Lappeenranta University of Technology
School of Business and Management
Degree Program in Computer Science

Karri Väänänen

Visualizations as tools for finding deeper insight into data

Examiners: Jari Porras, Professor

Ari Happonen, D.Sc. (Tech.)

Supervisor: Ari Happonen, D.Sc. (Tech.)

Abstract

Lappeenranta University of Technology
School of Business and Management
Degree Program in Computer Science

Karri Väänänen

Visualizations as tools for finding deeper insight into data

Master's Thesis. 2017. 89 pages, 24 figures, 1 table, and 3 appendices.

Examiners: Jari Porras, Professor
Ari Happonen, D.Sc. (Tech.)

Keywords: Data visualization, insight, exploratory data analysis, web-based visualization, Cesium framework, traffic, GIS, spatio-temporal data, data mining

In this thesis, data and information visualization are studied as ways to generate deeper insight into data. Open data from traffic and public transportation is used in developing web-based spatio-temporal visualizations. Also, the thesis aims to study the processes around exploratory data analysis and the role of human cognition concerning visualizations.

Information visualization, data mining and visualization processes, and human cognition in the context of visualizations are studied from literature. The commonalities in visualization creation and data analysis process are examined. In addition, web-based 2D and 3D visualization artefacts are created in an iterative process. The data management of visualization prototypes, including data discovery, collection, and preprocessing, is discussed. Also, a predictive machine learning method is briefly applied to traffic flow data.

The results show that a visualization process is iterative and similar to data mining processes. Visualizations have specifiable features that define useful visualizations. Moreover, there are many techniques for visualizations to reduce cognitive burden of a data analysis task. The created prototypes demonstrate the value of 3D in visualizing complex multi-dimensional datasets. They also demonstrate the numerous challenges data management poses on visualization development.

Tiivistelmä

Lappeenrannan teknillinen yliopisto
School of Business and Management
Tietotekniikan koulutusohjelma

Karri Väänänen

Visualisaatiot työkaluina syvän ymmärryksen etsimiseen tietomassasta

Diplomityö. 2017. 89 sivua, 24 kuvaa, 1 taulukko ja 3 liitettä.

Tarkastajat: Jari Porras, Professori
Ari Happonen, TkT

Hakusanat: Datan visualisointi, syvä ymmärrys, tutkiva data-analyysi, verkkoperäinen visualisointi, Cesium-framework, liikenne, GIS, aikaspatiaalinen data, tiedonlouhinta

Keywords: Data visualization, insight, exploratory data analysis, web-based visualization, Cesium framework, traffic, GIS, spatio-temporal data, data mining

Tässä diplomityössä tutkitaan data- ja informaatiovisualisaatioita tapoina syvän ymmärryksen luomiseen tietomassasta. Avointa dataa tie- ja joukkoliikenteestä käytetään spatiaalisten, ajallisten ja verkkoperäisten visualisaatioiden kehittämisessä. Työ pyrkii myös tutkimaan ihmisen ymmärryksen roolia visualisaatioissa ja tutkivan data-analyysin prosesseja.

Informaatiovisualisaatiota, data-analyysin ja visualisoinnin prosesseja sekä ihmisen ymmärrystä visualisaatioiden yhteydessä tutkitaan kirjallisuudesta. Visualisaatioiden kehittämisen ja data-analyysiprosessien yhteyksiä vertaillaan. Lisäksi verkkoperäisiä 2D ja 3D visualisaatioita kehitetään osana iteratiivista prosessia. Visualisaatioiden tiedonhallinnointia käsitellään, kuten datan löytämistä, keräämistä ja esiprosessointia. Myös ennakoivan koneoppimisen keinoja sovelletaan lyhyesti liikennevirtadataan.

Tulokset osoittavat, että visualisaatioprosessi on iteratiivinen ja samankaltainen data-analyysiprosesseihin verrattuna. Visualisaatioilla on eriteltäviä ominaisuuksia, jotka määrittävät hyödyllisen visualisaation. Lisäksi on monia visualisoinnin tekniikoita, joilla voidaan vähentää data-analyysityön ymmärryksellistä taakkaa. Luodut prototyypit osoittavat 3D visualisoinnin arvon kuvannettaessa monimutkaisia moniulotteisia joukkoja tietoa. Ne osoittavat myös millaisia haasteita tiedonhallinnointi asettaa visualisaatioiden kehittämiselle.

Acknowledgements

I would like to thank my supervisor and instructor Ari Happonen for providing such an interesting topic to work on over the summer and working tirelessly to allow this piece of work to happen. I would also like to thank the associated people on the project from the case company for taking the time and sharing valuable practical views on topic. Also, I want to thank my friends and family for showing interest and support for my work.

Lappeenranta, Feb. 12th 2017

Karri Väänänen

Table of contents

Table of contents **5**

List of symbols and abbreviations **8**

1 Introduction **9**

1.1 Background 10

1.2 Research questions 11

1.3 Scope and goals 11

1.4 Structure of the thesis 13

2 Related work **15**

2.1 Information visualization 15

2.2 Processes around data mining and visualizations 17

2.2.1 Rationale for the choice of the reference model 18

2.2.2 Knowledge discovery in Databases 20

2.3 Empirical view on iterative visualization creation process 21

2.4 Exploratory data analysis 24

2.5 Data visualization and cognition 24

2.5.1 Cognitive support, working memory & cognitive load 25

2.5.2 Visualization techniques for better cognitive support 26

2.5.3 Visualizations and grounded theory 28

2.6 Overview of visualization tools from literature 29

3 Data management & dataset creation **32**

3.1 Data discovery & collection 32

3.1.1 Automatic data collection scripts 33

3.1.2 Generating fake data 34

3.2 Data preprocessing 35

3.2.1 Evenly spaced time slot aggregation 35

3.2.2 Use of spatial databases 36

3.2.3 Preprocessing of events data 37

3.2.4 Data reduction 38

3.3 Holiday data 39

3.3.1 Scripts for creating holiday dataset dynamically 40
3.3.2 Visualizing holiday data 40

4 Preliminary traffic data analysis 43
4.1 Data sources and methods 43
4.2 Data overview 44
4.3 Preprocessing 46
4.4 Daily traffic and holiday data 48
4.5 K-nearest neighbor method for traffic forecasting 49

5 Development of visualization artefacts 53
5.1 Spatial travel time to events visualization 53
5.1.1 Data sources 54
5.1.2 Data flows 55
5.1.3 Distance matrix of public transportation travel times 56
5.1.4 QGIS visualization 57
5.1.5 Qgis2threejs plugin - From 2D to 3D 58
5.1.6 Cesium visualization – Usability improvements in 3D 60
5.2 Automatic traffic measurement data 62
5.2.1 Data sources 62
5.2.2 Methods to visualize in 2D 63
5.2.3 Creation of the line geometry 65
5.2.4 Temporal visualization in QGIS 66
5.2.5 LAM data visualization in Cesium 67
5.2.6 CZML generation 70
5.2.7 Cesium visualization data flows 71
5.3 Digitrafficview – A common Cesium implementation for visualizations 73

6 Discussion 75

7 Conclusions 78
7.1 Evaluation of artefacts 79
7.2 Future work 80

References 82

Appendix I. Open data sources.....87

Appendix II. Qgis2threejs terrain 88

Appendix III. Traffic pattern comparison.....89

List of symbols and abbreviations

API	Application Programming Interface
CRS	Coordinate Reference System
EDA	Exploratory Data Analysis
GIS	Geographic Information System
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
InfoVis	Information Visualization
IoT	Internet of Things
ITS	Intelligent Transportation Systems
JSON	JavaScript Object Notation
KDD	Knowledge Discovery in Databases
KNN	K-Nearest Neighbor
ORM	Object-Relational Mapping
RMSE	Root-mean-square Error
SciVis	Scientific Visualization
SDT	Spatial Data Type
UI	User Interface
UX	User Experience
VAST	Visual Analytics Science and Technology
WFS	Web Feature Service
WMS	Web Map Service
XML	Extensible Markup Language

1 Introduction

Data science, a kind of a buzzword for statistics, is a wondrous mixture of handling numerical data, cleaning and processing that data, visualizing, making analyses, and conclusions through various methods and processes. Ultimately, the raw data is refined into information, and knowledge. Data science, data mining, business intelligence, or simply statistics, all of them can be taken to mean the same thing which is, of course, making sense of data.

Over the years, there has been wide interest in data mining and visualization research. Visualizations have become more integral part in data analysis as it is often useful to inspect the data visually throughout the analysis process. Traditionally, visualizations have always functioned in a supporting role for data mining methods. They provide a framework for presenting the findings of an analysis in a more digestible way, and inspecting results visually. Visualizing data can give hints on what to analyze next and it can guide the analysis process by pointing out erroneous conclusions early on. They aid in understanding the results of mining and work as an essential interface for interpretation. In this thesis, however, we see visualizations as tools for generating new insight, new ideas, and raising questions about data. Visualizations can function as an exploratory analysis tool for finding information from data. Especially interactive visualizations can further encourage analysts to spend more time analyzing data, generating hypotheses, and ultimately leading to new discoveries. This well-known research direction is known as exploratory data analysis (EDA), or visual analytics.

Previous studies reviewed in this thesis show that visualizations are used to great extend in hypothesis generation, raising questions, and building up understanding of a problem. Understanding of human cognition plays an important role in assessing visual tools. Cognition and working memory have been studied extensively in the context of visualizations. As we see it, the role of data visualization is to ease the cognitive burden a complex dataset poses and provide a kind of “Aha!” insight into data. In this context, the value of visualizations cannot be overlooked. Visualizations provide understanding that cannot be gained effectively otherwise. Exploratory visualizations answer to the

challenge of obtaining the initial knowledge and questions of a problem on which traditional data analysis methods are based.

1.1 Background

Today, the growing demand for data analysts is well-recognized by both educational organizations and the private sector. Numerous companies from different fields are looking for more data analysis capable engineers. Whether the position is advertised as big data analyst, data science engineer, data analyst, or equivalent, it is clear that there is work to be done. However, sometimes it is not very clear what the job will actually be.

It is no wonder that these jobs emerge since the amount of data available has skyrocketed in the last couple of decades because of highly instrumented, sensor-rich and interconnected world we live in today (IBM et al., 2011). The rise of technologies such as Internet of Things (IoT), and on the other hand, the increasing openness of data have led to the rapid increase in the volume of available data. In addition to proprietary data companies produce and analyze internally, the ever-increasing amount of new open-data shifts the playground[1]. The companies that are able to make efficient use of this data can quite possibly gain competitive edge over their rivals. Both private companies and public organizations have much to gain from the situation presented here.

The prelude for exploratory data analysis is not only about collecting as much data as possible, and planning to make use of it. The other important part in addition to collecting the data is hypothesis generation. An analyst needs to understand to ask the right questions concerning both the data and the use case. A company could have a considerable amount of data available but they may not know what to do with it. This means they are not able to ask the right questions. The analysis of data, i.e. answering the questions, can only be made after the questions are known. In other words; knowledge can be attained after we know what we want to know. Generating these questions, hypotheses, requires knowledge

[1] Available private sector open-data jobs are forecasted to increase by 32 per cent in 5 years (2016-2020) spawning nearly 25 000 new jobs by 2020 in EU region alone (European Data Portal, 2015).

of both the data available and the business domain. Later in this thesis we explore visual ways to generate new hypotheses from data.

The increasing openness of data, possibilities of utilizing that data, and the fact that companies are recognizing the situation are the main factors leading to the work done in this thesis.

1.2 Research questions

The research questions in this thesis are split into two different viewpoints. The main questions are either generic or case related. In general, we seek to answer how visualizations are useful in terms of cognitive effort and hypothesis generation. On the other hand, our interests lie in answering the questions spawned by the cases included in this study. To demonstrate the visual analysis process, we create visualization prototypes for these cases. The research questions are as follows:

- Why visualizations make cognitive tasks easier?
- How development of visualization tools relates to knowledge discovery?
- Case 1: How visualize the accessibility of public events from transportation hubs as a part of a travel chain?
- Case 2: How to gain insights into road traffic patterns through visual means?

The primary research method used is artefact creation. Visualization prototypes or artefacts are developed to understand the visualization needs for both two cases. The visualizations are developed iteratively and evaluated against the best practices found in the literature. To build these visualizations, the related literature is studied and applied to the scenario presented here. The related work chapter aims to answer how a useful visualization is composed, what makes it effective, and how a visualization creation process is defined.

1.3 Scope and goals

The scope of this thesis revolves around data and information visualization. The divide between all the different terms trying to categorize visualizations are somewhat vague

and overlapping. Thus, it is difficult to come up with explicit boundaries that perhaps exclude some types of visualizations and includes others. For the purposes of determining boundaries we explore mainly information visualization from the literature as it is the field that has the most research conducted in the last couple of decades. Data visualization can be seen as a subfield of information visualization (Friendly, 2008a). Nonetheless, we do want to emphasize the difference between the terms data, information, and knowledge as each of them have a purpose in scientific text.

The purpose of this study is to understand the value of data visualization and what kind of challenges it poses. This is studied empirically by creating visualizations of large datasets. The aim of the visualizations is to generate new insight. We present visualizations and the development process of these visualization artefacts. We seek to generate hypotheses though visualizations instead of conducting extensive hypothesis testing activities. This means that the focus of this study is on exploratory visualizations and not in the traditional data mining type of analysis work. Also, computer visualizations being naturally explorative and highly creative – this thesis leans towards qualitative research activities. Visualizations are tied to exploratory data analysis and grounded theory activities. For example, geographic information system (GIS) visualizations have been previously used in support for grounded theory research (Knigge and Cope, 2006). However, the qualitative approach being dominant, a part of this thesis examines datasets using traditional analysis through regression and machine learning.

The research methods used in this study are literature study, artefact creation, and artefact evaluation based on the applicable literature. Visualization artefact creation implies data collection if the data is not readily available. Thus, a large part of the actual work before the visualization as an output is the collection and management of data. From the literature, the processes of information visualization and data mining are discussed briefly. The process is an important part in any activity and knowing the process steps helps at understanding the work we do and quite possibly improve the work. Moreover, human cognition in the context of visualization is studied.

In addition to studying theory, an iterative process for creating various visualizations artefacts is used in this work. Visualizations are created using well-established tools as well as programming frameworks allowing quick development of new features.

Programming frameworks speed up the development of features while allowing a high customizability for compromise-free visualizations that cannot be produced with high-level tools. Visualizations in this thesis concentrate on spatial and time domains. GIS visualizations are used extensively and web-based spatio-temporal visualization prototypes are created.

The creation of data visualization tools require data. A multitude of datasets are used in this work. Most of the data do belong to the onslaught of new open-data becoming available today. We utilized automatic traffic monitoring data available from a public API (Application Programming Interface) which was used in spatio-temporal visualizations. The other line of work, travel time and event visualizations, used public spatial event data together with travel time information from a journey planning service.

In addition to the main data sources, public weather data was collected from the Finnish Meteorological Institute, emergency data from the Department of Rescue Services, a small dataset of public transportation data from the city of Imatra, and a Python implementation was created for generating holiday data for different regions in Finland.

1.4 Structure of the thesis

In this chapter, we presented the background for the thesis, our motivation, scope, and goals of the thesis. In the next chapter, we describe information visualization, data mining and visualization processes, and human cognition in the context of visualizations. Information visualization and its subfields are explored briefly. We describe a few processes around visualization and data analysis work. They help to understand what steps are necessary when creating visualizations. Human cognition and perception are important topics in understanding what makes visualizations effective for gaining insight into data. We also explore a few studies on various visualization tools.

In the third chapter, data management is studied. We mention ways used to discover, collect, and preprocess various datasets used in this thesis. The preprocessing methods common for all data mining and visualization steps in the latter chapters are presented here. Generation of a supporting holiday dataset is also specified.

The fourth chapter illustrates an analysis and forecasting of traffic flow data. The dataset is explored visually and by using K-nearest neighbor (KNN) machine learning method. We visualize preprocessed data, aggregate new information, and compare our traffic flow prediction results with applicable research.

The fifth chapter portrays our visualization artefacts created for this thesis. The visualizations are done using QGIS software and Cesium framework. The chapter shows visualizations for two different cases, each of which have distinct characteristics when it comes to data preprocessing and visualization techniques.

The last two chapters are the discussion and conclusion where we evaluate our findings and discuss the future work.

2 Related work

This chapter aims to explain the value of visualization and to explore how creating visualizations is tied to data mining. Information visualization is defined and discussed while mirroring it to certain data mining processes. This approach sets the basis for visualization artefact creation done in the later chapters by offering us with a practical view on how to develop visualizations. Also, this chapter presents some ways to evaluate these computer visualizations for their usefulness and whether they are informative. This chapter will, thus, show which kind of various qualities of visualizations may hinder or improve their capability to offer information.

The collection of the sources discussed in this chapter were motivated by three different purposes. The first one is to explain, as well as, compare information and data visualization, data mining processes, and exploratory data analysis. The studies were gathered by searching publications online by using related keywords. The second purpose is to understand human cognition in the context of computer visualizations. A need for explaining how visualizations can be effective was raised shortly after the first round of studies were collected and the first visualization prototypes were created. The third reason for collecting sources was to gather a sample of computer visualization artefacts created earlier and detailed in an academic paper. This sample gives an overview of visualization prototypes that can be considered somehow useful, and thus, applicable to our work.

2.1 Information visualization

Visualization can be defined as *"the representation of an object, situation, or set of information as a chart or other image."* (Oxford Dictionaries, n.d.)

In order to put things in perspective it is often useful to look back in history. Information visualization today is not just a recent development branching out of computer science or statistics. The history of visualizations extends far to the times of Ancient Egypt. The earliest visualizations were maps depicting towns and diagrams of celestial objects. The first known examples of coordinate data used in creating maps; a sort of latitude and longitude, were from Ancient Egypt circa 200 B.C. or earlier. In the following centuries, the advancements in visualization were mostly about developing accurate measurement

methods and collecting accurate quantitative data. In the 18th century new graphic forms emerged such as thematic maps and timelines. Also, many modern charts; line graphs and bar charts were invented by William Playfair in this time. During the 19th century the use of graphics spread to public institutions – becoming a standard reporting tool. This was followed by the so-called Golden Age of statistical graphics later in the century. From the 1970s till today data visualization developed into a diverse research field. Currently, we have mature software tools producing various kinds of static or interactive visualizations in addition to flourishing research activity. (Friendly, 2008b)

Traditionally visualizations have been categorized into two different categories: information visualization (InfoVis) and scientific visualization (SciVis) (Tory and Moller, 2004). Information visualization includes the representation of abstract data or intangible information, and scientific visualizations are often about physical, real-world representations. However, this taxonomy is rather ambiguous and other terminology have been suggested as well. (Tory and Moller, 2004) In addition to this rough categorization a third category has emerged; visual analytics. Visual analytics is the spawn of the current *data deluge*[2]. Massive amounts of data available today, big data, can easily overwhelm analysts and make any kind of analysis from that data a very complex activity. (Kielman et al., 2009) Visual analytics is primarily enabled by visualizations and as a research field heavily leans towards the processes and methods of analysis making (Wong and Thomas, 2004). In Institute of Electrical and Electronics Engineers (IEEE) conferences visual analytics is known as Visual Analytics Science and Technology (VAST).

Information visualizations can be further categorized by data type. In a taxonomy presented by Shneiderman (Shneiderman, 1996) information visualizations are divided into seven different data types: 1D, 2D, 3D, temporal, tree, network, and multi-dimensional data. For each data type there exists several kinds of suitable visualizations (Zoss, 2015), and it is clear that the data type guides the choice of visualization approach. The taxonomy also recognizes seven task types common for information visualizations. These are overview, zoom, filter, details-on-demand, relate, history, and extracts. The

[2] Data deluge, also known as data flood is closely related to *information explosion* which is the rapid increase in the amount of information available (Oxford Dictionaries, n.d.).

first four are perhaps self-explanatory but the rest are vaguer. *Relate* task is the possibility to show relationships between items in the visualization. *History* task instructs to save the history of actions for undo functionality and progressive refinement, for example in Data-driven documents (D3) this could mean to implement the back and forward browser functionality. *Extract* task translates to the feature in which you can download the current data in display to another format for sharing or further data analysis tasks. The data types presented here can be stored in many different formats such as comma-separated values, relational databases, or shapefiles. In general, relational databases are considered as multi-dimensional datasets by Shneiderman and Keim (Keim, 2000).

Multi-dimensional visualizations can be very complex even to the point that the visualization is not understandable to the reader without explicit tutorial. A study (Keim, 2000) classifies different multi-dimensional visualizations based on three criteria: visualization technique, distortion technique, and interaction technique. Keim's interaction techniques are like the Shneiderman's data tasks. Visualization technique, however, is divided into graph, hierarchical, pixel-oriented, icon-based, and geometric types. Distortion technique considers the level-of-detail among data where some parts of data can be shown in higher detail than others. It is very similar to zoom task but can be coded into the visualization, function otherwise automatically, or differently altogether. Thus, distortion technique refers to a more advanced type of method than simple user interaction with the visualization. These taxonomies presented by Keim and Shneiderman show that the categorization of visualizations can be made differently depending on the study. However, it can be inferred that visualizations have at least three distinct qualities: visualization technique which is strongly based on the data type (for example; hierarchical data can be difficult to visualize using other than hierarchical visualization techniques), interaction, and more advanced features such as distortion.

2.2 Processes around data mining and visualizations

In this section, we review processes regarding data mining and the development of exploratory visualizations. The processes have much in common but are not alike. A reference process that can explain the visual analytics and insight generation through

visualizations, while considering the required data management, is selected. We concentrate on a knowledge discovery method and tie it with visualization making.

2.2.1 Rationale for the choice of the reference model

Routasuo in her thesis (Routasuo, 2013) discusses and compares ten different models from visualizations, data mining, and exploratory analysis. The thesis argues that the models from related fields, namely data mining and exploratory analysis are especially useful for visualization creation tasks due to similar goals and overall process structure. Our empirical experiences from visualization artefact creation do support this view. From our experience, especially the process steps handling the data, namely data collection, cleaning, and pre-processing are shared between a visualization process and a data analysis process. A very optimistic view would be that they differ only by the method used for analysis. Traditionally, data analysis is conducted using quantitative data mining methods but in visual analytics the analysis is left for human cognition and our ability to perceive patterns. This means that the evaluation of gained understanding may not be as easily defined as in traditional data analysis.

How data mining processes are described and on what level they are presented vary from study to study. A type of generic model by Myatt and Johnson (Myatt and Johnson, 2009) describes four high level steps.

Definition ➔ Preparation ➔ Analysis ➔ Deployment

This model, however, is very high level and its direct application to data analysis work can be difficult. Another very similar but slightly more describing process is an iterative data analysis process as outlined in an exploratory data analysis book (Andrienko and Andrienko, 2006):

1. formulate questions
2. choose analysis methods
3. prepare the data for application of the methods
4. apply the methods to the data
5. interpret and evaluate the results obtained

The process above stems from having specific questions from data we seek to answer. This is the traditional way of conducting statistics; we have a hypothesis (question) that we want to test. However, when no such hypothesis exists, or the questions we seek answers for are vague, the process is often called exploratory data analysis[3]. EDA is a form of hypothesis generation method which instead of being clearly defined is more of a philosophical approach or a set of guidelines for the job. (Andrienko and Andrienko, 2006)

The difficulty of defining an unambiguous model for exploratory analysis or visualization is apparent from the aforementioned sources. Although some models exist, none of them is willing to give a generic all-around model, such as in data mining, specifically for visualization tasks. This does suggest that exploratory analysis or visualizations have more of art-form-like properties than them being strictly guided processes.

Moreover, some well-known guidelines from visualization research, such as Shneiderman's Information Seeking Mantra (Shneiderman, 1996), are not very useful in this context. The mantra is a good baseline for visualizations and is summarized as: *"Overview first, zoom and filter, then details-on-demand".* While it is an exceptionally good guideline, the mantra itself is just a kind of a functional requirement for the visualization output. It does not guide the process or tell how to create a practical visualization – only how a practical visualization should function.

The thesis by Routasuo also recognizes several other models in data mining. One of them is a knowledge discovery process such as the Knowledge Discovery in Databases (KDD). While it is the most accurate and detailed process of the reviewed models – KDD is selected to be as a sort of a reference model for this thesis work. KDD is an industry standard and a battle-tested process backed by blossoming research. Although, Routasuo in her thesis interpreted knowledge discovery as a consecutive process, KDD is in fact described as an iterative process in its original research (Brachman and Anand, 1994; Fayyad et al., 1996).

[3] EDA was originated by John W. Tukey in his book Exploratory Data Analysis published in 1977. (Andrienko and Andrienko, 2006)

2.2.2 Knowledge discovery in Databases

In a study Fayaad et al. (Fayyad et al., 1996) describe KDD as *"the overall process of discovering useful knowledge from data"*. They do not regard it only as data mining which is a distinct step in the process. Data mining in this context refers to the application of algorithms such as the ones used in pattern recognition, machine learning, or statistics. Thus, data mining step can be understood to cover many different subfields, and we see that visual analysis could be one of them.

Simple visualization such as scatter plots, histograms, and line plots are seen as a crucial element in KDD. Not only are they a nice output for the final report but they are needed throughout the process. (Brachman and Anand, 1994) As Anscombe's quartet[4] demonstrates, graphs are invaluable tool for creating understanding, and in preventing incorrect assumptions about data and calculations. (Dayal, 2015) As such, graph visualizations are a key part in KDD process. Moreover, visualizations can be the only appropriate analysis tool we need for a task in order to confirm some hypothesis (Brachman and Anand, 1994). Like so, visualizations as data mining methods fit well in the KDD process data mining step.

Fayaad et al. in their paper (Fayyad et al., 1996), describe the nine steps in KDD process. In summary, the steps are:

1. Develop understanding of the application domain, establish the goal of the KDD process
2. Creation of the target dataset
3. Cleaning and preprocessing such as noise removal or handling of missing data fields
4. Data reduction and projection, dimensionality reduction
5. Matching the goals to a specific data mining method
6. Exploratory analysis and hypothesis selection

[4] Anscombe's quartet is a set of 4 datasets which all produce nearly the same statistical summaries such as mean, variance, linear regression, etc. but they result in wildly different graphs when plotted. (Anscombe, 1973)

7. Data mining (application of data mining methods)
8. Interpreting patterns
9. Act on discovered knowledge

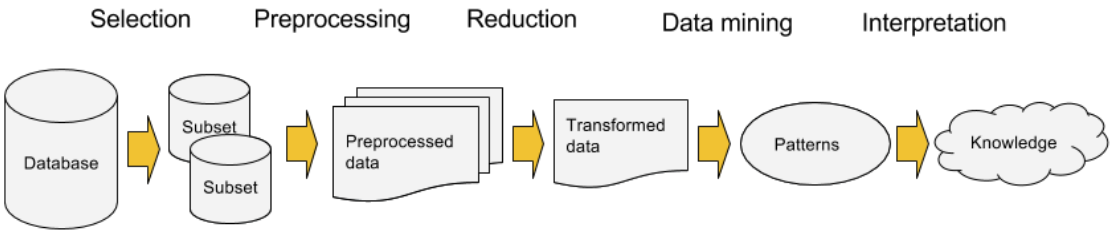


Figure 1. The KDD process (Fayyad et al., 1996)

In Figure 1, the KDD process is pictured in a data-centric way. The process starts with a database and continues with the selection of an interesting subset of data, preprocessing that data, and applying data mining methods to that data. The interpretation and evaluation of patterns, the results of data mining step, is the step that produces new knowledge. It is important to note that this process is both iterative and interactive. The knowledge discovery cannot be automated in traditional sense and it requires interaction by the analyst. This interaction, sometimes, requires the analyst to return to previous steps, hence making the process iterative. (Cios et al., 2007; Fayyad et al., 1996) As the analysis process matures, new questions may come up and new insights are gained. This leads to going back to previous steps and redoing them in the light of the new knowledge. It should be mentioned that the steps in the process may not always be conducted in this order (Han et al., 2011).

2.3 Empirical view on iterative visualization creation process

In this section, we mirror the previous processes from literature to our empirical view and how the process went ahead in practice. It is important to clarify the two completely different viewpoints on this. For one, we have the visualization process that stems from the KDD process. This is more of an internal process that guides the visualization artefact making. Another viewpoint is the software development process which involves a client. After all, the visualizations are created as kinds of prototypes for a case company.

Although, there are certain requirements for the visualization product, they are rather loose.

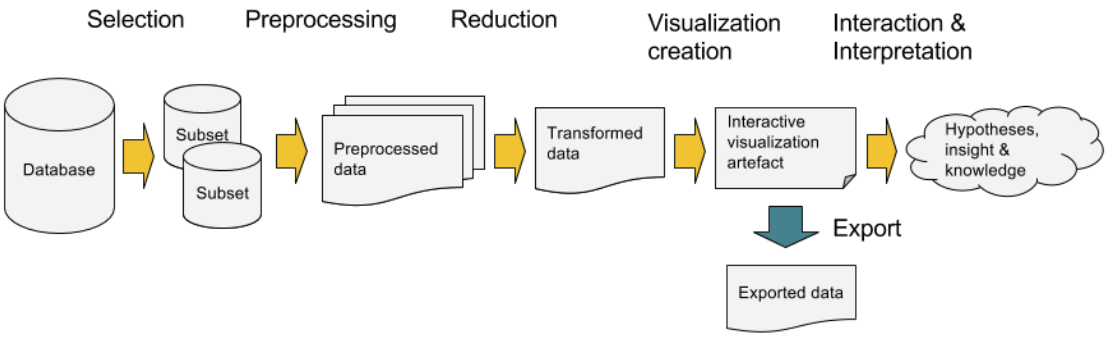


Figure 2. Visualization artefact creation in the context of KDD process

In our experience the visualization process is very similar to the processes found in data mining. Nonetheless, as argued in the previous sections, visualization creation as an analysis tool can be directly applied to KDD process. In Figure 2, the data mining and interpretation are replaced with visualization creation and interaction respectively. This conveys the view that visualization can function as an analysis method. Instead of interpretation of patterns produced by data mining algorithms, we employ a sort of visual analysis for finding patterns through interactivity and interpretation of those findings. This step uses the developed interactive visualization artefact. The output from the visual analysis are hypotheses, new insight into data, or even new knowledge.

The aforementioned visualization process based on KDD is very similar to the exploratory visualization process in the thesis work by Routasuo (Routasuo, 2013). In the study they constructed the process through synthesis of several data mining and visualization process studies. However, there are many differences. In their work they have added evaluation and presentation steps after the interpretation step. Although the evaluation of new findings is seen as an important step, they agree that it is difficult to conduct in practise. The study notes that exploratory visualizations are especially difficult to evaluate. Although to some extent, evaluation of the visualization happens implicitly while interacting with it – the user may note what works and what does not. However, it hardly means that the evaluation happens objectively. Another difference is the missing data reduction step. In KDD the data reduction step is separated from the preprocessing step. We see that the reduction step has its place also in visualization-centric process

because the plain preprocessed data can be too numerous and high-dimensional for actual use in web-based visualizations for example. In the web implementations, specifically, the time to transfer the dataset over the network has to be taken into account.

From the software development point of view, we identify an incentive for iterative development. The process for creating a useful visualization depended on several development cycles or rounds. Based on our study the process started with discovering different possibilities for visualizations. Basically, this means presenting what others have done in the past, and to what end. Exploring different tools and frameworks for visualization was important as the tools dictate what can ultimately be done. Knowing the possibilities of visualization spawned ideas on what data to use, and what patterns to look for in the data. Presenting different visualizations in the first round led to a kind of "eureka moment": combine transportation hub data with multitudes of events data and visualize the travel distances. Besides presenting third party work, having a working visualization prototype definitely helps with both selling the idea and seeing its weak points. After all, visualization can often only be understood through a visual example. Describing a visualization verbally can be an enormous challenge so a picture is truly worth a thousand words.

In the second round the created visualization spawned more ideas on improving the visualization, and it revealed new insight about the data. Improvements such as data export features came up. This is interesting since the taxonomy presented by Shneiderman (Shneiderman, 1996) suggests data export to be a vital feature in information visualizations. It does give strong support for both what do users expect from visualizations and that the primary features of different kinds of exploratory visualizations are indeed common. Ultimately, in our experience the process leading to useful visualizations is iterative. The requirement for collaborative and iterative development is also recognized in literature (Grainger et al., 2016). Moreover, the order of the steps in the visualization process is not set in stone. Sometimes, new requirements for the visualizations spawn the need for formatting data differently. Also, preprocessing step may happen at various points even before data selection if the dataset is not too large.

2.4 Exploratory data analysis

Traditionally, in literature, visual analysis and data mining do not mix. Often books discussed about one of them but disregarded the other. In traditional data analysis, the approach for the analysis begins with generating hypotheses, collecting data, and using data mining methods to support the hypotheses. In exploratory data analysis the data is gathered optimistically first-hand before knowing the problem or possible hypotheses. (Shneiderman, 2002) The latter approach fits well in the Big Data setting, as we have data but the data is not generated specifically to answer a certain question.

Graphical representations and EDA go together. EDA is often conducted using interactive visualizations. Interactivity means that the users are in control and they can specify what they are seeking. These sorts of discovery tools should also allow sharing of findings, and they should concentrate on the user experience. Overly complex tools will hinder gaining the understanding the tools are supposed to provide. Innovative tools that combine data mining and information visualization can offer deeper understanding of data. (Shneiderman, 2002) Visualizations are the backbone of visual analytics. Visual analytics aims to deliver new knowledge through the support of our cognitive abilities. The high-level tasks in visual analysis are searching for data, exploring the found data, and analyzing data for patterns. (Komenda and Schwarz, 2013)

2.5 Data visualization and cognition

The very nature of visualization is interesting since it relies heavily on our visual system, working memory capacity, and cognitive abilities (Lohse, 1997; Nielsen, 2016; Tory and Moller, 2004). However, these are highly complex fields of study and involve research from various disciplines such as neurophysiology, or psychology. Thus, it is very hard to give an exhaustive explanation on human cognition and visual system when it comes to visualization techniques in a computer science related topic.

A common term related to visualizations and cognition is insight. Insight is the main purpose of visualizations. Visualization produce pretty pictures and possibly interactivity but the end-goal is in gaining insight. Insight is new knowledge, a unit of discovery, or seeing something in a new light. Insight is very complex to define as it has many levels,

it is unexpected, and qualitative. New insight accumulates and builds on itself but it is also subjective. (Yi et al., 2008)

Yet another interesting term coupled with insight creation is *sensemaking*. Sensemaking is a loosely described iterative process where insight is both an end-product and an intermediate part of the process. The sensemaking process, with its insight discovery, creation, and retrospective characteristics, is better described by Yi et al. in their paper. However, what is important is that insight is not just an end-product of a visualization but it matures over time and shapes itself. (Yi et al., 2008) Visualization tools are just enablers in this context where the domain-knowledge of the analyst plays a major role.

2.5.1 Cognitive support, working memory & cognitive load

Human memory is basically a complex information processing system. Although, very effective at what it does – it has its limitations. Working memory is limited by its capacity and duration which differ from person to person (Huang et al., 2009; Lohse, 1997). In literature, visualization is seen as a provider of cognitive support (Lohse, 1997; Tory and Moller, 2004). This means that a visualization can shift a portion of the cognitive load to visual perception system reducing the cognitive effort required to complete a cognitive task. Lohse, also, mentions that people have separate working memory capacity for verbal, spatial, and visual objects. Based on this, one could argue that combining visual, spatial, and verbal elements in a single visualization would result in a better utilization of our cognitive capacity.

Human memory works on three levels. When processing information, the first level is called *sensory memory*. This is the interface we use to process information from outside. The second level is *short-term memory* or *working memory* which is achieved by paying attention to what we sense. If we disregard the information we sense and do not put effort into cognition, we may not be using our working memory capability. The third level is *long-term memory* which is for storing and retrieving information for later use. The limited working memory presents us a bottleneck for which visualizations offer remedy. (Huang et al., 2009)

A study (Lohse, 1997) states: *" – colour graphs with grid lines reduce cognitive overhead by shifting some of the information acquisition burden to our visual perception system freeing cognitive resources for other steps in the problem-solving task."* However, this in only true in the case where the information cannot be adequately handled by the working memory capacity. The study presents a case where graph visualization had no improving effect on a simple enough case where the data could be handled by the person's working memory. Thus, the study states that visualizations are effective in cases where the amount of cognitive effort increases beyond the capacity of working memory used for the task.

Visualizations along science texts can enhance deep learning and comprehension (Schwamborn et al., 2011). The study by Schwamborn shows that images help understanding by reducing cognitive effort. However, the study finds that the used computer tool may have an ill effect on learning time. A computer-based visualization tool may have such a steep learning curve that it increases the cognitive effort rather than reducing it. So, at first, the learning effort goes to learning the tool. Tutorials and guides about the tools should be created to ease the learning process. Moreover, it is essential to consider how long it takes to learn to use the visualization tool if the tool is only to be used briefly.

It is important to consider the cognitive effort when evaluating visualizations. Visualization can be evaluated traditionally by metrics such as response time and accuracy. These, however, do not take into account the person's cognitive effort they used at the time since the same results can be obtained by using different tools which require different levels of cognitive effort. (Huang et al., 2009)

2.5.2 Visualization techniques for better cognitive support

Studies (Huang et al., 2009; Lohse, 1997; Robertson et al., 2008) agree that visualization cannot automatically improve cognitive tasks but it depends on both the visualization and human factors. The literature presents us several techniques how to create effective visualizations, reduce cognitive effort, and gain new insight.

A study (Yi et al., 2008), based on extensive literature review, presents four different methods for insight generation: *provide overview*, *adjust*, *detect pattern*, and *match*

mental model. Provide overview is important in insight generation indirectly. It will not help gain insight by itself but rather promote areas where more insight could be gained from. Adjusting is the process where changing perspective and selecting elements from the dataset is possible. It allows filtering and grouping of data. Pattern detection in this context is the discovery of trends and other underlying relationships in the data. Matching of mental model means that the visualization should mirror the view we have on the world we live in. For example, as we perceive the world spatially, the data should be plotted in the same manner, in 3D or geographically. Matching of mental model certainly helps with understanding the purpose of the visualization. In contrast, abstract pixel-oriented visualizations (Keim, 2000) can be very confusing without explicit tutorial as they do not fit into any mental model most people are familiar (Figure 5). In these types of visualization data could be encoded as images but the underlying data does not originally come from sources that can be expressed as images in traditional sense. The four methods for insight generation are very similar to Shneiderman's Visual Information Seeking Mantra (Shneiderman, 1996). The mantra clearly includes the overview and the adjust steps.

Color is an important part of our perception system. In the study by Lohse (Lohse, 1997) colored graphs with grid lines led to better efficiency than mono-color graphs without grid lines. In the study, it is described that adding color and helper lines to graph visualization allowed human visual system to process information in parallel leading to improved information acquisition speed.

The literature suggests that visualizations, although being effective, cannot describe all the necessary information in one view. Hence, it is important to have several coordinated views which show the same dataset from different perspectives. These are often called *linked views*. Linked views can be especially useful for exploratory tasks of complex datasets, particularly spatio-temporal datasets. (Shimabukuro et al., 2004) Another effective visualization technique to reduce cognitive effort are adaptive visualizations. Adaptive visualizations are especially useful for exploratory tasks and they aim to provide personalized visuals. In adaptive visualizations data is represented by using semantic information to tie the different datasets. They are also able to adapt to user behavior and

different data characteristics. The datasets for adaptive visualizations can be linked using semantic web practices such as *linked data*[5]. (Nazemi et al., 2014)

Continuing on the adaptability aspect, modifying displayed data to more understandable format based on the use case was seen useful in a study by Kaiser (Kaiser et al., 2010). The study proposed an algorithm for time-distance transformations. For example, a map of a metro network could be adjusted to visualize the travel times rather than physical distances between stations. A traveler is usually more interested on the time it takes to get to the destination than the travel distance. Hence, visualizing the travel times can be extremely useful in this context. Yet, how it agrees with the *match mental model* principle is under question. Is it obviously clear to a common traveler that the map is adjusted in such a way? Does it match the mental model of an average metro user?

A somewhat controversial method in information visualization is animation. A paper by Robertson et al (Robertson et al., 2008) studied animation in multi-dimensional trend visualizations. Animation can be successful for visualizations meant for presentation purposes. However, static visualizations were seen to work better for analysis tasks and creating understanding of data. For analysis, movement can be confusing and it might require replaying the animation several times in order to make analytical observations. The study finds that making analysis from animation can be error-prone. Visualization methods such as traces and small-multiples worked the best for making analysis. Traces method is the static depiction of movement by drawing the trace of movement. Small-multiples means the method where there are several small pictures side by side, allowing comparisons to be made. Ultimately, what works for presentation of data does not necessarily work for analysis, the study finds.

2.5.3 Visualizations and grounded theory

Exploratory visualizations stand in an interesting middle-ground between qualitative and quantitative. The underlying data, for example in the case 2, are spatio-temporal facts measured in real-time from road traffic stations. This quantitative data is visualized for

[5] Linked data, in the context of semantic web, refers to the recommended practices for publishing structured content on the web (World Wide Web Consortium, 2016).

insight generation which, in turn, produces qualitative information. After all, insight is defined as qualitative. It is not exact nor certain. Therefore, insight gained from these visualizations can be highly subjective. (Yi et al., 2008)

In literature grounded theory has been used in conjunction with GIS visualizations. Grounded theory is used extensively in social sciences and can be summarized as *"the purpose of grounded theory is to build theories from data about the social world such that theories are 'grounded' in people's everyday experiences and actions."* (Knigge and Cope, 2006)

The study by Knigge and Cope combined qualitative information, such as interviews and field notes, with quantitative data representation in GIS software. The study sees many commonalities between grounded theory and exploratory visualizations. They both are exploratory, iterative, and recursive methods. They also share the possibility for having multiple perspectives, subjectivity, on data. Also, the study suggests that the temporal element can be very valuable for grounded theory approaches, since the understanding of process of change over time is essential for conducting grounded theory research.

The added value of this combination of different types of data is that it ties content to context. Knigge and Cope presents an example where seeing ethnographic data in geographic context allows gaining insight that would be hard to obtain otherwise. One interpretation could be made by looking at the geographic data alone. However, another interpretation could be made by considering qualitative information as well, such as interviews from the people living in that area. These different data sources can complement each other or have some contradictions that need to be resolved. The added explicit context could make analysis easier for an analyst who is not very familiar with the problem domain.

2.6 Overview of visualization tools from literature

Our literature study of visualizations yielded various studies (Table 1) where an existing tool was either described or a prototype visualization was created. We identify the type of tool created, the type of visualization along with its dimensions, and list the viewpoint of the study. We see that it extremely useful to explore what has been done before, and

on the other hand, why certain approaches in visualizations had been taken. What tools and frameworks are being used? As this thesis focuses on spatial and temporal domains, the tools gathered in this section reflect that view.

In the Table 1 we see that most of the visualization tools gathered here are web-based. The reasoning for the selection of a particular approach is not always evident from the collected studies. However, web-based tools are seen as enablers of rapid development. Web visualizations can also be composed from interchangeable and scalable modules. (Jones et al., 2016) The possibility of using web APIs is also recognized. The fact that web services are interconnected, they can share data in machine-readable format, and that they are platform independent makes them advantageous. (He et al., 2010) Web development tools are also numerous in open-source frameworks and libraries.

In addition to providing background to tool selection, these studies present interesting examples of working visualizations. For example, an article (Pack, 2010) presents several kinds of visualization methods. The article visualizes traffic data in spatial 2D visualizations with additional linked views, 3D visualizations, heat maps, and types of advanced graph diagrams. The study presents interactive spiral graphs for cyclic temporal data such as daily traffic patterns. Moreover, the article encourages to include specific visualization features, such as data exports, that are often referred in the literature.

Table 1. A few visualization tools gathered from literature

Tool	Type of artifact	Type of visualization	Paper viewpoint
GeoTime (Kapler and Wright, 2005)	Research prototype	Spatio-temporal	Demonstrate prototype
Gephi (Bastian et al., 2009)	Open source desktop application	Graph and network analysis	Tool description
Several tools (Sedlmair et al., 2011)	Proprietary/Unknown	Various	Design, development, and evaluation
(Sayar et al., 2006)	Design framework for web service	Geospatial	Integrating WMS/WFS services using AJAX
ManyEyes (Viegas et al., 2007)	Web-based	Various	Tool description

Incident Cluster Explorer (ICE) (Pack, 2010)	Web-based	Relationships, spatial, temporal, 3D, generic	Overview (concerning visualizations in transportation)
(He et al., 2010)	Web-based	Spatio-temporal GIS	Process-oriented spatiotemporal visualization method.
Marine GIS (Goralski and Gold, 2008)	Desktop application / research prototype	Geospatial	Tool description, presenting advancements in developing such a tool.
GeoViz (Zhang et al., 2016)	Web-based	Spatio-temporal	Cloud-enabled visualization
Survey Data Viewer (Jones et al., 2016)	Web-based, D3	Various	Tool description

3 Data management & dataset creation

This chapter describes various methods we used to discover, collect, and preprocess data for our visualization and analysis needs. The chapter focuses on practical aspects and explains the methods applicable to our study. We describe the ways to collect data from public APIs and the management of that data in relational and spatial databases. Later, some of the most important preprocessing steps for that data are discussed.

3.1 Data discovery & collection

One of the very first stages in this thesis work was data discovery. After defining the problem statement, the process for generating ideas for data sources started. Even though having a vague idea for visualization and the types of data we want, finding that data still takes time and work. Exploring possible open-source data sources is a time-consuming task. Countless datasets are being maintained in several different open data portals. These portals can help in finding the data but often many datasets are missing or the datasets are not relevant to your work. Thus, it is not enough to just go through these portals but a generic web search must be conducted as well.

Datasets containing data about border traffic, public transportation, road traffic, road emergencies, holidays, event calendar, and weather data were searched. Several datasets were discovered (Appendix I). The found data sources are accessible from various APIs, download services, and view services. The data sources were classified by multiple factors such as implementation and data format. Based on this work we identified a few important questions for classifying data sources:

- How often the data is updated? (Update frequency)
- Is any history data available or is the data real-time only? (Data log)
- Does the data have a spatial dimension? (Location)
- What protocol does the API use?
- In what format is the data served?

- Is access to data source open or restricted?

3.1.1 Automatic data collection scripts

Many of the required datasets for the work in this thesis were behind APIs. Hence, it is not just a simple download of a comma-separated values (CSV) file but requires somewhat more effort if history data is needed. To tackle this problem several data collection scripts were implemented in Python. A programming language that has a good library support for HTTP (Hypertext Transfer Protocol), XML (Extensible Markup Language) parsing, JSON (JavaScript Object Notation) reading, and database management was required. Python has all that and it is high-level so time spend writing code is minimized.

All the open data APIs used in this study worked differently and the data came in various formats. One API responds in XML format while another talks JSON. Moreover, the structure of the response does not follow any standards. For cases like spatial data there are protocols such as WFS (Web Feature Service) or WMS (Web Map Service) which can be read using standard software but generic open data APIs have their protocol formatted more freely. Therefore, the response must be parsed differently in each case whether it was XML, JSON, or something else. Besides parsing, the database tables must be customized for each case as well because the dimensions between the datasets do not match. For spatial data, perhaps the spatial dimension does not change but everything else is case determined.

Having database tables and response parsing in place the rest of the data collection goes similarly. Database connection was implemented using SQLAlchemy library. SQLAlchemy provides database engine agnostic interface for database connections, making switching between engines easier. Moreover, it implements a technique called Object-Relational Mapping (ORM) which helps managing the database tables and it decouples the SQL code from Python code. One of the advantages of this is that if you’re making changes to database tables you do not necessarily have to review the whole Python implementation. Basically, the added abstraction layer of ORM helps when making changes to underlying database implementation.

In addition to ORM, SQLAlchemy provides a toolkit for database commands. Operations such as *merging* simplify the process of adding and updating rows in the database. These sorts of simple methods ultimately add up and make writing efficient code easy. The robustness of the data collection is important since the download script runs automatically without oversight. We utilized *cron* software in UNIX-like computers to implement the periodic data downloads. In this case the only oversight you have over the script is log files and possible error messages mailed to you from cron.

Designing well-thought-out table schemas, and an implementation that honors data integrity helps to manage the data in the long run. After all, a major portion of the data analysis jobs is spent handling the data in the database. Even though, techniques such as ORM help in the application development, in data analysis, writing SQL for creating views and such cannot be avoided.

The collection scripts were created for South Karelian event calendar API, Digitraffic API, Finnish Meteorological Insititute API, and Rescue Services media channel RSS feed. All the services can be reached through HTTP. The data format is different so the logic after getting the response must be written separately for each case. Also, the database tables must be defined for each case.

3.1.2 Generating fake data

After data discovery phase and recognizing the need for certain datasets for visualization creating, the difficulty of attaining the data or the insufficient amount of data available can be a problem. Collecting the required data can take time especially when the data is behind bureaucratic channels. This may take weeks, and that time developing the analysis is hindered without coming up with a way for mimicking the dataset.

In our case, at first, we only had a day's worth of traffic station data. The available data was duplicated for a period of one month. The data generated from this small preview dataset was based on the real day's data by some random factor. In the case of traffic data with daily patterns, a pattern is preserved. While it is "fake" data, it is still based on the real values and mimics the generic daily pattern of traffic data. The generated data is useless for making actual analysis but the development of analysis code can continue until

the real dataset can be used in its place. The use of fake data, thus, sped up the development when the real data was not yet available. Besides generating data using random factors, upper and lower bounds can be useful as well to control data generation bounds.

3.2 Data preprocessing

Massive amounts of collected raw data is still a far cry from having a dataset for data analysis or visualizations purposes. Raw data from real world sources is most likely noisy, it may have missing values, and contain other anomalies. Data preprocessing and cleaning are the methods for correcting these anomalies and producing more coherent data. (Brachman and Anand, 1994; Cios et al., 2007; Fayyad et al., 1996)

In our case the data had missing values left from data collection disruptions, and it was in a format unsuitable for further time domain analysis. In contrast to automatic traffic monitoring data where the missing data was caused by connection interruptions and other technical issues, the events data for case 2 was originally created by people. The events data may have missing spatial information, incorrect dates, and some other missing or erroneous fields.

3.2.1 Evenly spaced time slot aggregation

Data preprocessing for traffic data stemmed from visualization needs where the data had to be modified in a way where it made more sense to do the calculations for the whole dataset instead of calculating it per extracted dataset. Automatic traffic measurement data (LAM data) is recorded from Digitraffic API or get directly from Finnish Transport Agency. Both data sources need the same sort of preprocessing. These data sources are later described in Chapter 5.

One of the first requirements for preprocessing raw LAM data was to aggregate it to evenly spaced time slots. As the dataset was delivered in Microsoft Excel supported format, the first attempt was to use the software in question. By cramping timestamps to 15 minute intervals, and using *Pivot tables* feature to aggregate data under time slots – the spreadsheet approach worked fine for a perfect dataset with no empty time frames.

However, LAM data is bound to have time frames which have no data. Either there have been connection issues leading to missing data or at the time period there were no road traffic measured at the site. Using spreadsheet programs to create all the missing time slots would be too cumbersome. Especially if the data aggregation to time series would have to be made every time there is new dataset available.

The problem was solved by writing a short script that reads an input CSV file, performs the aggregation operation, and writes an output file. The script detects the start and end dates of input data, and generates time slots for that time frame at even intervals. After the time slot creation phase it iterates the data, and through configurable functions either counts or sums the values under each time frame from determined column fields.

Filtering of rows was also implemented. The LAM data had entries categorized by vehicle type. For the border traffic analysis, only passenger traffic was to be considered. Since the aggregation combines data from several input rows the categorization information would be lost without coming up with a crafty way to store that information. By all means, that information could be appended to a field in the output but processing that field would again require more complex implementation later. Thus, it is easier to filter out the unwanted data in the aggregation phase.

The functionality in this script proved to be an essential way to preprocess LAM data. Supplementary to plain CSV processing, the feature had to be accessible programmatically from other Python scripts to preprocess data directly in a database. This required the CSV reading and writing to be decoupled from the actual time slot functions which were provided in a library for other implementations.

3.2.2 Use of spatial databases

A spatial database system is an extension to traditional database systems. A spatial database handles SDTs (Spatial Data Type) such as points, lines, and regions in geometric space. It implements features for efficient spatial operations such as spatial indexing. Spatial database describes objects in space but it also describes the space itself. (Güting, 1994) The space is determined by the used coordinate reference system (CRS). A spatial

database allows any spatial entry in the database to be translated to any other CRS format. Hence, any geometry initially stored in any format can be expressed unambiguously.

In this thesis work LAM data was originally stored into traditional relational database system, namely SQLite. However, accessibility to that data was poor since the use of that data required information not found in the database. This bit of information is, of course, the CRS format. Also, the data had to be exported to comma-separated files for reading. On every export of the data, the use of that data required the identification of CRS format, and subsequently translation of the coordinate information to the format used in the software reading the data. Specifically, it requires a lot of manual work to visualize this data in QGIS, a GIS visualization software. Furthermore, the process of visualizing this data takes increasing amount of effort when the data is in multiple CRS formats.

Spatialite was selected as the spatial database system as it is built on top of SQLite. It means that the SQL language processing is the same besides the added Spatialite functionality. This would make the migration easy. Unfortunately, there is no SQLAlchemy support for Spatialite so more generic tools for accessing the database must be used. In the case of Python this means using the default sqlite3 library with Spatialite extension loaded.

Spatialite was used extensively for storing the spatial metadata of LAM stations, line geometries for visualization, and the preprocessed LAM data. The database can be moved between computers relatively easily because it exists in a single file and that it is separated from the database containing the raw unprocessed LAM data. This helps in using it for data analysis, and sharing. After the LAM data is preprocessed into time slots and made Spatialite compliant, its use in QGIS is very easy. QGIS has Spatialite support built-in and it can read the data from the database efficiently resulting in fast render and analysis times.

3.2.3 Preprocessing of events data

In contrast to LAM data which is collected by machines over the network, events data is inputted by hand by humans. Data anomalies exists in both cases but they have distinct qualities. LAM data may have missing data from certain time frames due to connection

issues. The data, when available, is always consistent. The events data, however, has more variation to it. There may be fields missing, some events may not have coordinates set, some events are recurring, some do not have the time of day set, and so on. There may even be duplicates of the same event with different identifiers in the database. Even though the user interface, where the events are added, does pose control over the data, the data is far from perfect.

The preprocessing of events data in our case is more depended on the use case than the preprocessing of LAM data. Our visual analysis puts more emphasis on the spatial information of the events than, for example, for the temporal aspect of events. The temporal information can be discarded as our analysis is done purely based on the spatial information. Thus, the events with missing spatial fields were discarded as they are of no use to our analysis. The other data cleaning methods was to look for duplicates in the database. The data showed to have duplicate entries with different identifiers, and a lot of duplicate information because of recurring events. From the temporal perspective, these are valuable information but when we are only considering the spatiality of the events, they are redundant. This decreases the extent of the further preprocessing required later.

The filtered dataset is used in creating a distance matrix. This method is described later as part of the visualization creation process in Chapter 5. The creation of the distance matrix has both data preprocessing and data collection types of qualities since the implementation requires fetching of new information from a remote source and joining that to the existing data rows.

In a sense, the preprocessing of events data is alternating between data preprocessing, and data reduction methods. To reduce the amount of preprocessing, a dataset is trimmed. Also, after preprocessing there is further reduction done for the data. This shows that the order of steps in the KDD process is not set in stone and that the process is indeed iterative.

3.2.4 Data reduction

The divide between data preprocessing and data reduction may not be obvious. Some literature regards reduction as a part of preprocessing and some, such as KDD, keep it as

a separate step. What is important to consider is that in data reduction (data transformation, data projection) the goals of the task must be considered. The task affects what dimensions are included, i.e. what database fields are needed, and in what format the data is required by the analysis tools. (Fayyad et al., 1996) The aim of data reduction is to reduce the amount of data while minimizing the loss of information content in the context of the task (Han et al., 2011).

A common format for delivering data for analysis is comma-separated values. CSV format is so common that it is hard to find a data analysis tool that is not able to read it. CSV exported datasets provided a way for us to analyze the data in various tools such as QGIS and MATLAB. In addition to CSV files, a format specifically created for displaying spatio-temporal data on the web was used. In that case, the data was transformed into CZML format for web-based visualizations.

Data transformation, as part of data preprocessing, happens throughout the knowledge discovery process. In the later chapters, the datasets are transformed several times as is required. The previous sections described only the most meaningful preprocessing steps taken.

3.3 Holiday data

When analyzing data over longer periods of time, it is apparent that the data has certain patterns based on the time of day, day of the week, month, or season. Even longer periods of time must be considered in some analysis work. For example, in the cyclical financial markets it may even take decades to distinguish secular market patterns. Obviously, these sorts of long-term analyses are out of the scope of this thesis. The nature of all the above is that they are continuous and clearly defined; Tuesday comes after Monday, and a day lasts for 24 hours, et cetera. However, if we look at holidays; they all change based on where you live, what is your cultural background, and often company policies dictate when you have a holiday. The patterns how people travel are crucially different on a holiday versus a work day.

The irregularities of the times of holidays, and the people groups they affect pose difficulties in comparing the patterns, and ultimately understanding the data. The

questions such as “why is there a spike in the data on that particular date” or “is this date a holiday causing this anomaly” are often that come up. In these cases, the analyst may or may not have a feel for the effects of holidays in a particular data analysis problem. It is often left for the common sense, and the knowledge about the domain.

The problem with holidays may not seem such a big deal. One could argue that if such an anomaly is evident in the data, the solution is simply to check the calendar. In the case of simple analysis this may very well be the appropriate method. However, how to teach a computer to “check the calendar”, or how to automatize data mining over large datasets when the data is affected by holidays. In addition to knowing the special dates, the implicit information contained in certain holidays is arduous to model. How would you distinguish the Midsummer from the Easter celebration as they both have clearly different behavioral patterns?

3.3.1 Scripts for creating holiday dataset dynamically

The discovery phase for holiday data did not yield any worthy results. The open data sources for holiday data are not numerous. A couple of services offered Finnish holidays but they were either too expensive or behind a sketchy service provider. A third method for creating the holiday dataset was utilized. The holiday dataset was generated using a Python library. As the library used did not have support for Finnish holidays out of the box, the support was implemented by defining the logic for date generation.

The outputs from this configurable script are CSV files and JSON metadata for visualization. The output includes a true-false matrix where we have columns such as weekend, holiday, a day before holiday, and a day after holiday are defined. These vectors are also summed together to provide a weighted view for the visualization. Basically, the summed weights try to depict the number of people that are potentially on a holiday. This data can later be used in the traffic forecasting and other data analysis methods.

3.3.2 Visualizing holiday data

A comprehensive holiday dataset allowed the creation of interesting visualizations out of it. The benefit of having a holiday generator that can calculate the dates of holidays at will meant that we were not bound by the extent of the dataset. A decade of holiday dates

was generated and visualized in a polar visualization tool (Figure 3). This tool is built on D3 JavaScript library.

The dataset is read from a CSV file and a simple interface for changing the values shown in the graph was implemented. This allowed to browse the datasets interactively. The sum is the default view which will show the sum of the all holiday data in one view. It will emphasize the dates and times of the year when there are the most holidays. Hovering a mouse over the days will conveniently show the name of the holiday.

The data is visualized using a set of rings where each distinct slot represents a day. Ultimately, there are 365 days on each ring. The extra day on leap years is not drawn for simplicity. This, however, means that the leap years need to be considered on the data generation phase, when creating the CSV file and datasets. This metadata information is presented in a short JSON file that is also read by the visualization tool. The metadata contains the number of rings, i.e. years, and it describes the leap years by counting the days in a year.

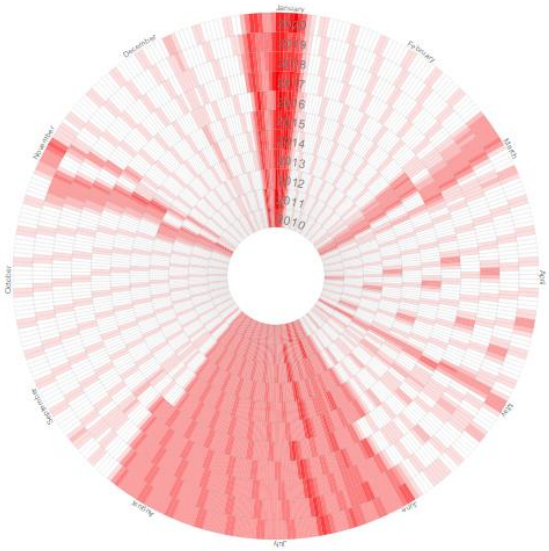


Figure 3. Polar visualization of holidays and weekends. The stronger the color the stronger an indication it is of a holiday date.

The visualization of holidays and the data outputs of this work help the subsequent research on data that changes based on holiday season. The visualization provides an overview and interactive way to assess which dates and times of the year are most suitable for data analysis and what seasons to compare for mining the effects of holidays. It is

effortless to see which months provide clean data with no holidays, and which are affected by the holidays. In addition to the visualization, the CSV outputs can be easily included in any data analysis project that needs this sort of information on holidays.

4 Preliminary traffic data analysis

In this chapter, we look for more conventional methods in data analysis. Instead of concentrating on visualizations, we analyze LAM data with data-driven machine learning. Thus, this chapter drifts away from the core subject presented in this thesis. However, it offers insight into the dataset and functioned as an important step along the process of understanding the dataset. Furthermore, the forecasting method used in this chapter can be later applied to the visualization artefacts discussed in the following chapters. These methods could extend the visualizations further and elevate them from simply displaying historical data to visualizing predictions based on this dataset.

4.1 Data sources and methods

The analysis of LAM data was done from two different data sources. For one we had the data collected through the open Digitraffic API. This data provided traffic volume and average speeds from all vehicle classes such as passenger cars, trucks, and buses. These different vehicle classes could not be separated from this data. The other dataset was acquired directly from the Finnish Transport Agency. This dataset was not as vast as the data collected from the open API which included hundreds of stations. The dataset contained only one station, Nuijamaa border station, at the border of Finland and Russia, and had data from March 2016 till the end of August 2016. This six months of data is very accurate, and it allows to distinguish different vehicle classes.

The methods used to analyze the datasets were simple numerical analysis and visualizations in spreadsheet program, and more detailed analysis in MATLAB, concentrating on forecasting traffic flows. In the spreadsheet approach, we visualize the dataset by color coding it in matrix form, calculating averages and counts over days which exceed set limit values, and by plotting charts of the results. The process tries to find definite traffic volume highs to each direction. In MATLAB analysis, the method utilized is based on research (Habtemichael and Cetin, 2016) where traffic is forecasted by identifying patterns. This data-driven method uses K-nearest neighbor algorithm to learn traffic pattern from history and applying that to predicting future short-term traffic flows.

4.2 Data overview

The first step before taking any of the two datasets for analysis was that they had to be preprocessed into evenly spaced time slots. This preprocessing stage was described in Chapter 3. Data used in this analysis is from Nuijamaa traffic station (LAM 306) which exists right on the border of Finland and Russia. The data extends from March 2016 till August 2016: a total of 172 days. The data is filtered in such a way that it only includes the traffic data from passenger cars. It does not count trucks or buses for example.

The evenly-spaced data was first analyzed using a spreadsheet program. Several helper fields were calculated and some field formats changed. The data was then transformed into a matrix form (Figure 4). As the dataset comes in even intervals we can identify a stride for daily data. For 15-minute intervals this stride value is 96 (96*15 minutes = 24 hours). The data transformed from rows of data fields to multiple matrices allowed for interesting color coding of data. Although, spreadsheet programs can be ill-suited for analyzing data in matrix form, it is still doable for simple cases.

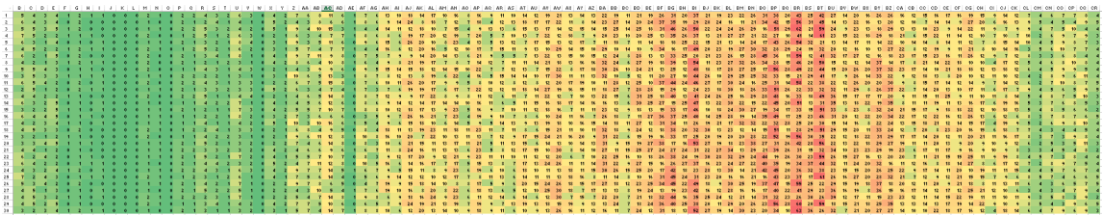


Figure 4. A quite effective data visualization can be achieved with simple color coding in spreadsheets. Every row shows data from a day. The columns represent 15-minute time intervals.

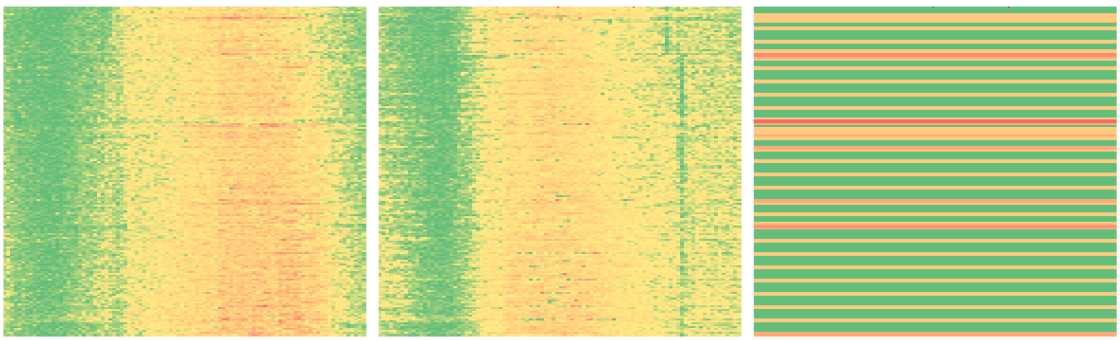


Figure 5. Colored spreadsheet data zoomed out produces pixel-oriented-like visualizations. Traffic volume in Nuijamaa station to east (left) and to west (middle). Holiday data on the right shows weekends and holidays for comparison. Overlapping

holidays and weekends are in stronger red color. The holiday data is defined by dates, not time of day, hence the horizontal lines.

After creating the matrix, the data can be inspected visually by zooming out in the spreadsheet program. This shows a type of pixel-oriented visualization (Figure 5). Even this sort of simple visualization starts to show patterns in the data. Most dominant is the daily pattern for each road direction. They are very similar in the long run. Moreover, the timing of major holidays shows slight anomalies in the data that can be seen from the visualization. It should be noted that this approach may have inaccuracies because of the scaling factor. A true pixel-oriented visualization displays data points as discrete pixels without scaling algorithms distorting the output (Keim, 2000).

The purpose of the first analysis was to inspect the data and calculate some metrics for defining an average day. A type of percentage of days over a limit chart was plotted. The amount of days over a hand-picked limit was counted for each 15-minute time frame. In this method, we look for definite times when the traffic volume is over a specified limit and we calculate percentages for these days occurring. Figure 6 shows the output for LAM station 306. From the figure, we can see that the westbound traffic is more shifted to mornings, and the eastbound traffic is more dominant in the evenings. This sort of pattern was not visible in stations further away from the border, only for this and another station very close to the border.

Additionally, the coefficient of variation (CV) is plotted as well. Coefficient of variation gives the reader of the chart a hint of how trustworthy the data might be. Especially at times of high traffic volume a missing data may increase the variation sharply. Variation will drop if the data is smoothed prior to analysis. Figure 7 shows the results for loess-smoothed data.

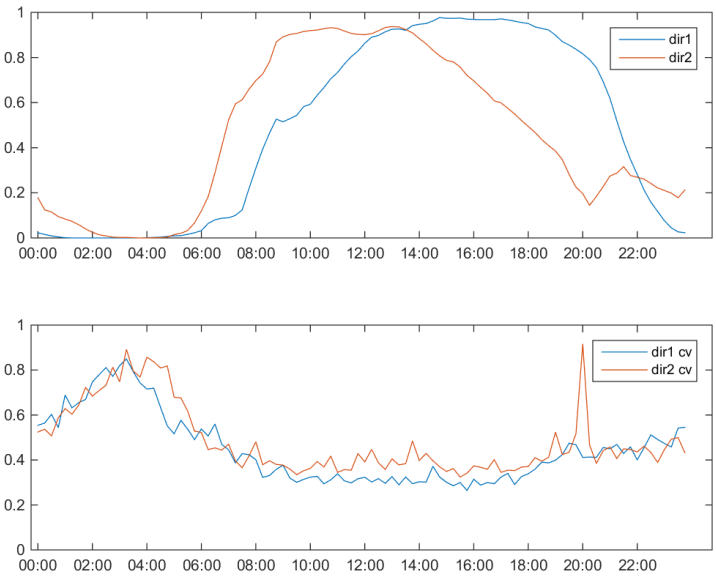


Figure 6. Percentage of days over a limit (50 veh/h) to each direction. Blue line is the direction to east, and red to west. The second chart shows the coefficient of variation.

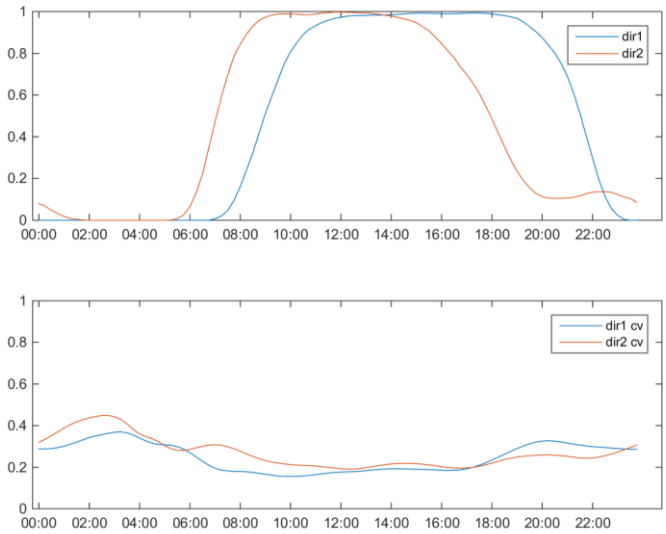


Figure 7. Loess-smoothed traffic data removes the sparsity in the data resulting in spikes in CV values. Naturally, CV is also slightly lower than for unsmoothed data.

4.3 Preprocessing

As previously mentioned the data was first processed into time slots that eliminates any sparsity the dataset has. Also, the unit of traffic volume was converted to vehicles per hour. Raw LAM data processed into 15-minute time slots would of course be in *vehicles per 15 minutes* and the data collected from Digitraffic API serves traffic volume in *vehicles per 5 minutes*. Using all this data together would be difficult in different units.

Thus, all the datasets were converted to *vehicles per hour* which is a unit often used in literature for traffic flow measurements.

In the real world, traffic is far from being a continuous and clean data source. Often, traffic may advance in pulses; a group of vehicles could pass a measurement station in one single file and the next measurement time frame shows no traffic whatsoever. It gives the data series a rough appearance. Also, the data collection could have technical problems. This noise in the data was smoothed out by using locally estimated scatterplot smoothing (loess) algorithm. Loess is a non-parametric regression method which is controlled by adjusting a span value. An appropriate span for the loess smoothing was selected visually (Figure 8). After a suitable span value was found, it was adjusted by the length of data to suit different datasets that may come in various lengths. This is important since the used span value depends on the interval and the length of the traffic volume dataset. Loess span of 0.005 was selected for 2 months of 15-minute interval data. This translates to about 30 data points of the whole set.

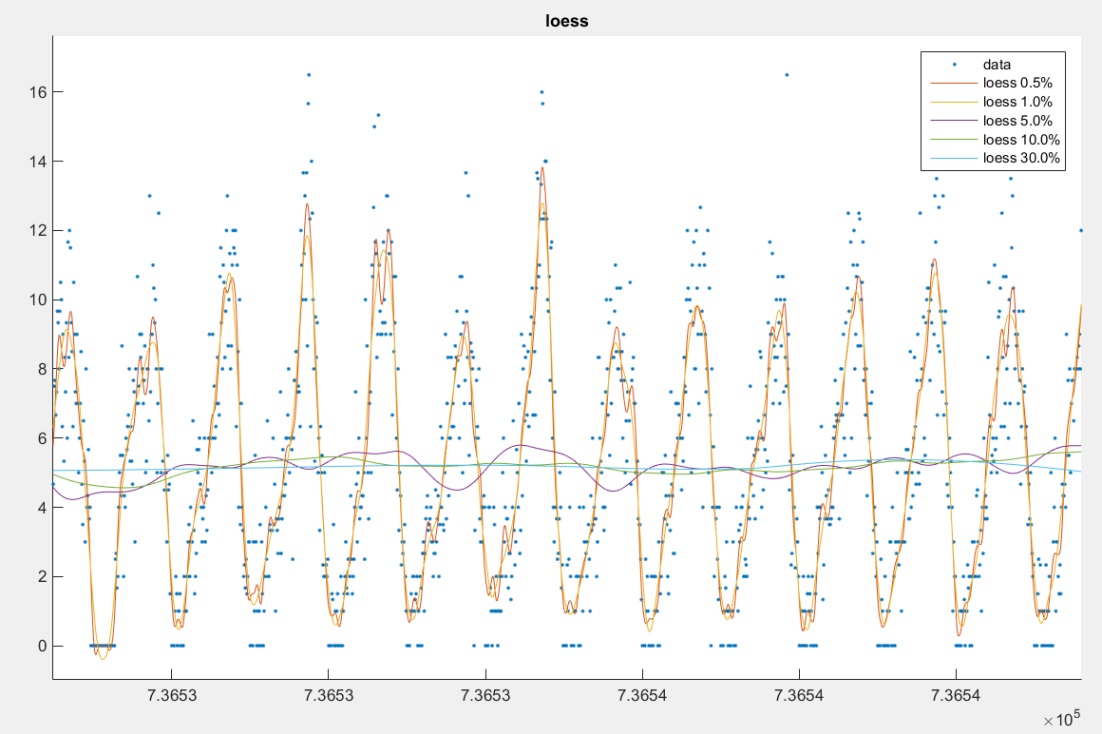


Figure 8. Configuring loess span value was done visually.

4.4 Daily traffic and holiday data

A visual inspection of holiday dates and daily aggregated data was conducted. Daily traffic was aggregated from the smoothed data by using numerical integration. Trapezoidal rule provided an effective way for estimating daily traffic volume. This daily data was then compared with the holiday dataset (Figure 9). The timing of holidays seemingly correlates with the timing of major changes in traffic volume. Also, the periodicity of weekends is clearly visible in the traffic through the border. It may be needless to say that holidays should be taken into consideration when conducting qualitative analyses based on traffic flow. Yet, it is interesting to note how the volume by the direction of travel behaves on dates that have holidays in both Finland and Russia, holiday only in Finland, or holiday only in Russia.

This chart provides a view based on purely quantitative data. It allows making analyses, educated guesses, and to gain insight to some extent. However, if it would have qualitative data such as why and where people travel at these times, it would make making analyses much easier. The added context and domain-knowledge is important even in these sorts of simple graphs visualizing people movement.

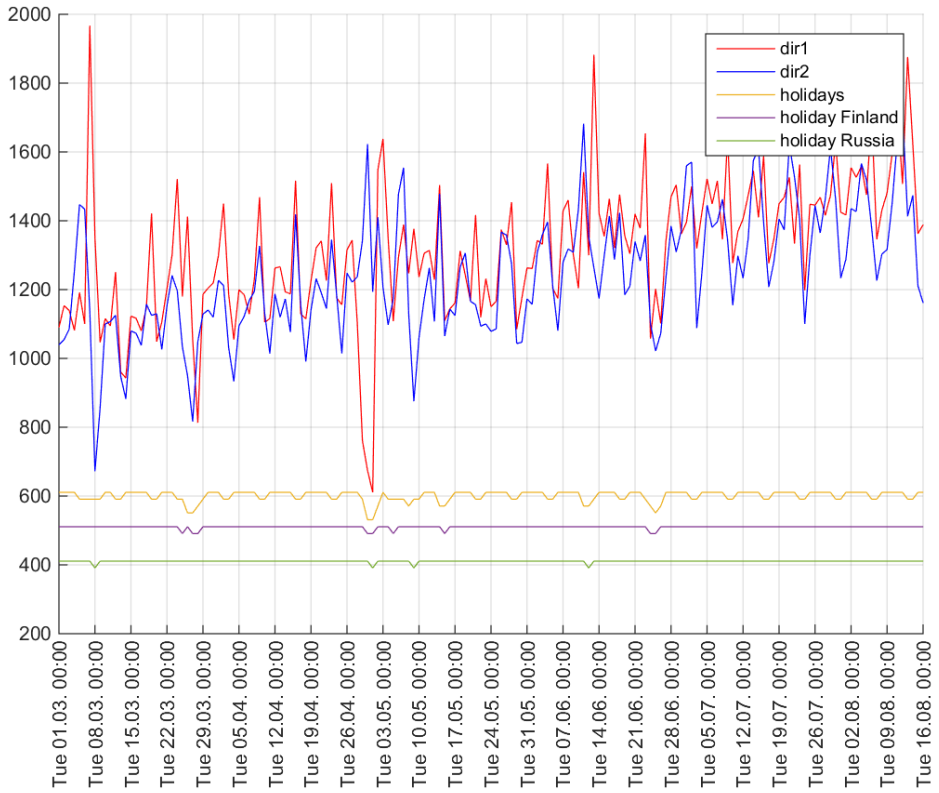


Figure 9. Daily traffic volume and holidays. LAM 306.

4.5 K-nearest neighbor method for traffic forecasting

Traffic flow forecasting has been widely researched topic lately. Numerous studies on the subject use various machine learning methods to predict traffic flows (Habtemichael and Cetin, 2016; Kumar and Vanajakshi, 2015; Min and Wynter, 2011; Yang et al., 2014). The motivation stems from Intelligent Transportation Systems (ITS) applications. Reliable traffic flow prediction can offer important information for ITS applications such as traffic management and traveler information. (Yang et al., 2014) Mainly, the research seeks to predict traffic volume or speeds short-term. A usual forecast duration is around one to two hours.

In this thesis, we apply a similar data-driven method as in a study by Habtemichael and Cetin (Habtemichael and Cetin, 2016) to our collected traffic volume data. The method seeks to find similar profiles from the history by using a type of K-nearest neighbor algorithm. The algorithm has several parameters:

- Maximum number of profiles
- Number of candidates (nearest neighbors)
- Lag duration, and
- Forecast distance

The number of profiles determines how far back in history we look for suitable candidate traffic profiles. For example, 90 profiles mean that the data from the last 90 days is used for candidate selection. The number of candidates dictate how many nearest neighbors we use to calculate the mean and predict the future traffic flow. The prediction is the mean of the nearest neighbors. Lag duration is the time frame we use to compare different profiles. Forecast distance is simply how far to the future we predict. The study (Habtemichael and Cetin, 2016) tried to find optimal values for these parameters. It favored small lag durations (1 hour was selected) and 10 candidate profiles. Naturally, the prediction accuracy is determined by the amount of data we have, more the better. However, it should be mentioned that looking too far back in history can be unreasonable as the old data may not reflect the current situation.

A KNN method requires a selection of a similarity measure. We chose to use root-mean-square error (RMSE) metric. It is important to remember that the data was cleaned using loess smoothing. We set up a controlled experiment for optimizing the variables in our case. Four runs of simulated predictions were done, each time choosing one independent variable. The fixed, default values for variables were in each case the following: 90 profiles, 5 candidates, 1-hour lag, and 1 hour forecast distance.

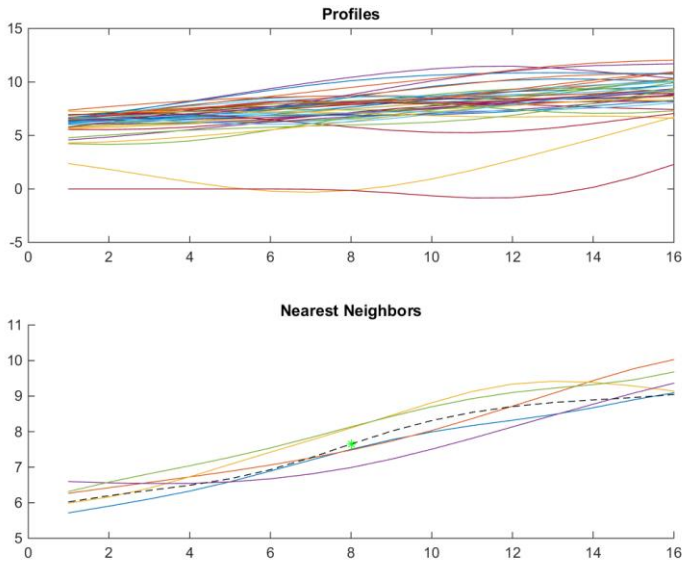


Figure 10. An example of the nearest neighbor selection (k=5) using RMSE as the similarity measure.

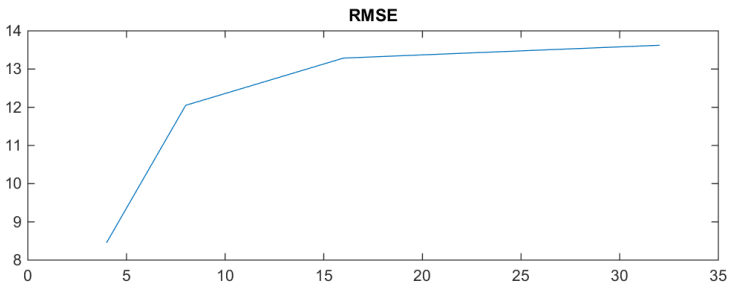


Figure 11. Forecast distance. Prediction error increases sharply from 1-hour (4 timeframes) to 2-hour (8 timeframes) forecast.

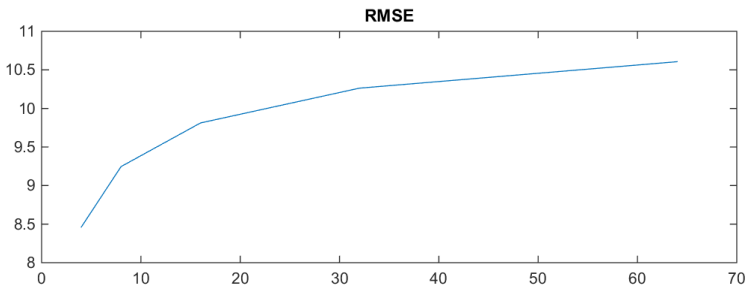


Figure 12. Lag duration. Short lag duration led to better prediction accuracy.

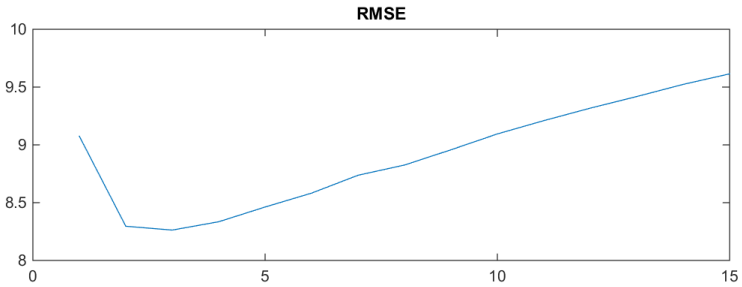


Figure 13. Number of candidate profiles. Better accuracy with a relatively small number of candidates.

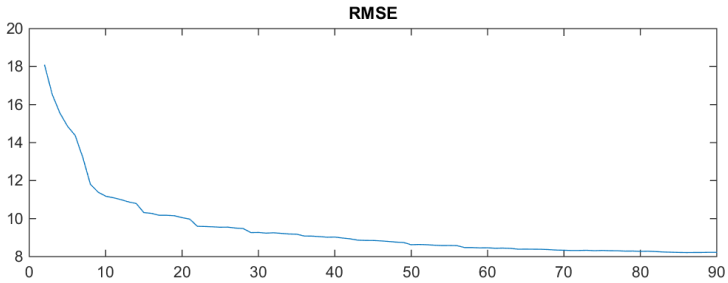


Figure 14. Maximum number of profiles. The prediction accuracy gets better the more data we have.

The analysis shows clearly that the method is useful for short-term traffic flow forecasting (Figure 11) but does not work too well for 2 hour and longer predictions. The prediction error evens out for longer than 4 hour forecasts possibly because the prediction starts to approach the mean of all the daily patterns. The optimal number of candidates (Figure 13) is much lower in our case than it was in the study by Habtemichael and Cetin. One of the reason could be that our data has much higher variance than the data in their case. Other reason might be that we need data from a longer period.

The KDD method selects candidate profiles by similarity. It is interesting to note that, for example, predicting a Saturday brought up profiles for other previous Saturdays to the list of nearest neighbors. These sorts of patterns can easily be hidden to the naked eye but the algorithm, by comparing these previous times, can mine information that shows similarity in patterns for specific weekdays.

5 Development of visualization artefacts

In this chapter, we review the steps in creating the visualization artefacts. The primary visualization artefacts created in this thesis are spatial 2D visualizations created using GIS software and web-based spatio-temporal 3D visualizations. As noted in the Chapter 2, the development of data-driven visualization tools is similar to knowledge discovery and data mining. Creating a useful computer-based visualization tool is not a trivial task. It requires iterative development, taking human cognition and mental model into account, and strong software development practice. A major part of a complex data-driven visualization comes from working with databases and transforming the data into suitable formats.

5.1 Spatial travel time to events visualization

One of the primary cases in this study was to build a visual analysis tool to discover events (destinations) that are difficult to reach from the nearest transportation hub. As a part of a travel chain, easy access from the hubs to destination can be a major selling point for a transportation company operating in those hubs. Often the destination may not be located in the immediate vicinity of the hub. Thus, another mode of transport may have to be considered. When using public transportation, it usually means that you do not have a car available for use in the hub and you need to reach the destination by local buses, for example. This last part of the trip, the last leg in the itinerary, is called the last mile.

The visualization presented here is effectively a measure for the travel time of the last mile. It can also work as a measure for the effectiveness of the last local connection from the major transportation hub, especially for destinations farther away. It highlights destinations that are hard to reach by several metrics. Considering the travel chain is important for both public transportation providers and event organizers. Public transportation is concentrated around traffic hubs as it allows building efficient travel chains. Travelling between these hubs is often very efficient. For a transportation company it is interesting to see how the travel chain fares to destinations further away from these hubs. In turn, event organizers are interested on how many visitors the event

attracts as having multiple transportation options to the event can give boost to the number of visitors.

5.1.1 Data sources

The following four data sources were used in this visualization: event data from South Karelian event calendar API (Etelä-Karjalan liitto, n.d.), event data from Finland 100 web page (Finland 100 Years, n.d.), train station data from an open API by Finnish Transport Agency (Finnish Transport Agency, n.d.), and journey planning and travel time data from Digitransit API (Finnish Transport Agency, n.d.).

The South Karelian event calendar data was the starting point. The API features two commands. One for getting an overview for all the events in the database. This type of command can be filtered by municipality, category, or target class. The other one is to get the details of an event which includes the geographic coordinates. Spatial information is not returned in the overview response. This means that in order to get the coordinates for all the events we need to make the same number of requests to the API as there are events.

Another event dataset was got from Suomi 100 project. Unfortunately, they do not provide a public API for the data so the only option is to use web scraping or look for AJAX implementations that may provide data in machine readable format. In Suomi 100 case, their website was partly created using this technology, and provided events dataset in a single JSON file. Although, this data source is not meant for public, is undocumented, and may change at any time, it is still usable for one-time data extraction.

Whereas the event data describe the destinations in the visualization, the train stations are the transportation hubs. Train stations were extracted from Digitraffic API for railroad traffic. The API is well-defined, and developed for nation-wide use. The station details can be get with a single request to the API. This returns descriptions of the stations as well as the spatial information.

The fourth dataset comes from a newly released journey planning API called Digitransit. Digitransit API has features for route planning, geocoding, background raster maps, and real-time information. The API uses GraphQL extensively. GraphQL enables clients to get a tailored response from the server. Thus, no extraneous data is returned – only the

data we want to get. The service offers many sorts of data. We utilized the itineraries between two locations. The itineraries work like travel chains, and include calculated attributes such as travel time, walking distance, or used transport modes.

5.1.2 Data flows

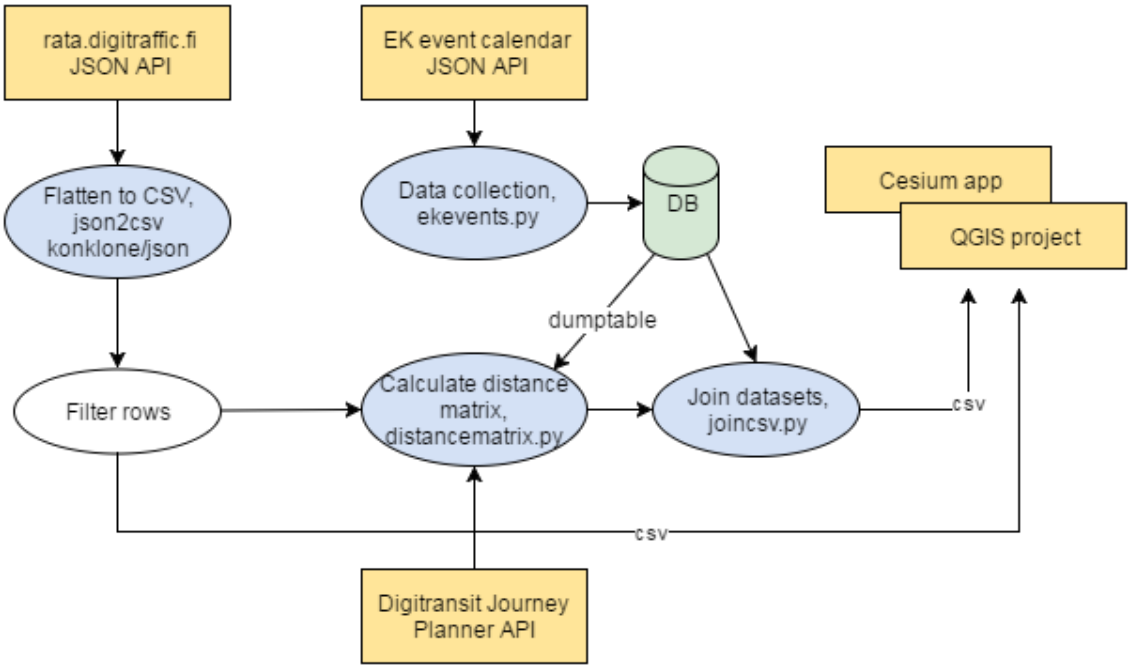


Figure 15. Cesium events visualization data flow diagram

In Figure 15, we describe the data flows for creating the visualizations. The data flows work similarly for both QGIS visualization and the web-based Cesium app. The web-based tool is described later in this chapter.

The South Karelian events API does not provide historical data, only the events after the current date. Thus, we used a data collection script over time to collect events to a permanent database. The relational database then serves as a starting point for data selection and preprocessing. Handling rows of data is much easier in a database than if the data would be stored as CSV files. Database actions such as data selection or filtering are trivial things to do and are implicitly described in the “dumptable” action.

In addition to using databases, a simple download of dataset was used for train station data. Station data is rather static and unchanging so a single time conversion is suitable in this case. Nonetheless, it requires similar actions in data selection. The hierarchical

JSON response needs to be flattened to a CSV file. Also, all the redundant rows should be filtered out. These are exactly the same actions that are done for the events data in the relational database. Moreover, the JSON response from Suomi 100 website had to be processed similarly to the previous datasets; flatten and filter.

The spatial information provides the data for the next step; distance matrix calculation. As we are exploring distances relative to stations, and that there are multiple stations, we are required to build a distance matrix between stations and destinations. The creation of the distance matrix is explained in the following section. All the above three datasets contain spatial coordinates in WGS 84 coordinate system. Thus, there is no need to do coordinate system transformations.

The distances and travel times from distance matrix need to be joined with the actual event data. Again, this is a trivial task in databases but doing it for two CSV files requires a little bit more work. It is true that the distance matrix calculation could very well be integrated to work with the existing databases so that the table joins could be done in SQL, and the output from the databases would then be datasets ready for visualizing. However, a more generic method of directly handling CSV files for this task was devised. Thus, data not in the database can be processed as well. Joining datasets by field identifiers is immensely important aspect to consider if deciding not to use a database throughout the whole process.

The process in Figure 15 illustrates the specific data flows for the Cesium visualization of travel time to events. In the QGIS enabled visualization of the same data, the table joins were done in the QGIS software, not by a separate tool. This underlines the fact that how important joining data between datasets is. Whether you do it for a web-based visualization or GIS software you need to consider how to combine datasets by fields. After the datasets have been created and combined, they are visualized. The same datasets were used as inputs for both QGIS and Cesium based visualization.

5.1.3 Distance matrix of public transportation travel times

The distance matrix was computed between the traffic hubs and the destinations (events). This is the first stage of the algorithm. The distance matrix is created between these two

datasets using great-circle distance computed using the Haversine formula. After the distance matrix is created and the closest hubs are found per destination, the travel time to the destination is calculated. An external service was used for calculating the travel time. The travel time can be get from Digitransit journey planning API. The API does not have a function to return the shorted travel time to the destination in which we are interested. Instead, we fetch a few different itineraries and select the quickest one. Although, this does not guarantee that it finds the absolutely best connection. Still, the same connections are recurring daily, so the whole search space does not have to be searched.

In the first step an N-times-N matrix of great-circle distances is created. It would be possible to calculate the travel times in the same extent but the retrieval of the travel times from the external service takes considerable amount of time so only the nearest occurrence is used. This makes the output a vector. This vector is then directly joined to the events data as a column. Besides the travel time that we are after, we include some additional information returned from the service such as walking time. The approach would allow to get much more additional information such as the type of transportation or travelled distance.

5.1.4 QGIS visualization

In the first phase the datasets gathered were visualized using a GIS (geographic information system) software called QGIS. This allowed to preview the data and examine its correctness quickly. QGIS is handy for exploring datasets in an easy to use interface without putting a great deal of effort. The tool allowed to join datasets by fields, and combine distance matrix data with the events data without the need for additional tools. After joining datasets, derived attributes were calculated. By using the Field Calculator feature, attributes such as human-readable travel time were formed.

QGIS has very efficient categorization tools for the datasets. Features can be colored or scaled based on the values they contain. The classification of features was chosen such that it does not have too many classes but just enough to show the events with small travel times from the problematic, unreachable events.

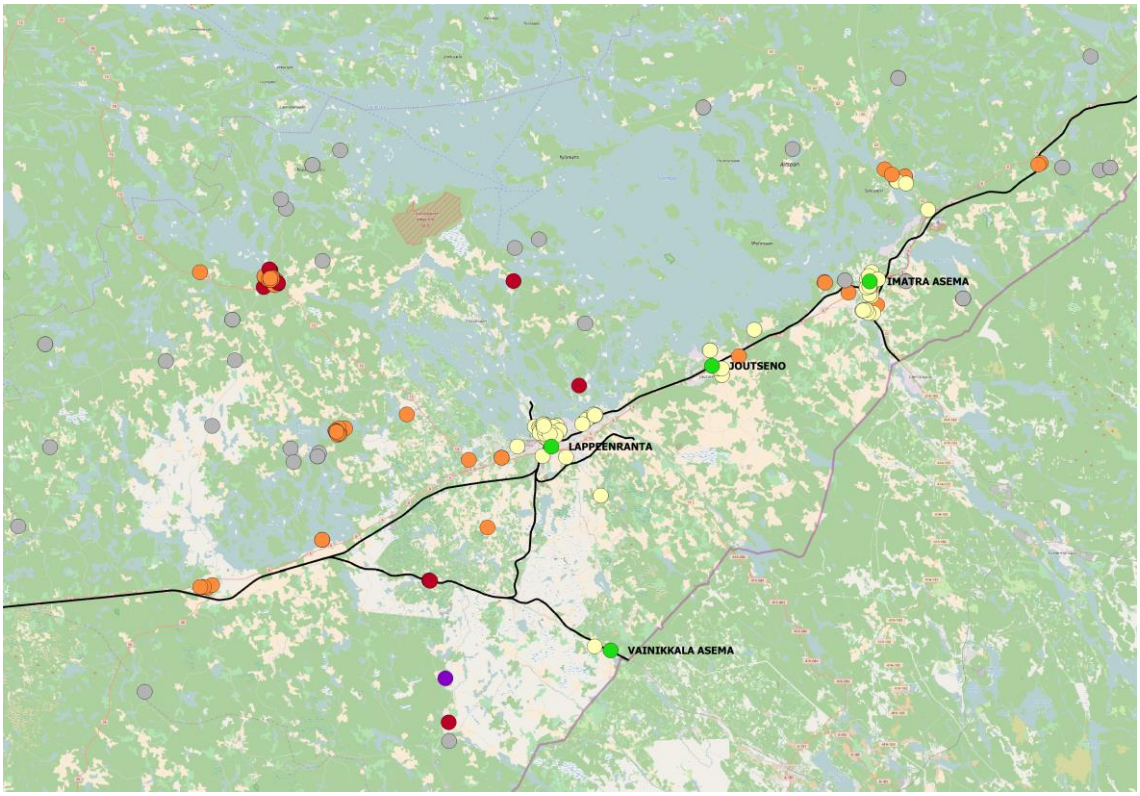


Figure 16. Standard QGIS output for printing. Gray: events with no connection, yellow: under 30 minutes travel time, orange: 30 to 60 minutes, red: 1 to 2 hours, and purple: over 2 hours.

In the end, the visualization does what it supposed to. It shows events, colored by the travel times to them, and creates a printable output (Figure 16). Yet, the visualization is not very interesting. It does not have any interaction element to it apart from the QGIS interface, and the output is a plain map with attributes and a legend box.

5.1.5 Qgis2threejs plugin - From 2D to 3D

Qgis2threejs is a plugin for QGIS software. It creates a 3D scene from the layers in a QGIS project. It has several options. Map canvas can be configured and the tool even allows adding height geometry for displaying realistic terrain e.g. mountains (see Appendix II). Qgis2threejs supports point, line, and polygon geometry. The plugin uses three.js library for WebGL rendering, and Proj4js for coordinate system transformations.

Qgis2threejs plugin produced scene is static. The base map is directly saved to the JavaScript output file as Base64 encoded variable. The cylinders and other geometries representing spatial data points are rendered on the map plane. Due to the simplicity of

the implementation, very complex scenes are either too cluttered or perform slowly as the amount of geometry to be rendered increases.

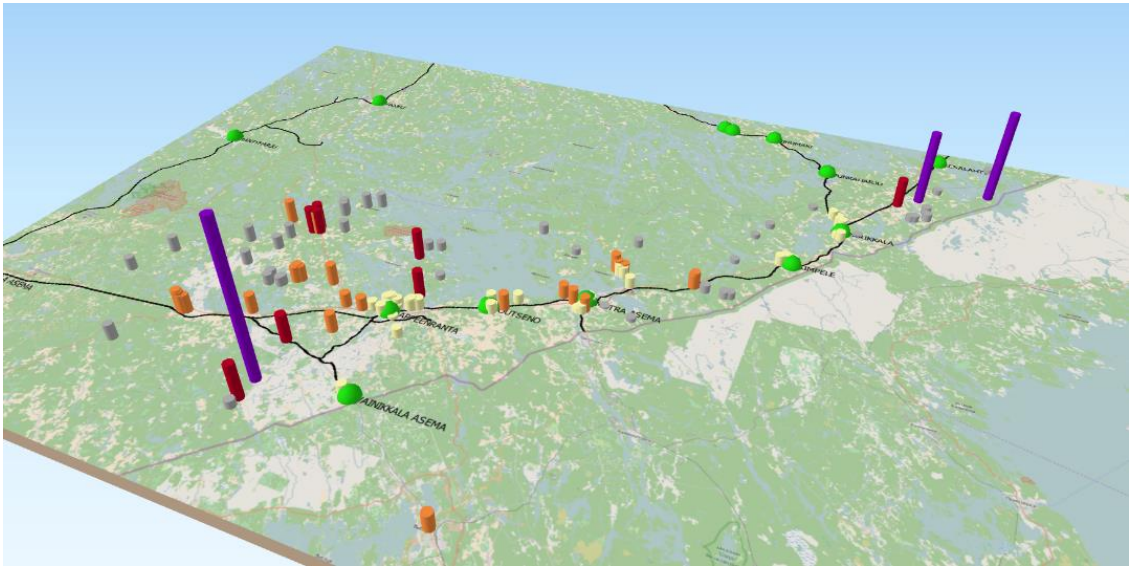


Figure 17. QGIS project converted to 3D using Qgis2threejs plugin.

The camera can be controlled; moved around a point and zoomed in and out. However, this does not enable the user to see more detailed base map on a closer camera position, more refined geometries, or distinguish entities close to each other. It is more than likely that several data points are located close to each other, forcing the user to decide between large geometries that work better when camera is far from the plane, and tiny or narrow geometries that give better separability of different entities but are hard to distinguish from farther camera angles.

In addition to issues in rendering 3D geometry, the implementation has poor support for displaying human-readable feature attributes. At the time of writing, the plugin did not support field name aliases in QGIS. When exporting attributes, the tool uses field names from the data file which can be rather cryptic at times. Field name aliases are meant to be human-readable and replace the plain field names from the CSV. Moreover, there is no support for selecting what attributes to export. Many field attributes calculated in QGIS may not have other use than the internal QGIS functionality or that they remain unused altogether. These sorts of attributes ought to be filtered out for pretty, clutter-free visualizations. After all, we would prefer to only display the information that is useful for

the task as redundant elements in the visualization can easily distract the user from what is important and what ideas we are trying to convey.

The implementation requires several changes to be more user-friendly. Field names should be displayed using their aliases. Redundant field names should be omitted – the configuration interface should have functionality for selecting specific fields. More importantly, the visualization should have rendering functionality for allowing effective zooming in and out. That means that the entities in view must be dynamically scaled based on the camera position, field of view, and such. This would allow the entities close to or on top of each other to be distinguished when zooming in. Another aspect is the usability of the camera. The plugin has two different camera modes. The camera modes, however, allow too much freedom which in such a simple visualization nonessentially complicates interaction. The "look at" point[6] of the camera could very well be fixed to the map plane. Also, limiting the range of rotation would prevent the camera from travelling to negative angles. The ability to see what is under the map plane is hardly of any use.

5.1.6 Cesium visualization – Usability improvements in 3D

The first 2D and 3D visualization in QGIS of events data were inadequate and rather clumsy. However, the 3D visualization demonstrated the value of the added dimension where the vertical space can also be used as an indicative method in addition to 2D-area and color. As a replacement, a web-based Cesium visualization tool was built.

The Cesium app was built on top of a shared architecture with the LAM data visualization tool. The general architecture of this tool is discussed later in this thesis. However, there are differences. The data fed to the visualization are the same CSV files as they are for the QGIS visualizations. The CSV files are read using an implementation from D3 library. This parses the files and provides them as JavaScript objects. The objects are further processed into custom Cesium data sources. This creates a one-to-one relationship

[6] A camera, as understood in 3D graphics, can be defined by two points in space. The position of the eye (camera) and the position where the eye looks at.

between CSV files and data sources in Cesium. These CSV files can be directly downloaded from the visualization interface.

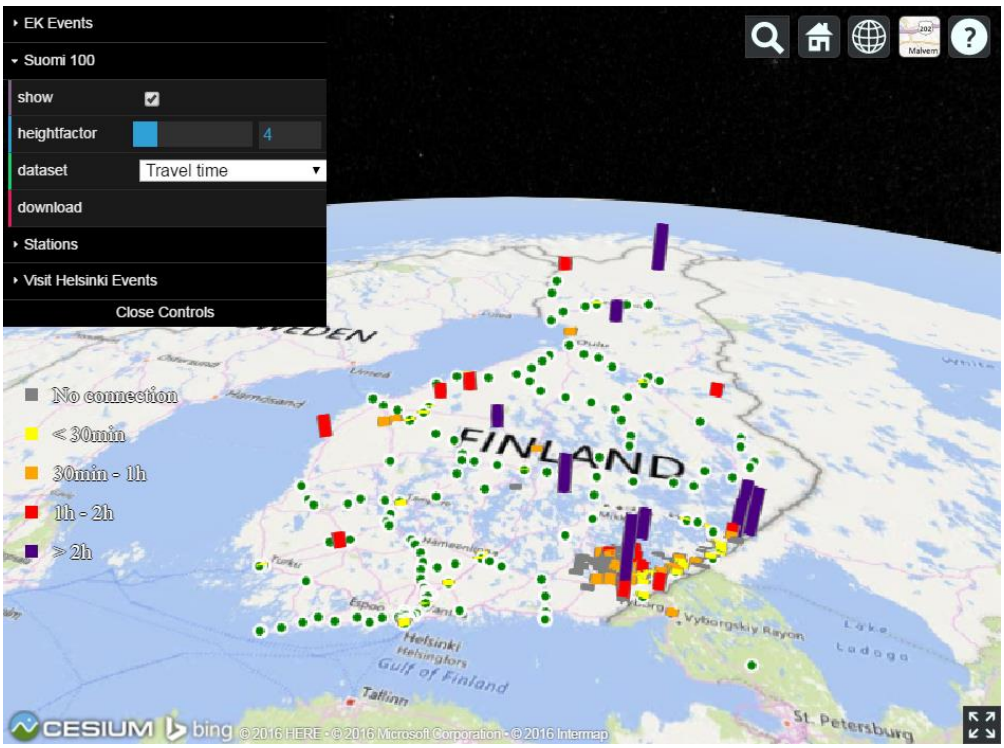


Figure 18. Cesium visualization of events

In the visualization (Figure 18) these data sources can be ticked on and off. There are several other functions as well. The visualization has three selectable datasets; travel time, relative travel time per distance, and walking distance. Travel time is the time it takes to travel to the event from the nearest station using public transportation. The travel time per distance is effectively inverted speed, and walking distance is the walking required when using the fastest public transportation connection.

These datasets are visualized in the heights of the bars. As such, the colored bars represent the events in the visualization. The height can further be altered by some factor. This is purely a convenience method if there are a lot of similar values in the dataset, or that the values are generally too small or large. This is also helpful when viewing the dataset from a distance or very close. However, optimally this should not be manual but calculated from the viewing distance and the data point density.

The legend in the visualization shows the categorization of the events data. The values are categorized to rough classes by travel time. The classes are; no connection, less than 30 minutes, 30 to 60 minutes, 1 hour to 2 hours, and lastly over 2-hour travel times. This is visualized in color. It gives a quick overview where the travel chain is inadequate, as stronger the color the longer it takes to get to that event from the nearest train station.

The Cesium approach proved to have many superior qualities over the quick QGIS exports. Because of the Cesium WebGL implementation, it had faster rendering times of the same data. Cesium features the whole world in 2D or 3D and features detailed map tiles downloaded on-demand from an external service when zooming in. This allows great freedom on what spatial data to visualize and it does not require the map canvas to be generated when the data changes. As it is a proven web framework, the framework is extensible and updated frequently. This interactive visualization is easy-to-use and it encourages users to interact with the data because of a robust user interface.

5.2 Automatic traffic measurement data

The second visualization artefact presented in this thesis is a spatio-temporal visualization of LAM station traffic data. It is an exploratory data analysis tool that includes traffic volume, average speed, and relative speed deviation metrics. The tool can be used to discover anomalies and patterns in traffic data. The data can be browsed using a simple UI and the data can be exported as HDF5 files. The visualization tool features 2D charts as linked views in addition to the 3D view.

5.2.1 Data sources

The datasets used in this visualization come from Digitraffic API (Finnish Transport Agency, 2016a). The service provides data from four different sources: travel time system, automatic traffic measuring stations (LAM), road weather stations, and road surface imaging. The data we used in this visualization is the LAM data. The automatic traffic measurement stations *"collect data about traffic amounts and speeds by the induction loops embedded in the road surface"*. (Finnish Transport Agency, 2015)

The service provides two kinds of datasets; static metadata, and real-time data. There are multitudes of metadata available but we naturally were interested in the descriptions for LAM stations. The LAM station metadata contains information such as the names and coordinates of the stations. The real-time data is served in XML format. The data does not include historical data so we recorded this data over a time period into a database using the data collection scripts described earlier in this thesis.

The real-time data for LAM stations is updated every 5 minutes. The data includes traffic volumes to each direction, and average speed of the vehicles recorded over that 5-minute period to each direction. This produces us four different data series over time for each LAM station. The direction of the road is determined by the road register address. Direction 1 is to the increasing address direction, and direction 2 vice versa. (Finnish Transport Agency, 2016b) The unit for traffic volume is reported as vehicles per 5 minutes. This is later converted to vehicles per hour, and all our visualizations use vehicles per hour as the default unit. The data obtained is effectively spatial, temporal, and multidimensional, all of which make it hard to visualize and handle in a 2-dimensional spreadsheet program.

Ultimately, the datasets visualized from this data are the traffic volume, average speed, and a derived metric; speed deviation. The speed deviation is calculated from the current average speed compared to the long-term average. This shows temporary drops in speed and could work as an indication of congestion because of traffic accidents, roadworks, or simply – overcapacity.

5.2.2 Methods to visualize in 2D

Several ways to visualize the data was examined. The first visualization was to simply plot the data on a map plane in QGIS and use color coding for displaying values. This was done for both traffic volume and average speed datasets. Again, QGIS provided a preview of the data at hand. The problem with this very simple approach is that it cannot display but one dimension per measurement point. Thus, only one direction can be visualized in the same view. A second way to visualize the data was to show the both values for each direction for each measurement station. Although, this shows both dimensions for each station, the visualization gets rather cluttered and is very hard to read.

Building on top of the single point per measurement station, an arrow indicator was considered. The idea for showing an arrow for each station was to calculate a derived metric for the general traffic direction. As we have traffic volume over time for both directions, it is simple to calculate. So, the arrow direction is determined by which direction has more traffic. This could be useful for indicating cyclical patterns such as commuting traffic where a certain road shows considerable traffic to one direction in the morning and to the other in the evening. However, it was not seen very useful because it hides the underlying directional data.

LAM stations are numerous as there are a total of 481 distinct station identifiers in our database. Because the network of these measurement stations is so dense, an alternative way to visualize them using lines was devised. Connecting each measurement station on a specific road gives us a *multiline* geometry. This geometry quite accurately follows the real geometry of the road when there are enough stations along the way. The problem with two directions must still be handled. To overcome this, the line geometry is separated for each direction and displaced slightly. This is imperative since a single line cannot visualize traffic to both directions.

The line geometry was by far the best method on a dense network of measurement stations. The values can be visualized using line width (Figure 20) or color (Figure 19). This visualization gives us so much freedom in dimensions that both the traffic volume and average speed could be displayed at once in a single view. One would be shown as line width and the other as line color. Sadly, QGIS software does not support this so only one category could be implemented. Although, it does sound lucrative to include them both, it might be more of a hindrance and ultimately limit the usefulness of the visualization. Less could be better in this case since the complexity of the visualization is already quite high.

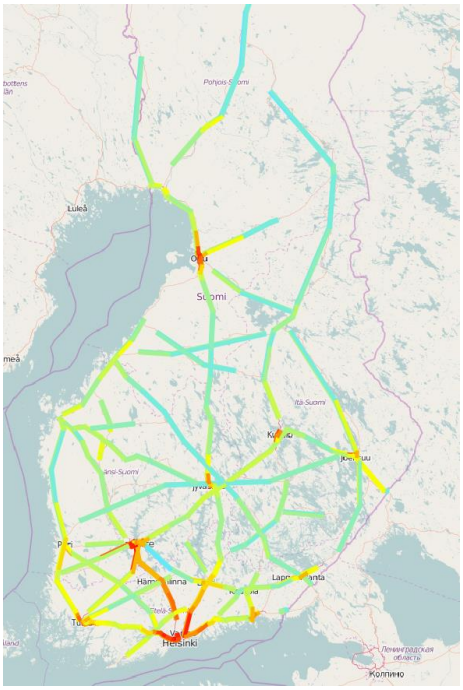


Figure 19. Traffic volume visualized using a color scale.

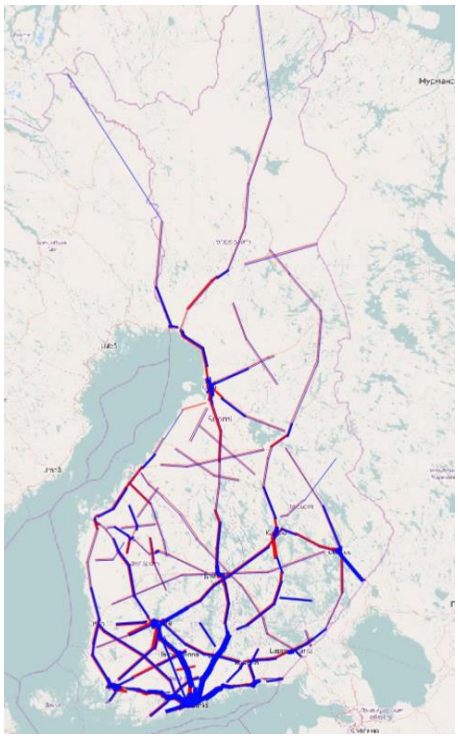


Figure 20. Traffic volume visualized using line width. Blue lines are for direction 1 and red lines for direction 2. Blue lines dominate in this figure because the draw order of the lines is not determined by the weight of each line. A problem in QGIS is that it does not feature adjusting draw order based on attribute values.

5.2.3 Creation of the line geometry

The kind of displaced line geometry required for this visualization is not trivial to come by. To the best of our knowledge, there are no generic tool for converting these sort of data points into line geometry. There are several rules we need to create. Firstly, the lines are constructed based on road structure; one geometry for each road. It would not make sense to connect LAM stations that do not exist on the same road. Secondly, there are two different geometries for each road that need to be displaced appropriately so that they are not overlapping.

The construction of this geometry required the use of a spatial database. The first phase is to create the displaced point geometry for each LAM station. The displacement is calculated by iterating ordered LAM stations in a certain road, calculating vectors

between them, and finally getting a perpendicular vector to the displaced position. These positions are later chained into polyline geometries. This results in two polyline geometries for each road (both directions).

The geometry above is ideal for the Cesium visualization created in this thesis. However, the QGIS visualization required a somewhat different geometry. For QGIS the road polylines were required to split into simple lines between two points. Without this split neither line width nor color could be accurately visualized using QGIS. This shows how specific these problems are. Even though we are visualizing the same data, the tool used dictates the format we need to feed the data.

It should be mentioned that the Cesium implementation could handle the geometry created for QGIS. The main driving force for creating the new geometry for Cesium was performance related. The amount of different entities in the split geometry was easily rendered in 2D in QGIS but the Cesium 3D implementation had hard time rendering all the entities, thus the reduction in the number of entities was done.

5.2.4 Temporal visualization in QGIS

The temporal dimension of the data was first demonstrated by using a QGIS plugin called Time Manager. The Time Manager Plugin simply filters out the data that does not exist in the configured range. In our case the time frame could be 15 minutes at a time. These 15-minute time slots are then iterated over the whole dataset which creates an animation of the data over time. The plugin allows to save this animation as separate pictures which then can be combined into a video.

The plugin poses several problems. First of all, since the output images are generated straight from the sub-window displaying the map and the data features or points, the output resolution is very hard to control accurately. We would like to generate images that are directly saved in the same resolution as common video resolutions such as 1080p, or 720p. Currently, this is not possible. Moreover, the generation of the animation cannot be done programmatically in practice.

Another problem is created by the filtering method. Large datasets such as this that were exported from a CSV file are too slow to be processed in this manner. This, however, is

not just a problem with the plugin but the QGIS software itself. With a lot of data, handling that kind of data source can become slow when filtered or shown all at once. Fortunately, the filtering can work at SQL statement level. Thus, when QGIS taps directly into a database, it can increase the filtering and data rendering speeds considerably. The use of databases in this manner requires a spatial database that is supported by the QGIS software. One of these spatial databases is Spatialite as described in Chapter 3.

5.2.5 LAM data visualization in Cesium

As in the events visualization, the second version of LAM data visualization was built on Cesium framework. Instead of reading and converting CSV files into Cesium data structures, this application reads dynamically generated CZML from the remote server based on a request query. At first, the geometry in the CZML file was generated based on the line geometry created for QGIS. Although it did work, the amount of entities led to poor performance. Thus, an alternative geometry was constructed as described in one of the previous sections.

A type of wall geometry is used in this visualization. Wall geometry is basically a polyline with an added height dimension perpendicular to the Earth's surface. The three datasets are then visualized using this height dimension. The direction of the road is separated by color; blue for direction 1, and red for direction 2. The walls were made transparent to improve the visibility of the entities behind high walls. Especially so that the both directions can be seen from any camera angle at the same time. The color of the wall entities is scaled based on road number to distinguish different roads and to separate minor roads from major ones.

The volume of the remote data is very large. Thus, generating CZML markup and sending it to a client takes a considerable amount of time. In practice, the data needs to be send in smaller chunks. This is a common practice in interactive websites to request more data as needed. Thus, a data query feature was implemented. It allows the user to ask for data from a specified period filtered by roads. The user can select which of the three datasets to use. The query gets a response from the server in CZML. This response includes the wall geometry for the roads, and additional data for the linked views.

In order to handle the vastness of the data in view at once, a filtering function by road number was implemented. The user may be interested in analyzing more than one road at a time, yet, displaying all the roads at once is excessive. The filtering was implemented both in the client and server. The server-side data filtering reduces the amount of data sent to the client thus making the data retrieval faster. The client-side filtering helps to build the interactivity aspect of the web app. The input into the filtering function is parsed such that it allows to specify roads separated by commas, and road ranges separated by dashes. An example input could look like: 1,2,5-10. This would return the data for roads 1,2, and 5 through 10.

The animation of traffic patterns is beautiful to look at. It can offer insight into where are the traffic hot spots, and how does the traffic behave over time. However, it can be hard to look for something specific that may happen at any time. To catch this kind of an event the user would be required to replay the whole data while watching for changes in the visualization. The solution for this was to build additional views that that show specific data when an entity is selected in the interface. These *linked views* offer both aggregate data about roads, and 2D-plotted overviews of the data for each measurement station.

The LAM station overview shows the dataset for that stations over time. This is shown in one single 2D chart. From this chart, we can look for anomalies, or verify observations we made from the running animation concerning that station. The other overview is for the roads. By selecting road geometry for any direction, a summary of aggregate values calculated from all the stations along that road is shown to the user. This shows a kind of profiles for each road. The dataset used in linked views is always the same that is shown on the 3D map.

A comparison between an average weekday and a day at the start of a major holiday is demonstrated in Figures 21 and 22. The selected LAM traffic station 142 is shown in the middle. On the lower right portion of the figures we see the traffic volume for the selected day at 15-minute intervals. The first figure shows a traffic pattern where the morning commute and the evening commute traffic volume peaks are clearly visible. In the second figure, however, the outgoing, northbound traffic is very dominant in the evening. This shows a typical holiday pattern. Observing this station over the next couple of days shows that the traffic volume peak shifts closer to midday as people are no longer tied by work

till 4 pm. This also shows that the holiday traffic for major holidays is a multi-day anomaly. In Appendix III we show another comparison using the linked views for roads.

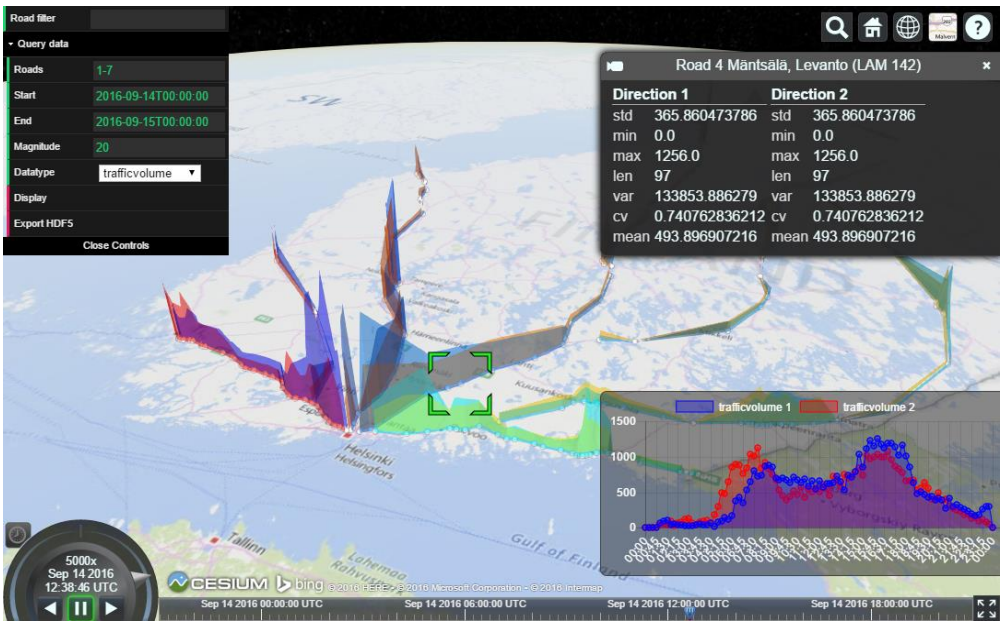


Figure 21. An ordinary weekday commuting traffic pattern.

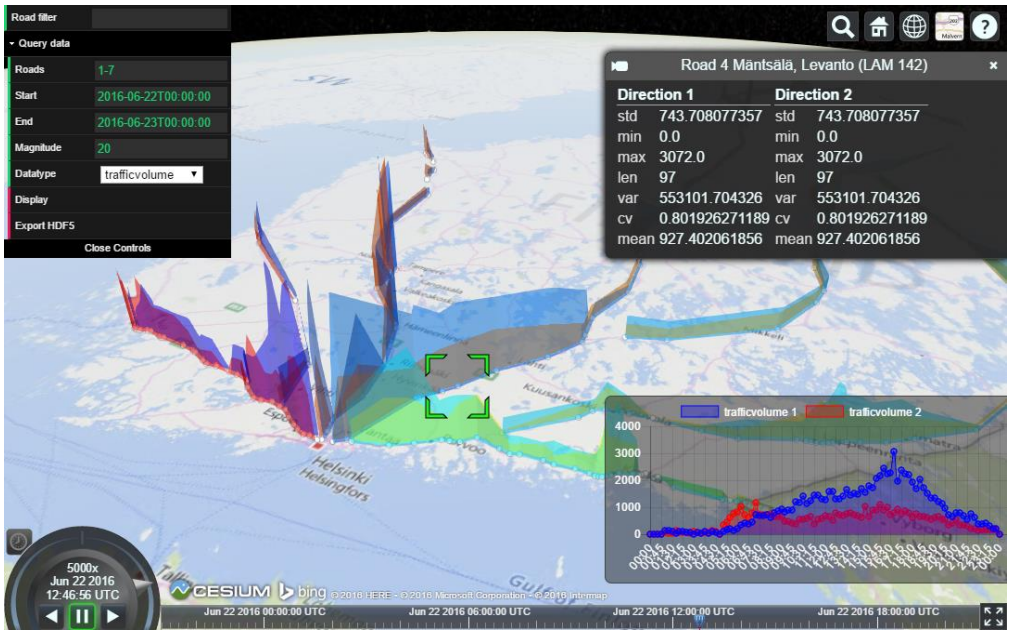


Figure 22. Traffic anomaly in the beginning of the Midsummer holidays.

5.2.6 CZML generation

CZML provides a high-level abstraction for data-driven Cesium visualizations. It is based on JSON, supports streaming data, and is an open, extensible format. The format describes spatial graphical primitives in a time-dynamic scene. (Hunter, 2016)

CZML is used to describe the wall geometries for the roads in the scene in this visualization. Each road has two entities with wall geometry for each direction. In addition to the wall geometry and description information, a set of custom attributes is added. The custom attributes are not described in the CZML and they need to be read in a custom implementation in the client. This is where the strength of this extensible format lies – we can embed data that is not directly supported by the format. In addition to entities for roads, a set of entities from each LAM station is created. These entities serve as an indication of the station locations, and provide specific datasets for each station which are not apparent from the generic road entities.

The generation of the CZML response is done in line with the processing and reading of data from database. As soon as, a dataset for a given road is read, aggregates are calculated and it is wrapped into a CZML entity. These entities could be streamed to the client separately. The packets then make up the CZML document. Another approach is to write the CZML response ready and send as a single JSON file to the client. In our case the problem with complex entities with a large set of temporal data becomes an issue. The streaming works well for small entities in large numbers. However, with large entities in moderate numbers, the added value of streaming CZML data diminishes. Thus, we used the CZML files in a traditional manner without streaming.

The high-abstraction CZML approach for LAM data removes the need for specifically parsing data and creating Cesium entities on the client. However, it may present a case of data redundancy due to its data structures. For animated geometries, the CZML structure may require the repetition of geographic coordinates for each timestamp-value pair, even when, the coordinates are static. This can lead to up to 50% redundancy in the data transferred over network. A solution could be to describe the geometry and timestamp-value dataset separately. However, this would somewhat negate the positive effects of higher abstraction of CZML. This is because the separated dataset needs to be processed

client-side, or a custom data format, not CZML, would have to be used. Thus, for performance centric applications, it may be recommended to use the lower level functionality of Cesium or define own JSON data formats.

5.2.7 Cesium visualization data flows

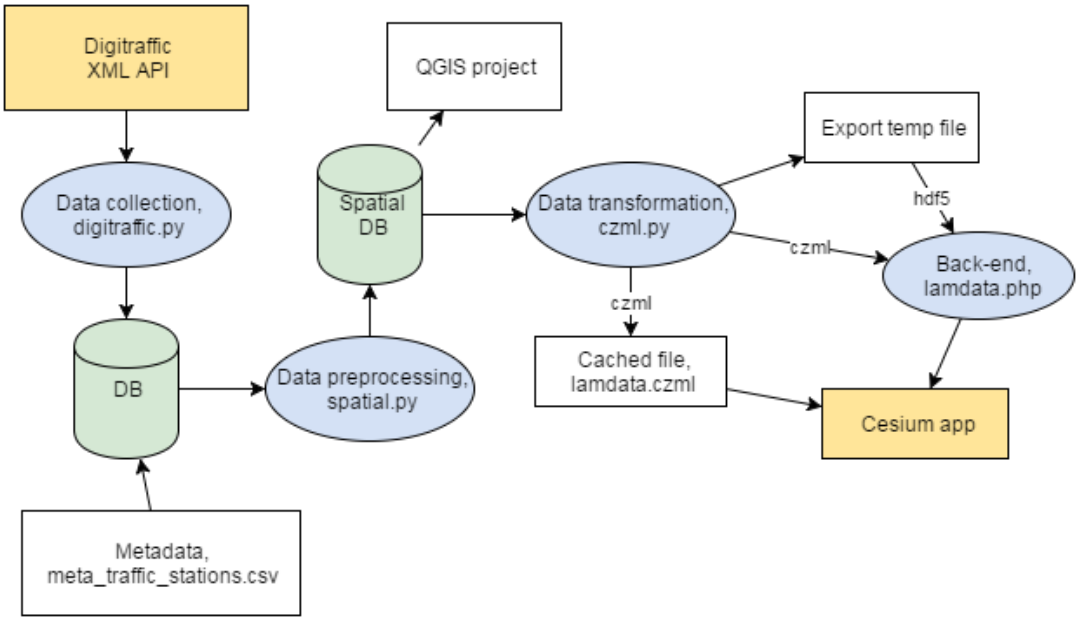


Figure 23. LAM visualization data flow diagram

In Figure 23 we present the data flow diagram for the LAM data visualization. On closer inspection, the data flows of this visualization seem to follow quite closely the first steps of the KDD process. The steps would be of course the preprocessing and the data transformation steps. The data flow diagram describes the path the data takes; through which processes, and what are the inputs and outputs. The inputs are the current, real-time data from Digitraffic API, and the supporting metadata. These are saved into the database by reading CSV files, and using the automatic data collection scripts. The outputs are the dataset for the Cesium app, and the app itself.

The data flow diagram shows two databases. The tables in these could very well be under the same database in practice. For illustrative purposes, as for the historical reasons how this thesis work was conducted, they are separated. However, in our work nothing prevents us from saving the data into the same database if it supports spatial data.

The data preprocessing is conducted after collecting the data. In this phase, simple aggregates over the data are calculated, the line geometries are created, and the aggregation of data into 15-minute time slots is conducted. The line geometry and the spatial database were explained earlier in this thesis. Creation of the 15-minute time slots is important for having a clean and continuous dataset for visualization as well as statistical analysis done in MATLAB as discussed in Chapter 4.

The preprocessed data in the spatial database is ready for viewing. A QGIS project reads the data straight from the database using the Spatialite bindings in QGIS. At this point the spatial database needs to have the line geometry meant for the QGIS visualization. The cleaned-up data is the same as it is for the Cesium app. The data is further processed for viewing it in the web. For this, a language created for displaying temporal and spatial data is used. This language is CZML, a JSON derivative. CZML describes entities and geometries used in Cesium applications. CZML is also extensible. A set of aggregate values, and other data is stored as custom data into the CZML response. This custom data, transferred inside the CZML response, is parsed by the client and displayed in the linked 2D charts.

The data transformation process also generates the files for export. For providing such a multidimensional dataset as a single download, a hierarchical data format was used. HDF5 is a binary data format for storing large amounts of hierarchical data in one file. HDF5 are generated from the data query parameters and the corresponding dataset is returned. As we are writing files, a temporary file is written in the server. This file is later sent to the client and cleaned by the server back-end implementation. In this web-based visualization we are transferring responses and files over HTTP protocol which requires a type of client-server architecture. The server back-end is responsible for handling the request and the query parameters. It will then forward the request to the data transformation script which, in turn, will read the data from the database and return a CZML response or a link to the generated HDF5 resource.

The use of CZML in this visualization means that the client is rather lightweight when compared to the events visualization in Cesium for example. There is no data parsing, or extraneous logic on the client-side as there are in the events visualization. Only the custom embedded data for the linked views need to be explicitly processed on the client.

5.3 Digitrafficview – A common Cesium implementation for visualizations

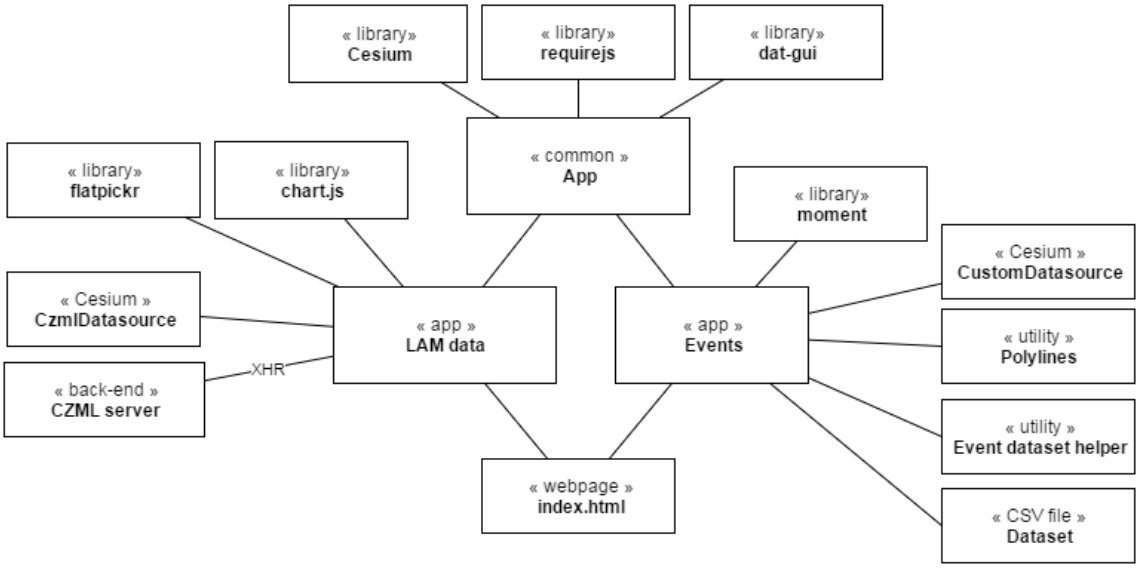


Figure 24. Digitrafficview architecture, libraries, and generic components.

A common web framework, Cesium, was used in both the events and LAM data visualizations. The common app is mainly composed of three components. Cesium provides the utility for defining and displaying spatial entities on a map plane. RequireJS helps with file and module loading. It provides the dependency handling for the numerous source files and is responsible for downloading CSV files from the remote server. Dat-gui module works as a simple GUI (Graphical User Interface) library for controlling the view and recording user input.

The common app is derived for use in specific visualization, namely events and LAM data. These visualizations require further functionality in addition to the generic app. LAM data implementation uses libraries for date picking in the GUI and a library for displaying 2D charts over the Cesium view. The events visualization includes a library for handling time formats and conversions. This allows human-readable time display for travel times for example. It also has custom implementations for parsing CSV files and processing the data into Cesium entities through Polylines and event dataset helper utilities.

Cesium framework has several levels of abstraction (Amato, 2012). LAM data visualization uses the highest-level abstraction, namely, CZML files. The Cesium entities are written in Cesium readable format already on the server-side. The data is then read in the application by using *CzmlDatasource* implementation. This is extended to read the custom data attributes included in the CZML file. Another way to create data sources in Cesium is to use *CustomDatasource* class. This was used in the events visualization. The class is a container for Cesium entities. These entities are created and added to the *CustomDatasource* instance after parsing the CSV files. Effectively the two apps use the Cesium framework on two different levels of abstraction. One will just read CZML described entities created server-side, and the other creates entities on the client.

This chapter looked into describing data sources and flows, conducting a preliminary visual analysis in QGIS, and finally, the development of visualization tools in two different cases. These both cases have a type of data suitable for visualizing in a similar way – spatially. The data sources differ in complexity and size. LAM data is a vast, multidimensional dataset. The events dataset is simple in comparison. This is also apparent from the methods used to visualize them. The simple, small dataset is considerably easier to visualize using CSV files and client-side processing. The LAM data required to take advantage of server-side processing and higher abstraction provided by CZML approach. However, the simple data could also be visualized using CZML abstraction, for example, in a unified visualization tool which would not require specific CSV processing in the client.

In addition to Cesium based visualizations the use of QGIS in creating these visualizations should not be understated. In both cases QGIS provided the first look on data and gave a sandbox for testing ideas about visualizations. The outputs from QGIS may not be as refined as the later web-based tools but it still functioned as an important analysis tool for preliminary exploration of the datasets.

6 Discussion

The related literature presented in this thesis studied information visualization and human cognition. Information visualizations were categorized and the basic techniques associated with visualizations were identified. We noted that, most importantly, the data type drives the type of the visualization used. Also, interactivity is a key part of a visualization that allows visualizing complex datasets.

Many forms of interactivity exist and these sort of interactive as well as non-interactive techniques in visualizations are often studied empirically. The visualization techniques are evaluated in respect to the limitations of human cognition. The cognitive load of visualization can be measured and compared to the capabilities of human perception and the cognitive system. The studies can have in-depth analysis of the human working memory when conducting a visual analysis task. These sorts of studies are incredibly valuable but can alienate us from the actual difficulties in creating useful visualization artefacts. Visualizations are ideal for presentational purposes and the best presentations may often be the simplest and most easily understood pieces of self-explanatory information, abstractions, and visual cues. Moreover, creating visualizations includes many other concepts such as data management, considering the best practices in visualizing, accessibility, and acknowledging difficulties in learning. Understanding the validity of the data, and ensuring the interpretability of the visualization are in the core of useful visualizations.

Visualization guidelines, such as Shneiderman's mantra (Shneiderman, 1996) and the *matching of mental model*, convey the best practices in visualization and they tell how users expect the visualization to work and what features are considered to go in sync with our perception system. However, knowing the best practices does not necessarily help with qualities that require domain knowledge from the analyst such as evaluating the validity of data. Without domain knowledge, it can be difficult to understand what part of data might be outliers and what might be actual correct data. Domain knowledge and the understanding of how the data is collected and used is immensely important to ensure the validity of the data, and that the analyst can create a visualization that is interpreted

correctly. A poor visualization can easily tell a false story about data if the reader is not vigilant.

An interactive visualization is a graphical user interface. Hence, the same principles can be applied as in UX (User Experience) and UI (User Interface) design. For example, a common topic among these is accessibility. Accessibility means providing features which help users with disabilities to interact with the interface. As users have different cognitive abilities, they can also have various visual or motor impairments. As color is important technique in visualizations, software with visualization features may have their default color scale designed such that it can be perceived by colorblind users (blue to yellow instead of red to blue) and that it accurately displays the changes in values visualized using a color scale.

As accessibility, usability is an important concept to consider in visualizations. An interactive visualization has little value if it is not user-friendly or the users will not want to interact with it. Similarly, the learning curve of the visualization should be considered. A visualization meant to supplement a news article needs to be very easily adoptable since the reader may not be expected to spend more than a few minutes reading the article. In comparison, a visualization for the reactor of a nuclear power plant might be allowed to have a steeper learning curve. It is essential to consider how long it takes to learn to use the visualization tool if the tool is only to be used briefly. No matter how complex the visualization product is, it should have at least a minimal guide or tutorial attached to it from very early on. Probably the best way to teach a visualization is through visual means. This means recording a video or by building tutorial elements within the visualization.

The visualization work in this thesis included many forms of data management. Data was stored in CSV files, hierarchical binary files, movable databases, and spatial databases. CSV files are the simplest format of storing data. They are supported by most tools and often provide the means to transfer data when making analysis of a dataset. However, they lack the speed and utility of databases when the amount of data reaches a certain point. Even for small datasets it is sometimes useful to utilize the functionality provided by database engines. Thus, a database centric working is recommended when creating interactive web-based visualizations. Also, for QGIS, complex datasets benefit from having the data in a database. However, some applications require CSV files to act as an

intermediate since they do not support reading from the database. This mangling of files can feel as a burden for the analyst. Although, popular tools, such as MATLAB, are beginning to support increasingly other file formats and hopefully make utilizing different database engines easier for cross-application work in the future.

The experiences from this thesis work show that analysis tools, such as Excel or QGIS, are not always ideal tools for visualization. A spreadsheet filled with values and perhaps a few somewhat descriptive header cells can be difficult to interpret. Information such as the purpose of a data vector, how it was collected, or what might cause outliers cannot be described in a single header. At this point, even if the values are colorized to provide some visualization, the Excel sheet is just a container for data. Plotting that data in graph makes it a visualization but if the context is not apparent from the graph, it still requires other knowledge to be able to be interpreted. A good visualization has this presentational aspect considered.

Presentation includes context. The context can be conveyed in several ways such as showing examples or by teaching the history of something. However, the context can also be inferred from spatial surroundings. This technique is used by many spatial visualizations and are tied to the famous mantra: *"Overview first, zoom and filter, then details-on-demand"* (Shneiderman, 1996). By following this design, the visualization always gives the overview first, and thus, the context to the zoomed-in entity. In this way, the user is not lost in the vastness of information. This idea can be taken further. LAM data visualization does have the spatial overview by using the map. It could also have the overview on the data where the data is aggregated over a time period. This aggregated, calculated value could function as an additional layer of overview. Aggregated values could function as an overview for more detailed time-domain data. This, by following the generic design guideline, would introduce more levels of detail to ease the cognitive task presented by this vast dataset.

7 Conclusions

The role of visualization is to offer new perspectives into an acknowledged challenge, and to offer a kind of “Aha!” insight into data. This leads to requirements for further improvements in the visualization and, thus, increases the understanding of the problem. This understanding is the basis for traditional data analysis where these observations can be verified. The verified knowledge then serves as a base for action taking.

Visualizations are interesting because they tie technical persons and business persons together a sort of automatically. A well-done visualization is, at its best, a simple to understand, informative, and even interactive. Instead of trying to explain correlation of sales using p-values to executives, visualization can do it effortlessly. However, visualization is not without dangers as correlation does not always mean causality. A phrase familiar from data mining applies to visualizations as well. The possibility of making false assumptions from visualizations must be recognized as even simple visualizations can lie.

The development process of a data-driven visualization tool has much in common with data mining processes. It has a similar pipeline from data selection through preprocessing to the interpretation of results. The process for developing a visualization tool is iterative, and the creation of these tools requires many side steps to discover what works and what is actually the thing we need to visualize. The visualization process returned to previous steps several times when the data was required in other formats for the next visualization. For example, the QGIS visualization required data preprocessed and transformed differently than the dataset used in Cesium visualization.

We noted that designing visualizations requires visual demonstrations of tools to understand what is ultimately required from the tool. The selection of tools is important. For vast, multi-dimensional, and spatio-temporal data, it was easily justified to use our own web-based visualization tool created specifically for the purpose. The ready-made tools available just did not offer the kind of flexibility and interactivity that was required from the visualization. The first 2D and 3D visualization created were inadequate in terms of their interactivity and ease-of-use. The following 3D visualizations demonstrated the value of the added dimension and the robustness of a web-based visualization.

We see that the approaches used in this thesis provide effective means for creating insights for understanding traffic patterns, and promoting suitable travel chains to events in public transportation.

7.1 Evaluation of artefacts

As noted earlier in the Chapter 2, the evaluation of visualizations can be difficult. Evaluation of the output, insights, is not a trivial task. However, we can look at the features and the usability of the tools created critically. We can compare the visualization artefacts to the taxonomies and best practices found from the literature. This section considers mainly the Cesium visualizations.

We can conclude that the both Cesium visualizations created support the taxonomy by Shneiderman at least to the width presented in his mantra. They both provide overview of the data, allow adjusting of the view, zooming, and lastly, they give details of an entity on demand. Additionally, filtering of data and exports are possible in both visualizations to some extent. The taxonomy tasks not implemented are *relate* and *history*. Neither of the visualizations support saving the state and going back a step. This sort of feature would potentially be difficult to implement due to the used framework, Cesium. Relate task is not supported either. For example, relationships between the entities in the visualizations cannot be shown in a way that selecting one entity would display or highlight other entities with similar properties. In the case of events visualization this could be implemented by highlighting events with similar travel times. Yet, for traffic data visualization this is much harder to define.

We argue that the usability of the tools is good in general, especially if we compare to the previous versions of the visualization prototypes using other methods. Cesium is a proven open-source framework and works well on most internet browsers. The use of WebGL, however, puts requirements on the user's graphics cards. Fast 3D rendering requires performance that many older laptops may not have. Even though the user interface is rather simple it has navigation features that are not very intuitive for laptop users with a touchpad. Navigation is done using by zooming, panning, and rotation around a point. Cesium 3D scene can be easy to navigate around using a mouse but touch-based interfaces may require more practice and feel less intuitive.

In addition to a complex 3D scene to navigate in, the visualizations have features to query, filter, and modify data in ways that may not be self-descriptive. Therefore, it was required to write how-to pages for describing the features. Although, the tutorial exists, it does not inhibit users from exploring the tool themselves. After all, the tools present an overview on the first launch that can be navigated even without a complete understanding of the tool's capabilities. It was noted that these sort of potential usability issues are of high-concern in a large organization that may not be willing to spend a lot of resources in training the tool for a large pool of employees. Thus, our findings suggest that even the simplest of tools benefit from having a minimal and easy-to-understand how-to guide or tutorial for the tool as early as possible.

The traffic visualization uses animation for displaying temporal data. Based on the cognitive study (Robertson et al., 2008) animation can be problematic for doing analysis. Hence, in LAM data visualization it is recommended to pause the animation occasionally and control the time manually. However, this does not diminish the positive effect of temporal element in the visualization. It is very effective for presentation of data, and the time can be adjusted to run at different speeds. A little less controversial aspect of the traffic visualization is linked views. LAM visualization implements several linked views which were seen very beneficial for creating understanding in various studies.

7.2 Future work

The created visualization artefacts could be improved on many aspects. One of the most valuable additions would be to add a similar daily aggregated data, as in Chapter 4, to the traffic visualization. This would create another layer besides the 15-minute interval data. Integrated daily traffic volumes would serve as a more high-level overview. It would allow to see dates that are different or interesting, and after that the 15-minute data could be further analyzed for that day. Moreover, it adds information content that is difficult to perceive from the continuous 15-minute data. One could recognize a rough estimate of the total daily traffic amount from the current visualization but it not as intuitive as having that as an accurate calculated value which is then visualized properly. This presents us a rather simple case where this arduous cognitive task is made considerable easier by using data mining and visualization.

Currently, the traffic visualization works by querying data from the server based on user inputted parameters. This approach functions well for the case where the user knows the dates that might have interesting data. The data could also be browsed in continuous fashion with functions getting data for the next or the previous day. The feature would add continuity to the user experience. However, we do not see this feature as crucial as showing daily aggregated total traffic amounts which would function as an overview for selecting a day for closer visualization. Based on this experience, we conclude that a dataset as complex as LAM data requires more than one layer of abstraction.

Another way to improve the visualization would be to add support for near-real-time analysis and traffic forecasting. The underlying data is already served in 5-minute intervals. Using the traffic flow forecasting method presented in Chapter 4, the visualization could be made to predict the future traffic. The visualization, however, would be more of a static version as the data does not change so often to make it an animation. This design direction would require a major redesign of the application.

There are a lot of other improvements that could be done. For one, we have the evaluation that shows a few ways to potentially improve the cognitive features. Other functions could be to enable modifying the magnitude parameter on the client, enable download of specific datasets by clicking a road, or add more filtering options. It was mentioned earlier that the filtering was implemented to some extent. The filtering could be improved by adding comparative filters such as when traffic volume is greater than a value. Also, we consider boolean “On/Off” switches for data layers and a type of slider GUI elements for adjusting values valuable addition to any kind of spatial data visualization. Another way to increase the value of the visualization could be to add more data sources such as visualizing travelling by trains or bus lines in addition to road traffic. Even mapping ferry traffic could offer more value to the visualization.

References

Amato, M., 2012. Cesium Language (CZML).

Andrienko, N., Andrienko, G., 2006. Exploratory Analysis of Spatial and Temporal Data. Springer-Verlag, Berlin/Heidelberg.

Anscombe, F.J., 1973. Graphs in Statistical Analysis. Am. Stat. Vol. 27, 17–21.

Bastian, M., Heymann, S., Jacomy, M., others, 2009. Gephi: an open source software for exploring and manipulating networks. ICWSM 8, 361–362.

Brachman, R.J., Anand, T., 1994. The Process of Knowledge Discovery in A First Sketch. AAAI Tech. Rep. WS-94-03.

Cios, K.J., Swiniarski, R.W., Pedrycz, W., Kurgan, L.A., 2007. The knowledge discovery process, in: Data Mining. Springer, pp. 9–24.

Dayal, V., 2015. Anscombe's Quartet: Graphs Can Reveal, in: An Introduction to R for Quantitative Economics, SpringerBriefs in Economics. Springer India, pp. 59–63. doi:10.1007/978-81-322-2340-5_9

Etelä-Karjalan liitto, n.d. Etelä-Karjalan Tapahtumakalenteri [WWW Document]. URL http://api.tapahtumat.ekarjala.fi/ (accessed 11.8.16).

European Data Portal, 2015. Creating Value through Open Data - European Data Portal [WWW Document]. Eur. Data Portal. URL https://www.europeandataportal.eu/en/content/creating-value-through-open-data (accessed 10.3.16).

Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., 1996. From data mining to knowledge discovery in databases. AI Mag. 17, 37.

Finland 100 Years, n.d. Suomi Finland 100 - Hae kalenterista [WWW Document]. URL http://suomifinland100.fi/tehdaan-yhdessa/tulevaa-ohjelmaa/hae-kalenterista/ (accessed 11.8.16).

Finnish Transport Agency, 2016a. finnishtransportagency/digitraffic [WWW Document]. GitHub. URL https://github.com/finnishtransportagency/digitraffic (accessed 11.8.16).

Finnish Transport Agency, 2016b. Current data from LAM stations - finnishtransportagency/digitraffic Wiki [WWW Document]. GitHub. URL https://github.com/finnishtransportagency/digitraffic/wiki/Current%20data%20from%20LAM%20stations (accessed 11.8.16).

Finnish Transport Agency, 2015. Presentation - - finnishtransportagency/digitraffic Wiki [WWW Document]. GitHub. URL https://github.com/finnishtransportagency/digitraffic/wiki/Presentation (accessed 11.8.16).

Finnish Transport Agency, n.d. rata.digitraffic.fi [WWW Document]. URL https://rata.digitraffic.fi/api/v1/doc/index.html (accessed 11.8.16a).

Finnish Transport Agency, n.d. For developers | Digitransit [WWW Document]. Digitransit. URL http://digitransit.fi/en/developers/ (accessed 11.8.16b).

Friendly, M., 2008a. Milestones in the history of thematic cartography, statistical graphics, and data visualization. U RL Httpwww Datavis Camilestones.

Friendly, M., 2008b. A Brief History of Data Visualization, in: Handbook of Data Visualization, Springer Handbooks Comp.Statistics. Springer Berlin Heidelberg, pp. 15–56. doi:10.1007/978-3-540-33037-0_2

Goralski, R., Gold, C., 2008. Marine GIS: Progress in 3D visualization for dynamic GIS, in: Headway in Spatial Data Handling. Springer, pp. 401–416.

Grainger, S., Mao, F., Buytaert, W., 2016. Environmental data visualisation for non-scientific contexts: Literature review and design framework. Environ. Model. Softw. 85, 299–318. doi:10.1016/j.envsoft.2016.09.004

Güting, R.H., 1994. An Introduction to Spatial Database Systems. VLDB J. 3, 357–399.

Habtemichael, F.G., Cetin, M., 2016. Short-term traffic flow rate forecasting based on identifying similar traffic patterns. Transp. Res. Part C Emerg. Technol. 66, 61–78. doi:10.1016/j.trc.2015.08.017

Han, J., Kamber, M., Pei, J., 2011. Data Mining: Concepts and Techniques, Third Edition, 3 edition. ed. Morgan Kaufmann, Haryana, India; Burlington, MA.

He, Y., Su, F., Du, Y., Xiao, R., 2010. Web-based spatiotemporal visualization of marine environment data. Chin. J. Oceanol. Limnol. 28, 1086–1094. doi:10.1007/s00343-010-0029-8

Huang, W., Eades, P., Hong, S.-H., 2009. Measuring effectiveness of graph visualizations: A cognitive load perspective. Inf. Vis. 8, 139–152. doi:10.1057/ivs.2009.10

Hunter, S., 2016. Cesium Language (CZML) Guide [WWW Document]. GitHub. URL https://github.com/AnalyticalGraphicsInc/czml-writer/wiki/CZML-Guide (accessed 11.11.16).

IBM, Zikopoulos, P., Eaton, C., 2011. Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data, 1st ed. McGraw-Hill Osborne Media.

Jones, A.S., Horsburgh, J.S., Jackson-Smith, D., Ramírez, M., Flint, C.G., Caraballo, J., 2016. A web-based, interactive visualization tool for social environmental survey data. Environ. Model. Softw. 84, 412–426. doi:10.1016/j.envsoft.2016.07.013

Kaiser, C., Walsh, F., Farmer, C.J.Q., Pozdnoukhov, A., 2010. User-Centric Time-Distance Representation of Road Networks, in: Fabrikant, S.I., Reichenbacher, T., VanKreveld, M., Schlieder, C. (Eds.), Geographic Information Science. Springer-Verlag Berlin, Berlin, pp. 85–99.

Kapler, T., Wright, W., 2005. GeoTime information visualization. Inf. Vis. 4, 136–146. doi:10.1057/palgrave.ivs.9500097

Keim, D.A., 2000. Designing pixel-oriented visualization techniques: Theory and applications. IEEE Trans. Vis. Comput. Graph. 6, 59–78.

Kielman, J., Thomas, J., May, R., 2009. Foundations and Frontiers in Visual Analytics. Inf. Vis. 8, 239–246. doi:10.1057/ivs.2009.25

Knigge, L., Cope, M., 2006. Grounded visualization: integrating the analysis of qualitative and quantitative data through grounded theory and visualization. Environ. Plan. A 38, 2021–2037. doi:10.1068/a37327

Komenda, M., Schwarz, D., 2013. Visual Analytics in Environmental Research: A Survey on Challenges, Methods and Available Tools, in: Hřebíček, J., Schimak, G., Kubásek, M., Rizzoli, A.E. (Eds.), Environmental Software Systems. Fostering Information Sharing, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, pp. 618–629. doi:10.1007/978-3-642-41151-9_58

Kumar, S.V., Vanajakshi, L., 2015. Short-term traffic flow prediction using seasonal ARIMA model with limited input data. Eur. Transp. Res. Rev. 7. doi:10.1007/s12544-015-0170-8

Lohse, G.L., 1997. The role of working memory on graphical information processing. Behav. Inf. Technol. 16, 297–308. doi:10.1080/014492997119707

Min, W., Wynter, L., 2011. Real-time road traffic prediction with spatio-temporal correlations. Transp. Res. Part C Emerg. Technol. 19, 606–616. doi:10.1016/j.trc.2010.10.002

Myatt, G.J., Johnson, W.P., 2009. Making Sense of Data II: A Practical Guide to Data Visualization, Advanced Data Mining Methods, and Applications, 1 edition. ed. Wiley, Hoboken, N.J.

Nazemi, K., Burkhardt, D., Retz, R., Kuijper, A., Kohlhammer, J., 2014. Adaptive Visualization of Linked-Data, in: Bebis, G., Boyle, R., Parvin, B., Koracin, D., McMahan, R., Jerald, J., Zhang, H., Drucker, S.M., Kambhamettu, C., Choubassi, M.E., Deng, Z., Carlson, M. (Eds.), Advances in Visual Computing, Lecture Notes in Computer Science. Springer International Publishing, pp. 872–883. doi:10.1007/978-3-319-14364-4_84

Nielsen, C.B., 2016. Visualization: A Mind–Machine Interface for Discovery. Trends Genet. 32, 73–75.

Oxford Dictionaries, n.d. visualization - definition of visualization in English | Oxford Dictionaries [WWW Document]. Oxf. Dictionaries Engl. URL https://en.oxforddictionaries.com/definition/visualization (accessed 9.29.16a).

Oxford Dictionaries, n.d. information explosion - definition of information explosion in English | Oxford Dictionaries [WWW Document]. Oxf. Dictionaries Engl. URL https://en.oxforddictionaries.com/definition/information_explosion (accessed 9.28.16b).

Pack, M.L., 2010. Visualization in transportation: challenges and opportunities for everyone. IEEE Comput. Graph. Appl. 30, 90–96.

Robertson, G., Fernandez, R., Fisher, D., Lee, B., Stasko, J., 2008. Effectiveness of animation in trend visualization. IEEE Trans. Vis. Comput. Graph. 14, 1325–1332.

Routasuo, N., 2013. Exploratory visualization of inter-organizational networks; The visualization process (Master's Thesis). Aalto University, Espoo.

Sayar, A., Pierce, M., Fox, G., 2006. Integrating AJAX approach into GIS visualization web services, in: Advanced Int'l Conference on Telecommunications and Int'l Conference on Internet and Web Applications and Services (AICT-ICIW'06). IEEE, pp. 169–169.

Schwamborn, A., Thillmann, H., Opfermann, M., Leutner, D., 2011. Cognitive load and instructionally supported learning with provided and learner-generated visualizations. Comput. Hum. Behav., Current Research Topics in Cognitive Load TheoryThird International Cognitive Load Theory Conference 27, 89–93. doi:10.1016/j.chb.2010.05.028

Sedlmair, M., Isenberg, P., Baur, D., Butz, A., 2011. Information visualization evaluation in large companies: Challenges, experiences and recommendations. Inf. Vis. 10, 248–266. doi:10.1177/1473871611413099

Shimabukuro, M.H., Flores, E.F., Oliveira, M.C.F. de, Levkowitz, H., 2004. Coordinated views to assist exploration of spatio-temporal data: a case study, in: Proceedings. Second International Conference on Coordinated and Multiple Views in Exploratory Visualization, 2004. Presented at the Proceedings. Second International Conference on Coordinated and Multiple Views in Exploratory Visualization, 2004., pp. 107–117. doi:10.1109/CMV.2004.1319531

Shneiderman, B., 2002. Inventing discovery tools: combining information visualization with data mining1. Inf. Vis. 1, 5–12.

Shneiderman, B., 1996. The eyes have it: A task by data type taxonomy for information visualizations, in: Visual Languages, 1996. Proceedings., IEEE Symposium on. IEEE, pp. 336–343.

Tory, M., Moller, T., 2004. Human factors in visualization research. IEEE Trans. Vis. Comput. Graph. 10, 72–84.

Viegas, F.B., Wattenberg, M., Van Ham, F., Kriss, J., McKeon, M., 2007. Manyeyes: a site for visualization at internet scale. IEEE Trans. Vis. Comput. Graph. 13, 1121–1128.

Wong, P.C., Thomas, J., 2004. Visual Analytics. IEEE Comput. Graph. Appl. 24, 20–21. doi:10.1109/MCG.2004.39

World Wide Web Consortium, 2016. LinkedData - W3C Wiki [WWW Document]. W3C Wiki. URL https://www.w3.org/wiki/LinkedData (accessed 11.25.16).

Yang, Z., Bing, Q., Lin, C., Yang, N., Mei, D., 2014. Research on Short-Term Traffic Flow Prediction Method Based on Similarity Search of Time Series. Math. Probl. Eng. 2014, 1–8. doi:10.1155/2014/184632

Yi, J.S., Kang, Y., Stasko, J.T., Jacko, J.A., 2008. Understanding and Characterizing Insights: How Do People Gain Insights Using Information Visualization?, in: Proceedings of the 2008 Workshop on BEyond Time and Errors: Novel evaLuation Methods for Information Visualization, BELIV '08. ACM, New York, NY, USA, p. 4:1–4:6. doi:10.1145/1377966.1377971

Zhang, T., Li, J., Liu, Q., Huang, Q., 2016. A cloud-enabled remote visualization tool for time-varying climate data analytics. Environ. Model. Softw. 75, 513–518. doi:10.1016/j.envsoft.2015.10.033

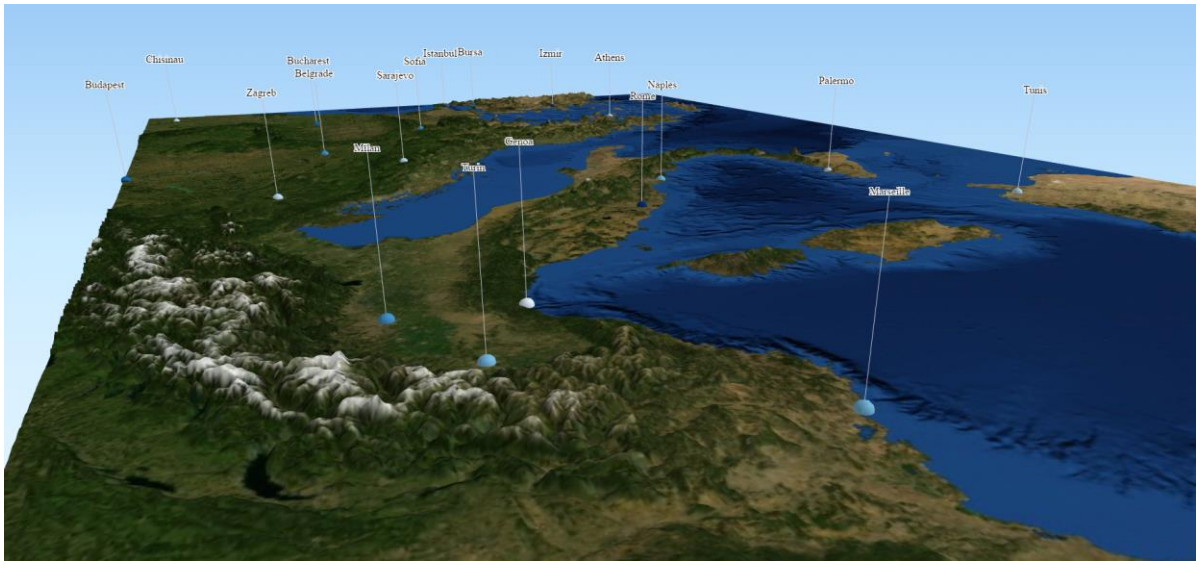
Zoss, A., 2015. Visualization Types - Introduction to Data Visualization - LibGuides at Duke University [WWW Document]. URL http://guides.library.duke.edu/datavis/vis_types (accessed 9.29.16).

Appendix I. Open data sources

Description	Publisher	Update frequency	Log	Location	Protocol	Format
Etelä-karjalan tapahtumakalenteri	Etelä-Karjalan liitto	realtime	0	1	HTTP	JSON/RSS
Matka.fi / Digitransit	Liikennevirasto	realtime		1	HTTP	XML
rata.digitraffic.fi	Liikennevirasto	realtime	1	1	REST/WebSocket	JSON
Digitraffic Sujuva palvelu	Liikennevirasto	1min/5min/24h	0	1	HTTP	XML/CSV
PAAVO tietokanta	Tilastokeskus	yearly	0	1	HTTP	WMS/JSON-stat
StatFin	Tilastokeskus	realtime	1		HTTP	JSON-stat
Eurostat avaintaulukot	Tilastokeskus	realtime	1		HTTP	JSON-stat
Liikenneviraston WMS	Liikennevirasto	realtime	0	1	HTTP	WMS
Liikenneviraston WFS	Liikennevirasto	realtime		1	HTTP	WFS
Liikenneviraston rajoitettu WFS	Liikennevirasto	realtime			HTTP	WFS
Ilmatieteenlaitoksen WMS	FMI	realtime	0	1	HTTP	WMS
Ilmatieteenlaitoksen WMS (säähavainnot)	FMI	realtime	0	1	HTTP	WMS
Ilmatieteenlaitoksen WMS (tutkadata)	FMI	realtime	0	1	HTTP	WMS
Ilmatieteenlaitoksen WFS	FMI	realtime		1	HTTP	WFS
Tilastokeskus Inspire WMS	Tilastokeskus		0	1	HTTP	WMS
Tilastokeskus karttaaineistot WMS/WFS	Tilastokeskus				HTTP	WMS/WFS
Museovirasto WMS	Museovirasto		0	1	HTTP	WMS
Maanmittauslaitos WMS	Maanmittauslaitos		0	1	HTTP	WMS
ULJAS	Tulli	monthly	1		HTTP	JSON,JSON-stat,CSV,XML
Pelastustoimen mediapalvelu	Pelastustoimi	realtime	0	1	HTTP	RSS
GeoNames, paikkatieto ja paikannimet	Unxos GmbH		0	1	HTTP	XML/JSON
Kansalliskirjaston API	Kansalliskirjasto				HTTP	JSON

Appendix II. Qgis2threejs terrain

Digital elevation data (DEM) combined with the satellite map data of Italy and its surroundings. DEM is a type of height map and read as a raster file in QGIS and Qgis2threejs.



Appendix III. Traffic pattern comparison

A comparison between an average weekday and a day at the start of a major holiday is demonstrated below in two figures. On the lower right portion of the figures we see the aggregated traffic volume for the selected time period for each traffic measurement station along Road 4. The first figure shows that the traffic volume peaks are concentrated on major cities. In the second figure note especially Road 4 where the traffic stays up high until a major branch in the road that divides.