaluation of existing implementations of point clustering algorithms has shown that from a cartographical point of view, the symbolizations used to indicate the count of features for each cluster, is actually wrong. Instead of manipulating the colour hue, which is a variable for qualitative data, adequate visual variables for quantitative data like the symbol size or the colour value should be chosen for the symbolization of the clusters. Further investigation about the implementation of these variables into a real time clustering approach would be benefiting, as the right symbolization technique would help to improve the cartographic communication with the user, and hence to increase the legibility of the map.

As mobile devices are established more and more in our daily lives, the way, people use web maps shifts away from classical desktop or laptop computers to smartphones and tablets. These devices demand for specific mapping techniques and solutions, as the input devices as well as the smaller screen estate lead to new issues concerning map navigation and map legibility. A grid-based clustering algorithm might help to declutter maps, and thus to improve the user experience.

Concluding, to validate the working hypothesis of this research, the investigations have proven that under certain circumstances, a grid-based clustering algorithm can already serve as an alternative to the existing real time, distance-based clustering methods. However, the eligibility strongly varies on the scope of the map and the calculation power of the used device. The survey revealed that for the usage in map mashups, people preferred the distance-based method, although it was perceived controversially. A major reason might be the simple fact, that some people prefer things they already know, while others are open to new approaches. As only 40 participants could be interviewed, a bigger audience would be necessary to gain more insights about the preference of users.

However, the survey clearly indicated that this approach is highly eligible for the application in thematic web mapping, especially for the animation of spatiotemporal data series. These findings correspond to the evaluation of the cartographic theory about data visualization, which recommend the application of grid-based structures, as well.

With further improvements in the calculation performance of the algorithm, as well as the implementation of enhanced visualization techniques like the use of proportional symbols, the eligibility of the prototype could be increased for various settings, and could help to bring more variety to the available real time clustering solutions for web mapping applications.

LITERATURE

ARNBERGER, E. (1977): Thematische Kartographie. Mit einer Kurzeinführung über Automation in der thematischen Kartographie. Braunschweig: Westermann (Das Geographische Seminar).

AUER, M. (2012): Realtime Web GIS Analysis using WebGL. *International Journal of 3-D Information Modeling (IJ3DIM)*, Special Issue on Visualizing 3D Geographic Information on the Web. Special Issue on: 3D Web Visualization of Geographic Data. IGI-Global.

BATTERSBY, S. E.; FINN, M. P.; USERY, E. L.; YAMAMOTO, K. H. (2014). Implications of Web Mercator and Its Use in Online Mapping. *Cartographica: The International Journal for Geographic Information and Geovisualization* 49(2), 85-101. University of Toronto Press.

BERTIN, J. (1983): *Semiology of Graphics: Diagrams, Networks, Maps*. Madison, Wisconsin: University of Wisconsin Press.

BLOK, C. (2000): Monitoring Change: Characteristics of Dynamic Geo-spatial Phenomena for Visual Exploration. In: FREKSA et al.: *Spatial Cognition II*, LNAI 1849. Berlin; Heidelberg: Springer, 16-30.

BOSTOCK, M. (2012a): Spherical Mercator. Online: <http://bl.ocks.org/mbostock/3757132>, cited: 24.03.2015.

BOSTOCK, M. (2012b): Lambert Azimuthal Equal-Area. Online: <http://bl.ocks.org/mbostock/3757101>, cited: 24.03.2015.

BURIGAT, S.; CHITTARO, L. (2008): Decluttering of Icons Based on Aggregation in Mobile Maps. In: MENG, L., ZIPF, A. & WINTER, S. (Ed.): *Map-based Mobile Services*. Berlin; Heidelberg: Springer, (Lecture Notes in Geoinformation and Cartography), 13–32.

CARTWRIGHT, W. (2003): Maps on the Internet. In: PETERSON, M. (Ed.): *Maps and the Internet*. Oxford: Elsevier Science Ltd, 35-56.

COCKBURN, A. (2001): *Writing Effective Use Cases*, Boston, MA: Addison-Wesley.

DELORT, J.Y. (2010): Hierarchical Cluster Visualization in Web Mapping Systems. In: *Proceedings of the 19th International Conference on World Wide Web*. New York, NY, USA: ACM (WWW '10), 1241–1244. Online: <http://doi.acm.org/10.1145/1772690.1772892>, cited: 24.03.2015.

DETWILER, J. (2014): Evolution of Web Mapping Technology. Online: https://www.e-education.psu.edu/geog863/resources/13_p3.html, cited: 24.03.2015

EDWARDES, A.; BURGHARDT, D.; WEIBEL, R. (2005): Portrayal and Generalisation of Point Maps for Mobile Information Services. In: MENG, L.; ZIPF, A.; REICHENBACHER, T. (Ed.): *Map-based Mobile Services: Theories, Methods and Implementations*. Heidelberg: Springer, 11-30.

ESRI (2014): ArcGIS Viewer for Flex – Clustering. Online: <http://resources.arcgis.com/en/help/flex-viewer/concepts/index.html#//01m30000004z000000>, cited: 24.03.2015.

- FIELD, K. (2012): Using a binning technique for point-based multiscale web maps. ESRI. Online: <http://blogs.esri.com/esri/arcgis/2012/06/08/using-a-binning-technique-for-point-based-multiscale-web-maps/>, cited: 24.03.2015.
- GAFFURI, J. (2012): Toward Web Mapping with Vector Data. In: XIAO, KWAN et al. (Ed.) – Geographic Information Science. 7th International Conference, GIScience 2012, Columbus, OH, USA, September 18-21, 2012, Proceedings. New York: Springer, 87–101.
- GARRETT, J. (2005): Ajax: A new Approach to Web Applications. Online: <http://web.archive.org/web/20080702075113/http://www.adaptivepath.com/ideas/essays/archives/000385.php>, cited: 24.03.2015.
- GAN, G.; MA, C.; WU, J. (2007): Data clustering: theory, algorithms, and applications. Philadelphia, Pa.: SIAM.
- GOOGLE (n.d.): MarkerClusterer for Google Maps v3. Online: <http://google-maps-utility-library-v3.googlecode.com/svn/trunk/markerclusterer/docs/reference.html>, cited: 24.03.2015.
- HAKE, G.; GRÜNREICH, D.; MENG, L. (2002): Kartographie: Visualisierung raum-zeitlicher Informationen. Berlin, New York: de Gruyter.
- HUANG, H.; GARTNER, G. (2012): A Technical Survey on Decluttering of Icons in Online Map-based Mashups. In: PETERSON, M. (Ed.): Online Maps with APIs and WebServices, Lecture Notes in Geoinformation and Cartography, Heidelberg: Springer, 157–175.
- JOHNSON, Z.F. (2011): Hexbins! Online: <http://indiemaps.com/blog/2011/10/hexbins/>, cited: 24.03.2015.
- JOKAR ARSANJANI, J.; MOONEY, P.; HELBICH, M.; ZIPF, A. (2015; accepted): An exploration of future patterns of the contributions to OpenStreetMap and development of a Contribution Index, Transactions in GIS. Wiley.
- KEIM, D.; PANSE, C.; SLIPS, M. (2005): Information Visualization: Scope, Techniques and Opportunities for Geovisualization. In: DYKES, J. (Ed.): Exploring Geovisualization. 1. Ed. Amsterdam: Elsevier, 23–52.
- KRAAK, M.J. (2001): Web Cartography: Developments and Prospects. London: Taylor & Francis.
- KRAAK, M.J. ; ORMELING, F. (2003): Cartography: Visualization of Geospatial Data. (2.nd ed.). Harlow: Prentice Hall.
- KRIZ, K. (2009): Are we Living in a Cartographic Illiterate Society? In: LEHN, A.; GARTNER, G.; CARTWRIGHT, W (Ed.): Cartography and Art (Lecture Notes in Geoinformation and Cartography). Berlin, Heidelberg: Springer, 1-10. Online: http://dx.doi.org/10.1007/978-3-540-68569-2_6, cited: 24.03.2015.
- KHRONOS (2011): What Is WebGL? Online: https://www.khronos.org/webgl/wiki/Getting_Started, cited: 24.03.2015
- KRUG, S. (2005): Don't make me think: a common sense approach to Web usability. Berkeley, CA: New Riders.

- KRYGIER, J.; WOOD, D. (2011): *Making Maps: A visual Guide to Map Design for GIS*. New York, NY: Guilford Press.
- LEAFLETJS (2014a): Reference – Iprojection. Online: <http://leafletjs.com/reference.html#iprojection>, cited: 24.03.2015
- LERNER, R. (2006): At the Forge - Creating Mashups. *Linux Journal* 147/2006. Online: <http://www.linuxjournal.com/article/8984>, cited: 24.03.2015.
- LIENERT, C.; JENNY, B.; SCHNABEL, O.; HURNI, L. (2012): Current Trends in Vector-Based Internet Mapping: A Technical Review. In: PETERSON, M. (Ed.): *Online Maps with APIs and WebServices*, Lecture Notes in Geoinformation and Cartography, Heidelberg: Springer, 23-34.
- MAHE, L.; BROADFOOT, C. (2010): Too Many Markers! Google Geo APIs Team. Online: <https://developers.google.com/maps/articles/toomanymarkers?hl=de>, cited: 24.03.2015.
- MAC EACHREN, A. (1995): *How maps work: representation, visualization, and design*. New York: Guilford Press.
- MEIER, J.; FARRE, C.; BANSODE, P.; BARBER, S.; REA, D. (2007): *Performance Testing: Guidance for Web Applications*. Online: <http://perftestingguide.codeplex.com/downloads/get/17955>, cited: 24.03.2015.
- MENG, L.; ZIPF, A.; REICHENBACHER, T. (2005): *Map-based Mobile Services: Theories, Methods and Implementations*. Heidelberg: Springer.
- MUEHLENHAUS, IAN (2014): *Web Cartography. Map Design for Interactive and Mobile Devices*. Boca Raton, FL: CRC Press.
- NEUMANN, A. (2012): *Web Mapping and Web Cartography*. In: KRESSE, W. & DANKO, D. M. (Ed.): *Springer handbook of geographic information*. Berlin; New York: Springer, 568–586.
- NIELSEN, J. (1993): *Usability Engineering*. Boston: AP Professional.
- OGAO, P. (2002): *Exploratory visualization of temporal geospatial data using animation*. – Dissertation, University Utrecht, Utrecht.
- OGC (2007): *Styled Layer Descriptor profile of the Web Map Service: Implementation Specification*. Online: <http://www.opengeospatial.org/standards/sld>, cited: 24.03.2015
- OPENLAYERS (n.d.): *OpenLayers.Strategy.Cluster*. Online: <http://dev.openlayers.org/releases/OpenLayers-2.13.1/doc/apidocs/files/OpenLayers/Strategy/Cluster-js.html>, cited: 24.03.2015.
- OPENSTREETMAP (2015a): *Slippy Map*. Online: http://wiki.openstreetmap.org/wiki/Slippy_Map, cited: 24.03.2015
- OPENSTREETMAP (2015b): *History of OpenStreetMap* http://wiki.openstreetmap.org/wiki/History_of_OpenStreetMap, cited: 24.03.2015
- PEHNKE, O.; SCHMID, B. (2012): Per Anhalter durch JavaScript. In: *JavaSPEKTRUM* (2), 8–12. Online: <http://www.sigs->
-

datacom.de/fileadmin/user_upload/zeitschriften/js/2012/02/pehnke_schmid_JS_02_12.pdf, cited: 24.03.2015.

PETERSON, M. (2003): Chapter 1 - Maps and the Internet: An Introduction. In: PETERSON, M. (Ed.): Maps and the Internet. Oxford: Elsevier Science (International Cartographic Association), 1–16.

PETERSON, M. (2008): Maps and the Internet: What a Mess It Is and How To Fix It. In: cartographic perspectives, 59, 4-11.

PETERSON, M. (2012): Online Maps with APIs and WebServices. Heidelberg: Springer.

ROICK, O.; HAGENAUER, J.; ZIPF, A. (2011): OSMatrix – Grid-based Analysis and Visualization of OpenStreetMap. State of the Map, Wien, July 2011.

ROICK, O.; LOOS, L.; ZIPF, A. (2012): A Technical Framework for Visualizing Spatio-temporal Quality Metrics of Volunteered Geographic Information. In: LÖWNER, M.-O; HILLEN, F.; WOHLFAHRT, R. (Ed.): Geoinformatik 2012 - Mobilität und Umwelt. Konferenzband, 28. - 30. März 2012, Braunschweig. 1. Aufl. Herzogenrath: Shaker (Berichte aus der Geoinformatik), 263–270.

ROSENBERG, M (n.d.): MapQuest: An Overview of MapQuest. Online: <http://geography.about.com/od/streetroadcitymaps/a/mapquest.htm>, cited: 24.03.2015

SARODNICK, F.; BRAU, H. (2011): Methoden der usability evaluation. Verlag Hans Huber.

SHNEIDERMAN, B. (1996): The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In: Proceedings of the IEEE Symposium on Visual Languages, Washington: IEEE Computer Society Press, pp.336-343.

SUTCLIFFE, A. (2003): Scenario-based requirements engineering, mini-tutorial. 11th IEEE International Requirements Engineering Conference, Monterey Bay, CA, 8-12 Sept. 2003, pp.320-329.

TREVES, R. (2011): Point Clustering Usability Example. Online: <http://webmapdesign.blogspot.cz/2011/09/point-clustering-usability-example.html>, cited: 24.03.2015.

TURNER, A. (2006): Introduction to Neogeography. Sebastopol, Calif: O'Reilly.

W3C (2011): Scalable Vector Graphics (SVG) 1.1 (Second Edition). Online: <http://www.w3.org/TR/SVG/>, cited: 24.03.2015

W3C (2014): The Canvas Element. Online: <http://www.w3.org/TR/html5/scripting-1.html#the-canvas-element>, cited: 24.03.2015

WOOD, J.; DYKES, J.; SLINGSBY, A.; CLARKE, K. (2007): Interactive Visual Exploration of a Large Spatio-temporal Dataset: Reflections on a Geovisualization Mashup. In: IEEE Transactions on Visualization and Computer Graphics 13 (6), 1176–1183.

YUN, S. (2007): The User-participated Geospatial Web as Open Platform. Hg. v. The 11. International Seminar on GIS.

Links:

BOOKING-1: Paris. Online: <http://www.booking.com/city/fr/paris.html>, cited: 24.03.2015

CITYBIKES-1: CityBikes API Documentation. Online: <http://api.citybik.es/v2/>, cited: 24.03.2015

FLEX-1: Clustering Example. Online: <https://developers.arcgis.com/flex/sample-code/clustering.htm>

GOOGLE-1: MarkerCluster Example. Online: https://googlemaps.github.io/js-marker-clusterer/examples/advanced_example.html, cited: 24.03.2015

LIMESURVEY-1. Online: <http://limesurvey.org/>, cited: 24.03.2015

LVISA-1. Online: <http://lvisa.geog.uni-heidelberg.de/sigdb-client/release/source/explore.php>, cited: 24.03.2015

MARINETRAFFIC-1: Online: <https://www.marinetraffic.com/>, cited: 24.03.2015

MARKERCLUSTER-1: Online: <https://github.com/Leaflet/Leaflet.markercluster>, cited: 24.03.2015

MARKERCLUSTER-2: Clustering Example. Online: <http://leaflet.github.io/Leaflet.markercluster/example/marker-clustering-realworld.388.html>, cited: 24.03.2015

MARKERCLUSTER-3: Examples. Online: <https://github.com/Leaflet/Leaflet.markercluster/tree/master/example> , cited: 24.03.2015

OPENLAYERS-1: Clustering Example. Online: <http://openlayers.org/en/v3.2.1/examples/cluster.html?q=cluster>, cited: 24.03.2015

OSMATRIX-1. Online: http://koenigstuhl.geog.uni-heidelberg.de/osmatrix/#diff/allotments_area/10/0.0274658203125/51.470691106434884?start=7&end=8, cited: 24.03.2015

TOPOJSON-1: TopoJSON. Online: <https://github.com/mboostock/topojson>, cited: 24.03.2015

LIST OF FIGURES

Figure 1: Example of suboptimal symbolization resulting in a cluttered map. (http://en.velib.paris.fr/Stations-in-Paris)	4
Figure 2: Stages of methodology used in the thesis	6
Figure 3: The Xerox PARC Map Viewer, one of the first interactive maps on the internet. (DETWILER 2014).....	12
Figure 4: selected events in the history of web mapping services (PETERSON 2012, 14).	13
Figure 5: Comparison of synchronous and asynchronous client-server communication methods (PETERSON 2012, 9).	15
Figure 6: Composition of a typical web mapping application. Sources (clockwise from bottom left: OpenStreetMap / Stadt Wien, MA 41 / ASTER GDEM / Stadt Wien MA 22).....	17
Figure 7: Combination of raster- and vector-based mapping technologies. The vector layer representing tree objects can be modified on the fly. Left: classification by family membership; right: filter for a specific tree species (LVISA-1).....	21
Figure 8: Left: Mercator projection as commonly used in base maps. (BOSTOCK, M. 2012a) Right: Lambert Azimuthal Equal-Area (BOSTOCK, M. 2012b). Note how the polar regions are distorted on the left.	22
Figure 9: Scaling of point symbols on various scale levels (EDWARDES et al. 2005, 14)	23
Figure 10: Classification of phenomena characteristics, and possible map forms (MACEACHREN 1995, 303 & 304)	26
Figure 11: Visual variables, after KRYGIER & WOOD (KRYGIER & WOOD 2011, 177).....	30
Figure 12: Cluster symbols indicating amount of contained features with varying icon size..	31
Figure 13: Procedures of cartographic generalization. (Edited; HAKE et al. 2003, 169).....	33
Figure 14: The result for a hotel search in Paris on a booking portal. Unnecessary POIs increase the map complexity. (Source: BOOKING-1).....	35
Figure 15: Illustration of the distance-based algorithm (using the Leaflet.js Markercluster plugin. Base map: © OpenStreetMap contributors)	36
Figure 16: Examples of a hexagonal-shaped grid (left) and a square-shaped grid (right). (OSMATRIX-1 & MARINETRAFFIC-1).....	38
Figure 17: Grid-based cluster visualizations. Left: icons positioned in the centre of the cell, right: adapted location based on distribution of the underlying features.	39
Figure 18: Results of the Markercluster algorithm. Left: Zoom Level 14, Right: Zoom Level 15. (Map source © OSM contributors).....	49
Figure 19: Result of the grid-based algorithm using dynamic cluster symbol positioning based on the mean location of contained features. Left: Zoom 14, Right: Zoom 15 (map source: © OSM contributors).....	49
Figure 20: Result of the grid-based algorithm using laminar symbolizations. Left: Zoom 14, Right: Zoom 15 (map source: © OSM contributors)	50
Figure 21: From left to right: MarkerCluster, grid-based markers, grid-based polygons.	51
Figure 22: Workflow of the grid-based clustering algorithm.....	55
Figure 23: Average time needed for code execution in the first test scenario (10 000 features).	60
Figure 24: Average time needed for code execution in the second test scenario (50 000 features).	61

Figure 25: Average time needed for code execution in the third test scenario (10 000 features).	63
Figure 26: Average time needed for code execution in the fourth test scenario (50 000 features).	64
Figure 27: Internet experience of the interviewed audience (N=40).	68
Figure 28: Possession of digital devices. (N=40)	68
Figure 29: Results of the question, how easy the location of the bike stations could be determined (N=40).	69
Figure 30: Results of the ratings for unnecessary map complexity (N=40).	70
Figure 31: Usage of cluster representations as landmarks (N=40).	70
Figure 32: Perception of marker behaviour on zoom change (N=40).	71
Figure 33: How intuitive was the grid-based clustering algorithm using colour-coded cells as representations to the participant? (N=40).	71
Figure 34: Eligibility to fulfil the task of locating bike stations (N=40).	72
Figure 35: Ranking of the visualizations based on popularity (N=40).	72
Figure 36: Results indicate that grid lines may serve as orientation support on web maps (N=40).	73
Figure 37: Regions with the strongest developments, according to the participants.	74
Figure 38: Understanding of Incident-Allocations (N=40).	75
Figure 39: Understanding of the Map (N=40).	75
Figure 40: Representation of regions by cluster symbolizations (N=40).	76
Figure 41: The visualization was suitable to communicate the data (N=40).	76
Figure 42: Ranking based on popularity for the second use case (N=40).	77
Figure 43: Composition of existing real time point-clustering solutions. Clockwise from top left: Google Maps MarkerCluster v3 (Google-1), ESRI ArcGIS Flex (Flex-1), OpenLayers 3.0 (OpenLayers-1), Leaflet.MarkerCluster (MarkerCluster-3)	81

LIST OF TABLES AND LISTINGS

Table 1: Visual hierarchies for map elements of web maps. (Modified from MUEHLENHAUS 2014, 64)	28
Table 2: Composition of scenarios for performance evaluations.	59
Table 3: Time needed to compute 10 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 1).	60
Table 4: Time needed to compute 50 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 2).	61
Table 5: Time needed to compute 50 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 3).	62
Table 6: Time needed to compute 50 000 points, relative to the performance of the MarkerCluster algorithm (Scenario 4).	63
Listing 1: Definition of a cluster cell object	56
Listing 2: Styling of the Cluster Symbolizations using CSS rules.	57
Listing 3: List of all available options to customize the clustering algorithm.	58
Listing 4: Minimal setup of the grid-based clustering algorithm.	58

APPENDIX

Web Mapping: Clustering Techniques

Dear participant!

Thank you for your interest in participating in this small online survey. The goal of this survey is to gain insight into the usability of different visualization techniques in web maps. Your feedback is highly appreciated.

Your answers will be stored in an anonymized way.

If you have further questions, don't hesitate to write me.

Thank you!

Andreas Kiefer

andreas.kiefer@univie.ac.at

There are 18 questions in this survey

Use Case 1: Markercluster

Bike stations in Paris (1)

In the video clip below, you see a map of Paris, France with all existing bike stations.

As there are about 7000 Stations, they may be grouped together into *Cluster Markers*, depending on how close they are to each other (this is called *marker clustering*).

For single stations, a green circle means "bikes available" and red means "no bikes available".

The red marker is your position.



Your task is to find the bike station closest to you.

Please watch this short video clip. When you are done, rate your impressions in the questions below.

[]How do you rate the following statements? *

Please choose the appropriate response for each item:

	No	Rather no	Not Sure	Rather yes	Yes
It was easy to figure out the bike stations near the red marker	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the map unnecessarily complex	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I used the cluster markers as landmarks (for orientation)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the change of the markers locations (on zoom) irritating	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was suitable to fulfill the task	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Use Case 1: Grid Cluster

Bike stations in Paris (2):

In the video clip below, you see the same situation as before. This time, a grid was laid over the map, and the bike stations were grouped into the cells.

Again, for single stations, a green circle means "bikes available" and red means "no bikes available". Multiple stations are

grouped together into *Cluster Markers*.

138

The red marker is your position.  You need to find a bike station closest to you.

Please watch this short video clip. When you are done, rate your impressions in the questions below.

[] How do you rate the following statements? *

Please choose the appropriate response for each item:

	No	Rather no	Not Sure	Rather yes	Yes
It was easy to figure out the bike stations near the red marker	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the map unnecessarily complex	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I used the cluster markers as landmarks (for orientation)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the change of the markers locations (on zoom) irritating	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was suitable to fulfill the task	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

[] Did you use the gray grid lines for orientation on the map? *

Please choose **only one** of the following:

- ☐ Yes
☐ No

Use Case 1: Grid Cluster (2)

Bike stations in Paris (3)

In the video clip below, you see the same situation as before. This time, a grid was laid over the map, and the bike stations were **grouped into the cells**.

Multiple stations are grouped together into *Cells*. The more (darker) red a cell, the more bike stations are in it.



The *red marker* is your position. **You need to find a bike station closest to you.**

Please watch this short video clip. When you are done, rate your impressions in the questions below.

[]How do you rate the following statements?

Please choose the appropriate response for each item:

	No	Rather no	Not Sure	Rather yes	Yes
It was easy to figure out the bike stations near the red marker	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the map unnecessarily complex	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I used the cluster cells as landmarks (for orientation)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was intuitive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was suitable to fulfill the task	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Use Case 1: Comparison

[]

Which visualization technique did you like more?

A short reminder:

- **The first technique grouped the stations according to their density**
- **The second technique grouped the stations into regular grid cells**
- **The third technique grouped the station into colored cells**

*

All your answers must be different.

Please number each box in order of preference from 1 to 3

The first one (Markercluster)

The second one (Grid Cluster)

The third one (colored cells)

[]Please write, why:

Please write your answer here:

Use Case 2: Markercluster

Time Series (1):

In the video clip below, you see a map of Europe. An animation shows the temporal development of (made-up) incidents from 2001 to 2010.

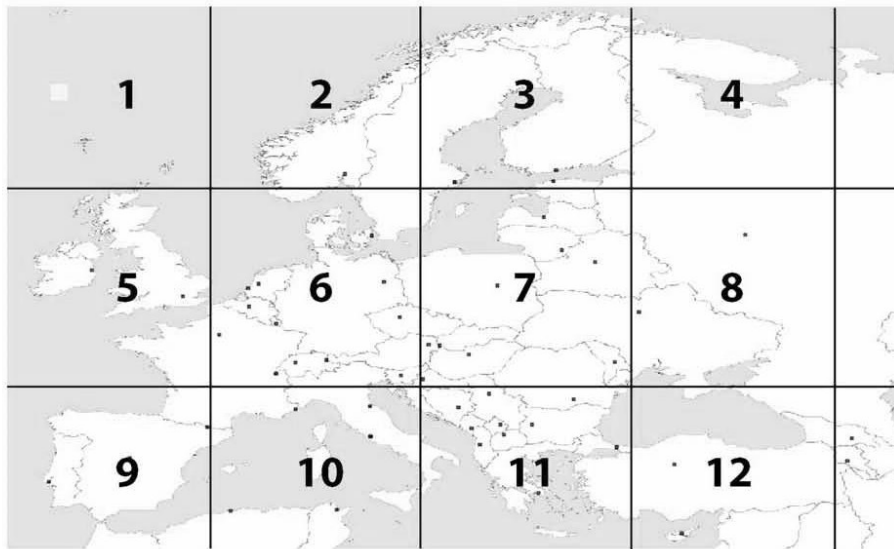
The incidents are grouped into clusters, according to the distance to each other. Each number represents the amount of incidents, that happend during the active year.

Please watch this short video clip. When you are done, rate your impressions in the questions below.

[]

In which regions did you see the strongest developments?

Please choose them from the map and select the number(s):



*

Please choose **all** that apply:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 5
- ☐ 6
- ☐ 7
- ☐ 8
- ☐ 9
- ☐ 10
- ☐ 11
- ☐ 12

[]How do you rate the following statements? *

Please choose the appropriate response for each item:

	No	Rather no	Not Sure	Rather yes	Yes
It was easy to understand the allocation of incidents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I thought the map was easy to understand	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was suitable to communicate the data	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It was clear which region each cluster represents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Use Case 2: Grid Cluster

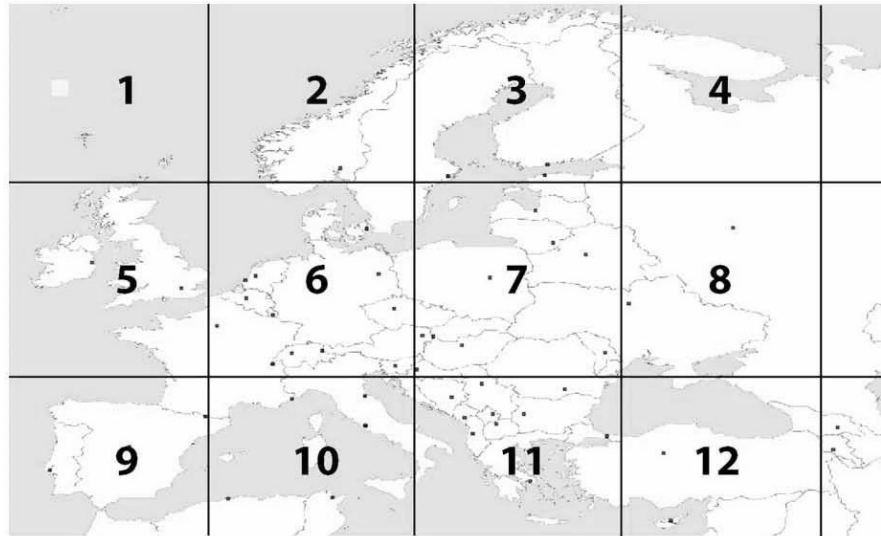
Time Series (2):

In the video clip below, you see the same scenario like before. This time, the incidents are grouped into grid cells.

The number for each cluster indicates the amount of incidents for the given year.

Please watch this short video clip. When you are done, rate your impressions in the questions below.

[]

In which regions did you see the strongest developments?**Please choose them from the map and select the number(s):**

*

Please choose all that apply:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 5
- ☐ 6
- ☐ 7
- ☐ 8
- ☐ 9
- ☐ 10
- ☐ 11
- ☐ 12

[]How do you rate the following statements? *

Please choose the appropriate response for each item:

	No	Rather no	Not Sure	Rather yes	Yes
It was easy to understand the allocation of incidents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I thought the map was easy to understand	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was suitable to communicate the data	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It was clear which region each cluster represents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

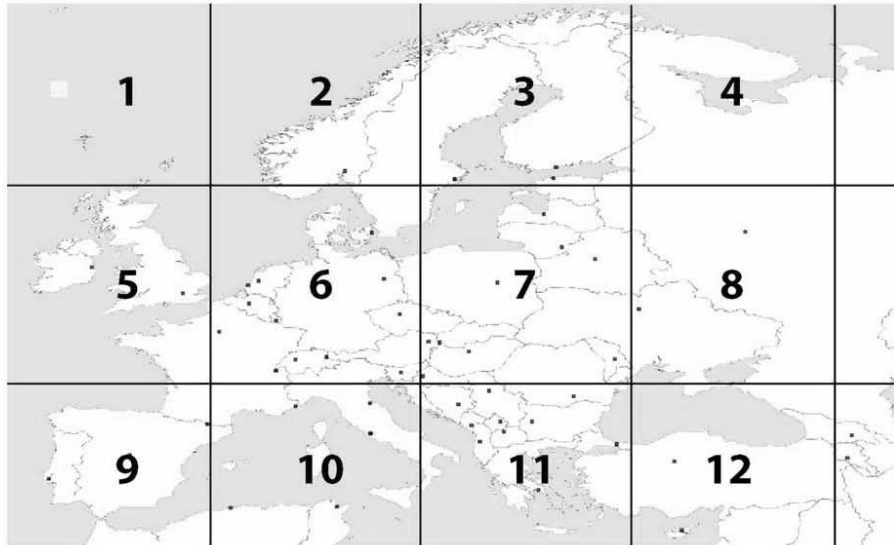
Use Case 2: Grid Cluster (2)**Time Series (3):**

In the video clip below, you see the same scenario like before. This time, the incidents are represented using laminar symbolizations: the grid cells are colorized. The more (darker) red a cell, the more incidents happend there.

The number for each cluster indicates the amount of incidents for the given year.

Please watch this short video clip. When you are done, rate your impressions in the questions below.

[]

In which regions did you see the strongest developments?**Please choose them from the map and select the number(s):**

*

Please choose **all** that apply:

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 5
- ☐ 6
- ☐ 7
- ☐ 8
- ☐ 9
- ☐ 10
- ☐ 11
- ☐ 12

[]How do you rate the following statements? *

Please choose the appropriate response for each item:

	No	Rather no	Not Sure	Rather yes	Yes
It was easy to understand the allocation of incidents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I thought the map was easy to understand	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The visualization was suitable to communicate the data	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It was clear which region each cluster represents	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Use Case 2: Comparison**[]****Which visualization technique did you like more?****A short reminder:**

- **The first technique grouped the incidents according to their density (using markers)**
- **The second technique grouped the incidents into regular grid cells (using markers)**
- **The third technique used color-coded cells to visualize the incidents**

All your answers must be different.

Please number each box in order of preference from 1 to 3

<input type="text"/>	The first one (Markercluster)
<input type="text"/>	The second one (Grid Cluster)
<input type="text"/>	The third one (colored cells)

[]Please write, why:

Please write your answer here:

Internet Experience

[]Your Internet Experience *

Please choose the appropriate response for each item:

	Never	Rarely	Sometimes	A lot	All the time
If I need information, I use the web for research	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
If I travel to a new destination, I use the web to plan the trip	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I use online banking	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I buy products online	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I use Social Networks (Facebook, Twitter, ...)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I use mapping services like Google Maps or Bing Maps	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

[]I own the following products *

Please choose **all** that apply:

- ☐ Computer / Laptop
- ☐ Smartphone
- ☐ Tablet
- ☐ Navigation System (dedicated or App)

Demographic Questions

[]Please select your gender *

Please choose **only one** of the following:

- ☐ female
- ☐ male
- ☐ no answer

[]How old are you? *

Only numbers may be entered in this field.
Each answer must be at most 120

Please write your answer here:

Thank you for your feedback!

Have a nice day.

You can now close this window.

Submit your survey.

Thank you for completing this survey.

ANDREAS KIEFER, B.Sc.

BILDUNG

10/2012 – 05/2015 (voraussichtlich)	Kartographie und Geoinformation (M.Sc.), Universität Wien
10/2008 – 08/2012	Geographie (B.Sc.), Ruprecht-Karls-Universität Heidelberg 09/2010 – 02/2011: Auslandssemester ERASMUS, „Univerzita Karlova v Praze“, Prag
10/2007 – 08/2008	Angewandte Informatik (B.Sc.), Ruprecht-Karls-Universität Heidelberg
09/1997 – 06/2006	Theodor Heuss Gymnasium, Mühlacker

BERUFLICHE ERFAHRUNGEN

02/2014 – 05/2014	Auftragsarbeit für die Österreichische Akademie der Wissenschaften, Institut für Stadt- und Regionalforschung
03/2013 – 12/2014	Projektmitarbeiter, Institut für Geographie und Regionalforschung, Abteilung Kartographie und Geoinformation, Universität Wien
02/2012 – 10/2014	Studentische Hilfskraft, Geographisches Institut, Abteilung Geoinformatik, Universität Heidelberg (http://lrg.geog.uni-heidelberg.de)
02/2012	Tutor für die Einführungskurse in SPSS 18, Geographisches Institut, Abteilung Anthropogeographie, Universität Heidelberg
03/2011 – 08/2011	Praktikum, Tchibo GmbH, Hamburg, Bereich Expansion & Immobilienmanagement
10/2011 – 09/2012 und 10/2009 – 08/2010	Studentische Hilfskraft, Geographisches Institut, Abteilung Anthropogeographie, Universität Heidelberg

PUBLIKATIONEN

KOENIG, K., KIEFER, A. & HÖFLE, B. (2013): Web-based visualization and object-based analysis of 3D geoinformation from laser scanning point clouds. *gis.SCIENCE*. Vol. 26(2), pp. 70-76.

KOENIG, K., KIEFER, A., PETERS, R. & HÖFLE, B. (2014): Webbasierte Visualisierung und Analyse von Vegetationsobjekten aus Laserpunktwolken. *Geoinformatik 2014, Hamburg, Germany*.

KOENIG, K., KIEFER, A. & HÖFLE, B. (2013): Laser Scanning for 3D Vegetation Characterization: Web-based Infrastructure for Exploration and Analysis of Vegetation Signatures. In: *Proceedings of the ISPRS WG VII/5 ISPRS Workshop*. 09.-10.09.2013, Cologne, Germany, pp. 153-154.

WEITERE QUALIFIKATIONEN

Sprachen:	Deutsch (Muttersprache), Englisch (fließend in Wort und Schrift), Französisch (Grundkenntnisse), Spanisch (B1), Tschechisch (A1), Russisch (A1)
Programmierkenntnisse:	JavaScript (sehr gut - u.a. jquery, ExtJS, OpenLayers), PHP (gut), SQL (gut, MySQL, PostgreSQL), Python (gut - arcpy), Java (Grundkenntnisse) Webprogrammierung (sehr gut - OGC Standards, HTML 5, CSS 3)
Software / EDV:	GI Software (sehr gut – Esri ArcGIS, Q-GIS, Geoserver), Erdas Imagine (gut), Linux / Unix Systeme (gut), MS Office, Adobe Illustrator und Photoshop, MS Access, SPSS
Portfolio:	http://homepage.univie.ac.at/andreas.kiefer/

Ich versichere:

- dass ich die Masterarbeit selbstständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfe bedient habe.
- dass alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Publikationen entnommen sind, als solche kenntlich gemacht sind.
- dass ich dieses Masterarbeitsthema bisher weder im In- noch im Ausland (einer Beurteilerin/ einem Beurteiler zur Begutachtung) in irgendeiner Form als Prüfungsarbeit vorgelegt habe.
- dass diese Arbeit mit der vom Begutachter beurteilten Arbeit übereinstimmt.

Datum

Unterschrift