City traffic visualization

Dimitrov, V.

Award date:
2014

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science

City Traffic Visualization

Master Thesis

Valcho Dimitrov

Supervisors:
Prof. dr. ir. Jack van Wijk
MSc Mark Stobbe

Eindhoven, October 2014

Abstract

Sooner or later every modern city faces the challenges to cope with increased traffic and street congestions. Increased e-commerce means an increase in the number of delivery trucks driving around. Understanding how a city works can be used by researchers and governments to enforce new policies for regulating the traffic. Because of this, cities are pioneering with creating measuring sites that collect traffic data for further analysis.

NDW is relatively young and fast developing system that collects and stores such data for the major roads around the Nederlands. Unfortunately there still haven't been many applications that can present the data in easy to understand and analyze format. Furthermore, the few products that have been released are aimed to be commercial oriented rather than research helping.

This thesis describes an interactive visualization of NDW traffic data. We designed and built a prototype that does has two main functions. First, it cleans the raw NDW data. We remove the unnecessary information and organize the rest in a more useful to us matter. This gives a vast reduction in used memory storage space. Second, we created various types of visualizations: visualizing traffic data over a geographical image and visualization using line and bar charts. The first type of images allow the exploration of the spatial property. The user can see the traffic situation in a whole city for any period in time. Charts assist the user in understanding the vast amount of data. Multiple controls are provided to change the dimensions such as temporal properties, vehicle class, location, etc. Finally we combine everything in a multi user system.

Acknowledgements

I would like to thank my supervisor Prof. dr. ir. J.J.van Wijk for his guidance during this project. His comments, feedback and countless advices on the prototype and the report helped me to improve my ideas and achieve my goals.

I would also like to thank Mark Stobbe for his help with the technical part of this project. His help with Java development and databases proved to be more than helpful.

I would also like to thank all my friends and colleagues for their feedback on my work and for their help and advices on the design of the images in this paper and the visualizations. Finally I would like to express my gratitude to my family for providing me with the opportunity to come to Technical University of Eindhoven and for their constant support during the time I spent here.

Contents

Chapter 1

Introduction

The rise of e-commerce means an increase of delivery trucks driving around in our cities. Governments are forced to create new policies for regulating the increase in traffic, keeping our cities functional and safe. Improvements to infrastructure and better city planning also play a major role in smart logistics.

Modern cities have many measuring sites for collecting real-time traffic data. Every minute they collect data about traffic flow, realized travel time, estimated travel time, traffic speed and vehicle classes. For smart logistics it is important to understand how the city works. What are the main problems within the city? What is limiting an increase in traffic? What is the effect of new policies such as environmental areas, delivery time windows, etc.?

The huge amount of data collected every minute poses a challenge to the experts as it becomes more and more difficult to analyze it and extract the useful information. Meanwhile, due to the increasing number of measuring sites and upgraded hardware, the collected data becomes more complex to be analyzed.

Nationale Databank Wegverkeersgegevens (NDW) collects such data, summarizes it and presents it to researchers to investigate. Unfortunately, it is relatively young service and its limited user group slows down the development of software applications which make use of the data. Furthermore, there is no universal method to read and understand the traffic as different users are interested in different properties of the data.

In this project we aim to provide several visualization techniques that visualize the vast amount of city traffic data, which can be used to see how different types of policies would affect the traffic flows of both people and freight.

This thesis is structured in the following way. In Chapter 2 we analyze the problem and set the main goals for this project. Chapter 3 presents some solutions that have been developed in the area of visualizing data on a geographical image and a discussion of some existing applications dealing with traffic data. Chapter 4 describes the shape and format of NDW data and how we process it to suit our needs. Chapter 5 introduces our visualization solutions and gives details about choices made. Finally we combine everything in one system (Chapter 6).

Chapter 2

Visualization Requirements

The current project is intended to support the researchers from the "Smart Logistics" Research Lab in Operations, Planning, Accounting and Control (OPAC) group at Eindhoven University of Technology. As stated in the name, the OPAC group is focusing on the planning and control of operational processes, both within the manufacturing and service sectors. These include:

1. Manufacturing Sector: Production control, distribution, transportation, warehousing, and retail processes;

2. Service Sector: Maintenance management, health care, and public transport.

Smart Logistics focuses on the efficiency and effective alignment of planning and scheduling of ICT infrastructure, people, and governmental policymaking [21]. Smart Logistics equals **3P+I** (i.e., Planning, People, Policy and Infrastructure), and is the synchronized interplay of these four key domains.

The Transportation Chain consists of three parts: the Pickup Chain (or the first mile), the Transit Chain, and the Delivery Chain (or the last mile). The Smart Logistics Labs research is completely embedded in the Transportation Chain, working on topics related to separate parts of the chain and on the integration of the separate parts. The challenges in the Transportation Chain are to efficiently handle increased uncertainty and high complexity. This is important since the real-life world does not fit into a deterministic and static straitjacket, which is assumed by many published models and industry tools. Any decision, action, plan or schedule built on unrealistic assumptions is bound to be less than optimal once realized. Understanding how the city works in reality can be a key factor in producing better and better solutions to transportation problems. The road network of the city is its arteries. It is important to know its condition at any time in order to maintain it, optimize and expand it, and use its full potential. For that it is important for a researcher to be able to answer the following questions.

2.1 Objectives

In this thesis we try to visualize NDW traffic data and develop an application, with the help of which users can answer questions as:

1. Where are vehicles entering the city?

2. Where are vehicles leaving the city?
3. Where are vehicles traveling around in the city?
4. How many vehicles are going in and out of the city at any time?
5. Is there any difference in the number of vehicles and their trajectories between work days and weekends, months, seasons?
6. What is the difference in the behavior of different types of vehicles?
7. Are there any exceptions from the "normal" behavior and can there be identified?
8. How does the traffic situation looks like at some time?
9. When is all this happening?

2.2 Requirements

Based on the description of the project we have the following technical requirements:

R1: The application should take as an input traffic data and produce as an output visualization images.

R2: The application should work with historical traffic data provided by Nationale Databank Wegverkeersgegevens (NDW).

R3: The application should provide a view of the data over a geographical map.

Analyzing the research questions above revealed several more requirements. To answer the questions *where* and *when*:

R4: The application should provide temporal and spatial views of the data.

R5: The application should present the data with respect to time and space.

Furthermore, we can identify also the need to:

R6: The application should present the traffic for a city level.

R7: The data should be presented with the respect to vehicle class.

R8: User should be able to identify different type of traffic (e.g. incoming traffic, outcoming traffic, city center traffic, etc.).

R9: The application should provide a momentary view of the traffic situation.

R10: User should be able to compare traffic data with the respect to time, vehicle class and location.

R11: The application has to be interactive, allowing the user to make the selections mentioned above.

R12: The application should not be very complicated to use. Users should spend no more than 30 minutes learning how to use it.

R13: The produced visualization images should be estetically pleasing.

2.3 Users

As we mentioned above, the current project has as a goal to develop a useful tool to help the researchers at the Smart Logistics lab in their work. This includes professors, assistant professors, PhD and MSc students. All of them are working in the academic field and have background knowledge in the field of logistics and transportation, and at least basic skills in working with computer. Users can be expected to have experience in reading maps and analytical skills in reading and understanding simple charts. The group of possible users can be extended to include other researchers from the OPAC group and other users from industry interested in historical traffic data, but they should own at least the skills already mentioned.

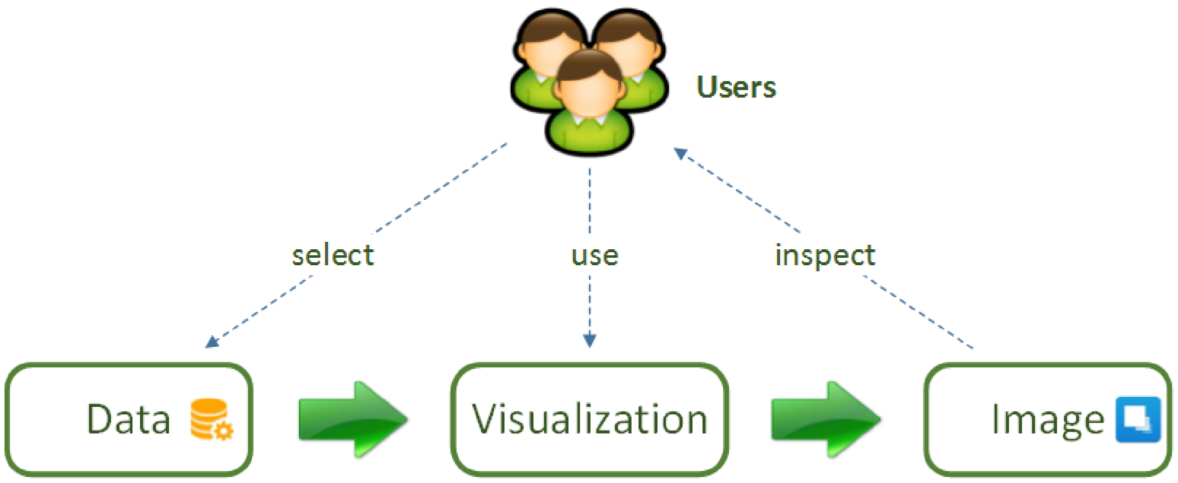


Figure 2.1: Basic workflow for *City Traffic Visualization* application

Chapter 3

Related Work

We are interested in showing traffic data, taking both its geospatial and temporal properties into account. Route maps are one of the most common forms of presenting geospatial data. Cartography has been around since the dawn of human civilization. In the past several decades lots of work has been done to improve our maps and to be able to fit more and various information. Unfortunately, most of the work that has been done in the field of visualizing traffic data on a map is in the case where the data itself consists of object trajectories. Nevertheless some of the visualization techniques and recommendations that have been developed can be employed in other projects.

Visualizing the spatial component of the data is usually done by presenting the information on the map, on the locations where the event for which we have measurements is happening. This can include drawing the route of a vehicle or showing the average vehicle speed at the point where a speed camera is located. Most traditional for the visualization of traffic movement are arrows or flow lines drawn on a map or image (Figure 3.1).

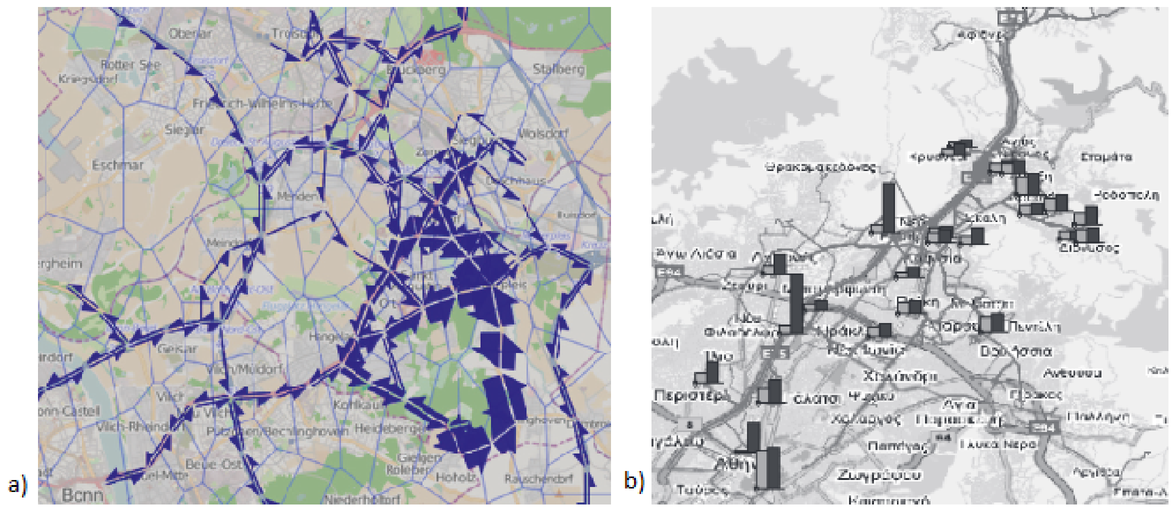


Figure 3.1: Examples of using arrows and bars to present information on a map. On image a) arrows are used to display the direction of the movement of a single vehicle. The thickness of the arrows is proportional to the number of times the vehicle passed through that stretch. On image b) bars are used to indicate the minimum and median times spent in different places during the trips of the trucks of a delivery company. Images are taken from [8] and [5].

Irrespective of the spatial component, data that changes over time can present significant challenges to visualization and analysis. Currently, map animation has become the standard approach to portraying time dependent data and dynamic phenomena [5]. Another very popular technique is presenting multiple views of the same map region over various time moments (Figure 3.2). Andrienko [4] argues that although these techniques can prove to be very useful in showing the time variances in the data, it is still difficult to consider all aspects of the dimension of time in a single visualization. To help with this, interactive tools which show the data in charts with linear time axis are often used. The issue with them is that they can address only one specific case at a time. Stephen Few has done lots of work in the field of creating and improving charts([13] [14] [11] [12]). Many of his advices are used in this thesis, more details are given in Chapter 5.

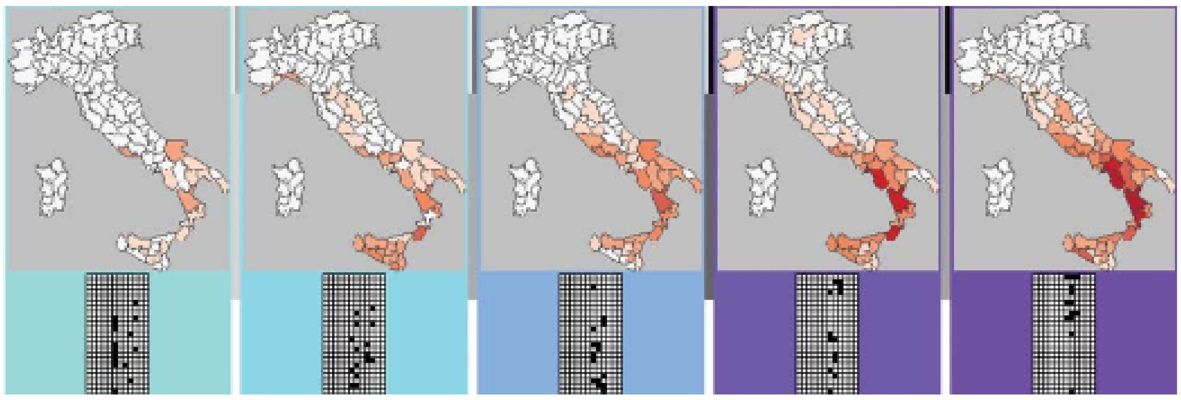


Figure 3.2: Spatial distribution of forest fires for the same period of the year for the span of 25 years. Each image aggregates data for the period of 5 years. A darker red color indicates larger number of forest fires in that region. Images can be arranged in a 2D matrix like shown here or used to create an animation. Image taken from [4].

In real life scenarios it is often necessary to visualize large data sets. Simply showing everything on the map will produce messy and practically unreadable images. The main methods used to deal with large data sets include data aggregation and summarization; applying more sophisticated computational techniques, such as data mining; and developing projections of the data that move items away from their geographic location to fill graphic space more efficiently (Figures 3.2 and 3.3).

Apart from the academic work, there are several commercial products available around the Internet that aim at visualization of traffic data. One of the most popular ones is the Google Live Traffic feature of Google Maps[1]. The live traffic uses colors to show the relative speed of the vehicles on the particular stretch in the city (Figure 3.4). The color used for some road is selected after comparing the average speed of the vehicles to the speed limit there. Green color means that the traffic speed is closer to the limit and red mean that the speed is much lower than it. Unfortunately the application does not present the velocity in numbers which can give you false idea of the traffic situation. Having a central street and a highway in the same color can be interpreted as the average speed of the vehicles is the same on both places although the speed limit of the highway is 120km/h and in the city 50-80km/h.

Google Live Traffic uses data which it collects from devices running Android OS smart-

[1] `https://maps.google.com/`

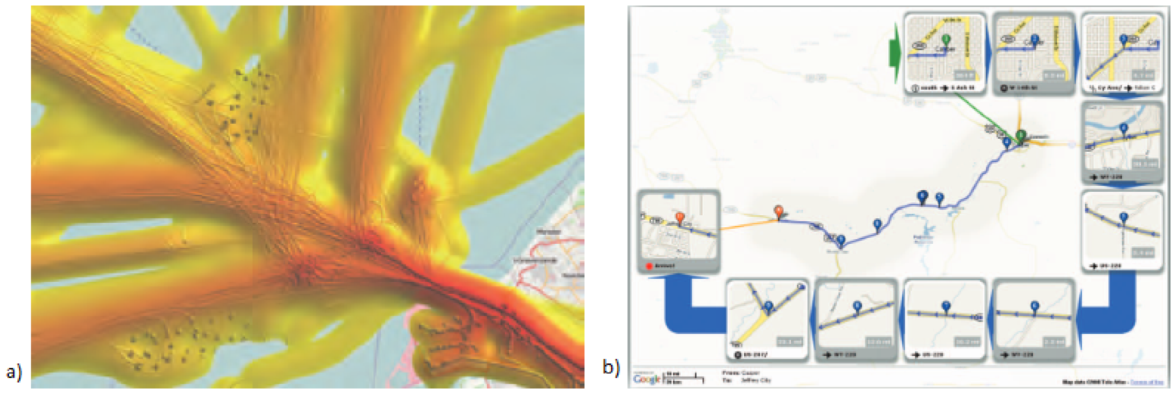


Figure 3.3: a) Aggregation techniques to visualize the vessel density in the area of Rotterdam harbor. The author uses colors and illumination to show the variation of the density levels [23]. b) Detail lenses to present more details about the route. Lenses can be used to display more and varied information like zoomed images or traffic numbers [19].

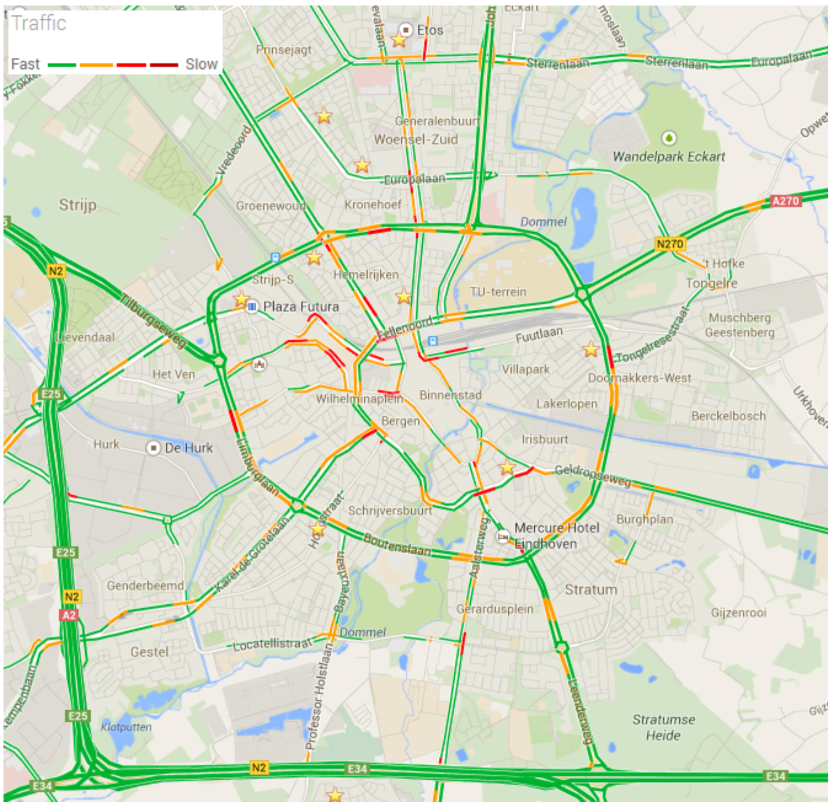


Figure 3.4: Google Live Traffic snapshot of the traffic in the area of Eindhoven. The image shows the traffic speed of the vehicles, relative to the speed limit of the roads.

phones and tablets [16]. This allows it to use a large amount of speed data. Unfortunately, in most of the cases it cannot collect information about the type of vehicle the device is in. This makes it impossible to find the difference between the average speed of small cars and big lorries. The second drawback of using smartphone data is that Google cannot tell how many vehicles have passed through a given point as it has information only about the number of Android devises that has passed. Another limitation back is the color map that is used to draw the traffic lines. It consists of only four colors: green, orange, red and dark red. Although it is intuitive for most users to understand (green means good and hence in our case fast traffic and for red the opposite), the green and the red color seem brighter that the orange and the dark red. Hence initially the user's attention is attracted by them and this can lead to making wrong decisions by simply ignoring or not seeing the orange and dark red streets who have slower traffic. Also the maps provided have large green areas. This can affect the readability if the map, especially when a fast traffic road is going through such areas. One big advantage of Google Live Traffic is that it keeps history of the traffic data and can recreate the traffic speed at any past time upon user request.

Another application that shows traffic congestions is Onderweg[2]. It is developed by one of the NDW partners and uses NDW traffic data. It uses all the available traffic speed data and uses it to create image of the current traffic situation (Figure 3.5).

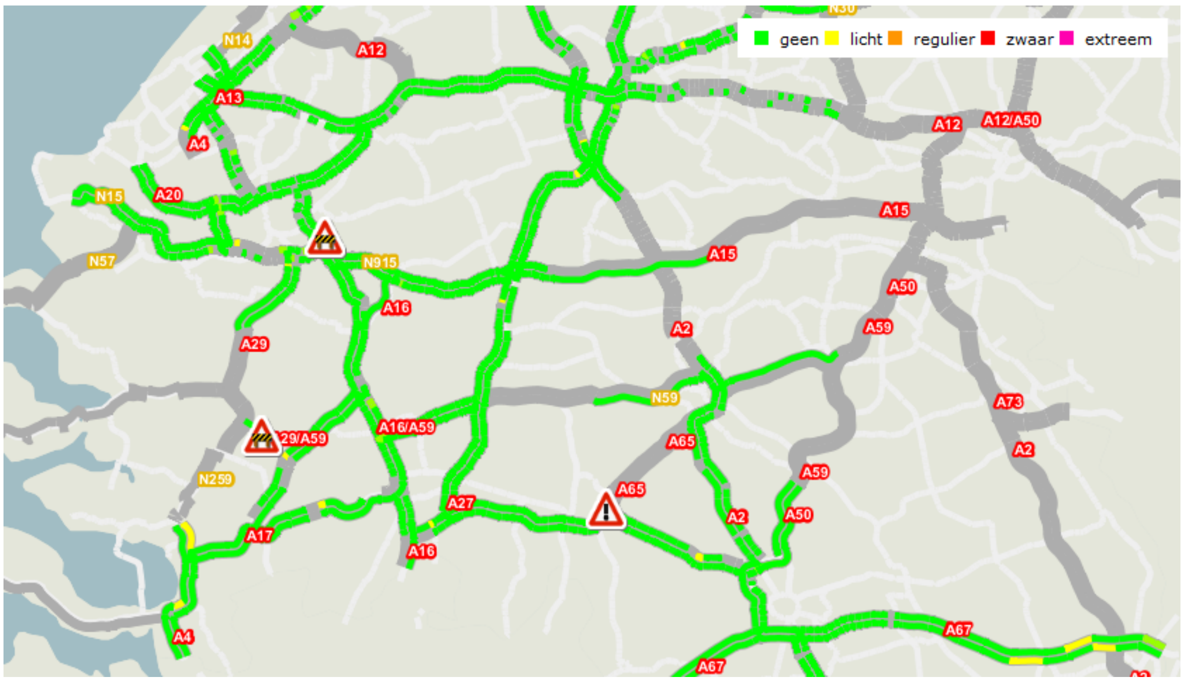


Figure 3.5: Snapshot of the congestions in low-mid part of Netherlands made using Onderweg. The image shows the relative size of the traffic congestions.

Unfortunately, NDW provides very little traffic speed data, which limits the usability of the application. At present, such data is available only for some of the major highways. An advantage of this tool is that it can provide information about current roadworks and hence this information can be used by the user to understand why some roads have congestions. Onderweg suffers from the same shortcomings as Google Live Traffic. Firstly the colormap is badly designed as the yellow color is very bright and at some places it merges with the gray background used. Secondly, even though traffic speed data is available per vehicle class (short vehicles, medium length vehicles, long vehicles, etc.), it seems that the application does not

[2] http://www.onderweg.nl/

make use of it. It does not show information about the total number of passing vehicles, even though this information is also available.

INRIX[3] is another product that aggregates data from several different data sources, including NDW. As a result it can display the current traffic situation (congestions). It has the same traffic visualization features as Google Live Traffic and Onderweg, and also the same problems as described above.

It seems that there is not much work done and no freely available products that make use of traffic data as number of vehicles that are passing through a street or conjunction. This can be either because of the unavailability of such data, but also because of the limited applicability of these tools. The lack of either the data and the tools makes it very difficult for researchers to understand in detail how the city works.

[3] http://www.inrixtraffic.com/

Chapter 4

Data Preprocessing

4.1 Nationale Databank Wegverkeersgegevens (NDW)

De Nationale Databank Wegverkeersgegevens (NDW) or "The National Data Warehouse for Traffic Information" in English, is an organization that was established in 2007 and involves various authorities working closely together to develop and maintain a traffic data database for the road network of the Netherlands. NDW started serving traffic data about the most important roads somewhere in 2009. Since then the databank has grown rapidly and currently it ields more than 200TB of data from 20,400 measurement sites spread over 6,000 kilometers of roads. Some of the main priorities are insuring high quality of the data and uniformity. NDW ensures that all traffic data is entered properly into the historical database. If there are any errors or delays in the real-time traffic data, they are being corrected before entered in the database. Uniformity is achieved by presetting some rules and agreements about definitions and calculation methods. This insures that the data coming from the different partners has the same shape and is easy to use. We have to note that it does not mean that we have 100% error free and available data.

NDW provides two types of information: traffic information and status information. Each measuring site collects data every minute and then made available to the users in around 75 seconds. Traffic information includes:

- Traffic flow: the number of vehicles that pass a measurement site within a certain period of time;
- Average speed;
- Realized or estimated travel time;
- Vehicle class - derived from the length of the passing vehicles.

All the accumulated real-time data is saved in the historic record and is thus is available for later use and consultation.

The status information refers to the availability of a certain road:

- Road works and event-related traffic measures on virtually all the roads in the Netherlands;
- Reports of congestion, accidents and incidents on all the national roads;

- Safety-related announcements (such as a wrong-way driver) that are issued by the traffic control centers;
- The status (open/closed) of bridges;
- The status (open/closed) of peak and regular lanes.

The database is primarily intended for the NDW partners themselves. Other parties that want to access the historical data need to acquire a license which can happen after submitting a request to NDW for this.

4.2 Traffic Data

This master thesis was done using the data that was available with the university license. This allows us to access only the historical traffic information and hence the developed application is making use only of this data.

As we mentioned above, the data is collected on the specific measuring locations of the road network. Hence the data which is available for download is grouped by measuring locations. NDW has a very well organized user interface which allows you to request only the data which is of interest to the user. This can later save you processing time and storage space for the data itself. The most important selection filters which can be used by the user are:

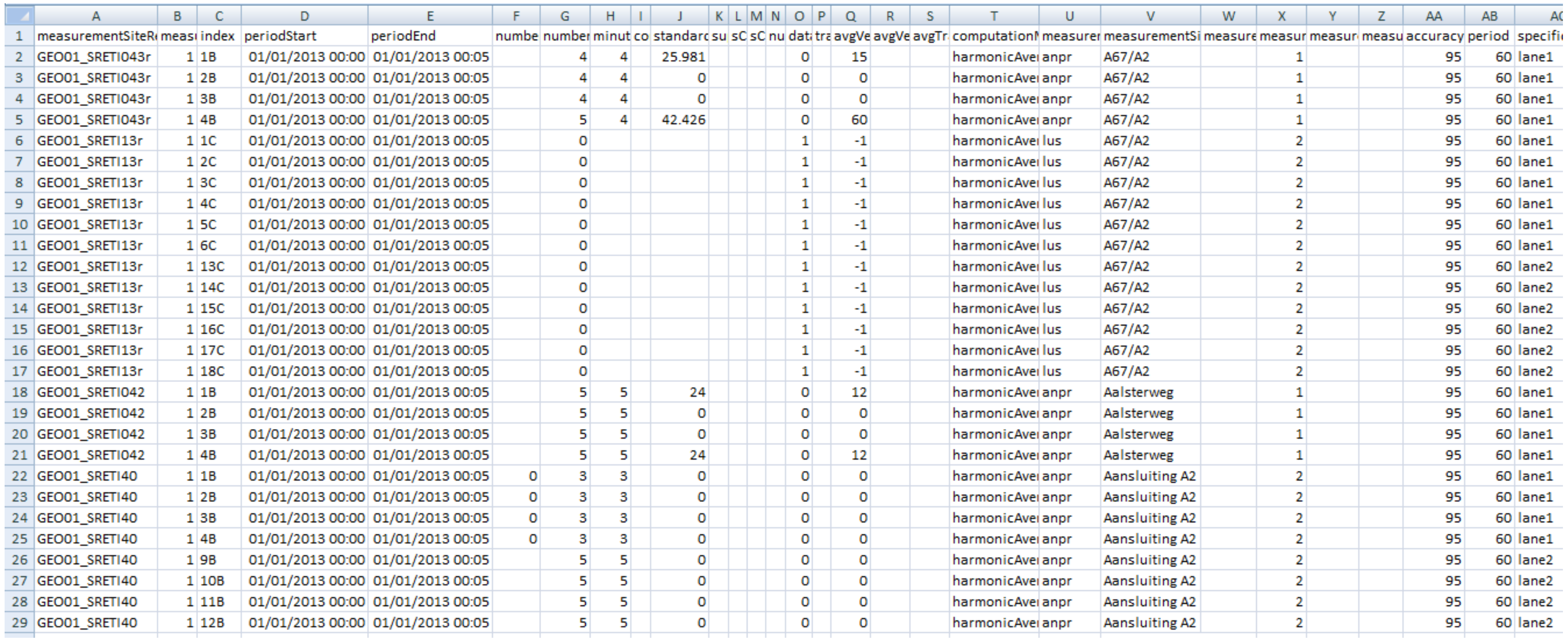
- Select the locations for which you need information;
- Select the time period for which you need the information;
- Select which type of information you need: intensity of the traffic, average speed of the vehicles and average travel time (or any combination of these);
- Filter based on the completeness (availability) of the measurements.

For a full description of the user interface consult the ”Handleiding Historische gegevens” available on the NDW website [1].

The selected data is available for download in *Comma Separated Values (.csv)* files. This allows us to open them with a text editor or spreadsheet application (like Excel or similar) and explore the content (see Figure 4.1).

We can see that the provided data has several attributes (Table 4.1). The fields that are highlighted in blue represent the meta data for each measuring location, showing the physical location of each measuring point and also the measuring equipment that is used. This meta data stays constant, it does not change with time and does not depend on the result of each measurement. Each location has a unique name (*measurementSiteReference*). If some measuring location has to be moved, NDW does not correct the location fields (from V to AV), but instead creates a new entry in their database with a new unique name and new updated meta data. Hence we assume that once one measuring location is put into the database, its meta data does not change. In this assumption we exclude the case where the entry is edited because of some technical mistakes, like a spelling error or wrong data in the field.

[1] http://www.ndw.nu/pagina/nl/4/databank/65/historische_gegevens/



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC
1	measurementSiteR	meas	index	periodStart	periodEnd	numbe	number	minut	co	standarc	su	sC	sC	nu	dat	tra	avgVe	avgVe	avgTr	computationM	measurer	measurementSi	measure	measur	measur	measu	accuracy	period	specifi
2	GEO01_SRETI043r	1	1B	01/01/2013 00:00	01/01/2013 00:05		4	4		25.981					0		15			harmonicAve	anpr	A67/A2		1			95	60	lane1
3	GEO01_SRETI043r	1	2B	01/01/2013 00:00	01/01/2013 00:05		4	4		0					0		0			harmonicAve	anpr	A67/A2		1			95	60	lane1
4	GEO01_SRETI043r	1	3B	01/01/2013 00:00	01/01/2013 00:05		4	4		0					0		0			harmonicAve	anpr	A67/A2		1			95	60	lane1
5	GEO01_SRETI043r	1	4B	01/01/2013 00:00	01/01/2013 00:05		5	4		42.426					0		60			harmonicAve	anpr	A67/A2		1			95	60	lane1
6	GEO01_SRETI13r	1	1C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane1
7	GEO01_SRETI13r	1	2C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane1
8	GEO01_SRETI13r	1	3C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane1
9	GEO01_SRETI13r	1	4C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane1
10	GEO01_SRETI13r	1	5C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane1
11	GEO01_SRETI13r	1	6C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane1
12	GEO01_SRETI13r	1	13C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane2
13	GEO01_SRETI13r	1	14C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane2
14	GEO01_SRETI13r	1	15C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane2
15	GEO01_SRETI13r	1	16C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane2
16	GEO01_SRETI13r	1	17C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane2
17	GEO01_SRETI13r	1	18C	01/01/2013 00:00	01/01/2013 00:05		0								1		-1			harmonicAve	lus	A67/A2		2			95	60	lane2
18	GEO01_SRETI042	1	1B	01/01/2013 00:00	01/01/2013 00:05		5	5		24					0		12			harmonicAve	anpr	Aalsterweg		1			95	60	lane1
19	GEO01_SRETI042	1	2B	01/01/2013 00:00	01/01/2013 00:05		5	5		0					0		0			harmonicAve	anpr	Aalsterweg		1			95	60	lane1
20	GEO01_SRETI042	1	3B	01/01/2013 00:00	01/01/2013 00:05		5	5		0					0		0			harmonicAve	anpr	Aalsterweg		1			95	60	lane1
21	GEO01_SRETI042	1	4B	01/01/2013 00:00	01/01/2013 00:05		5	5		24					0		12			harmonicAve	anpr	Aalsterweg		1			95	60	lane1
22	GEO01_SRETI40	1	1B	01/01/2013 00:00	01/01/2013 00:05	0	3	3		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane1
23	GEO01_SRETI40	1	2B	01/01/2013 00:00	01/01/2013 00:05	0	3	3		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane1
24	GEO01_SRETI40	1	3B	01/01/2013 00:00	01/01/2013 00:05	0	3	3		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane1
25	GEO01_SRETI40	1	4B	01/01/2013 00:00	01/01/2013 00:05	0	3	3		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane1
26	GEO01_SRETI40	1	9B	01/01/2013 00:00	01/01/2013 00:05		5	5		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane2
27	GEO01_SRETI40	1	10B	01/01/2013 00:00	01/01/2013 00:05		5	5		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane2
28	GEO01_SRETI40	1	11B	01/01/2013 00:00	01/01/2013 00:05		5	5		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane2
29	GEO01_SRETI40	1	12B	01/01/2013 00:00	01/01/2013 00:05		5	5		0					0		0			harmonicAve	anpr	Aansluiting A2		2			95	60	lane2

Figure 4.1: An example of real data obtained from NDW and opened in MS Excel. The table contains only traffic intensity data.

excel column	column title	excel column	column title
A	measurementSiteReference	Y	measurementSiteIdentification
B	measurementSiteVersion	Z	measurementSide
C	index	AA	accuracy
D	periodStart	AB	period
E	periodEnd	AC	specificLane
F	numberOfIncompleteInputs	AD	specificVehicleCharacteristics
G	numberOfInputValuesused	AE	startLocatieForDisplayLat
H	minutesUsed	AF	startLocatieForDisplayLong
I	computationalMethod	AG	LocationCountryCode
J	standardDeviation	AH	LocationTableNumber
K	supplierCalculatedDataQuality	AI	LocationTableVersion
L	sCDQ_Low	AJ	alertCDirectionCoded
M	sCDQ_SD	AK	specificLocation
N	number_of_sCDQ	AL	offsetDistance
O	dataError	AM	LOC_TYPE
P	travelTimeType	AN	LOC_DES
Q	avgVehicleFlow	AO	ROADNUMBER
R	avgVehicleSpeed	AP	ROADNAME
S	avgTravelTime	AQ	FIRST_NAME
T	computationMethod	AR	SECND_NAME
U	measurementEquipmentTypeUsed	AS	messageType
V	measurementSiteName1	AT	publicationTime
W	measurementSiteName2	AU	deducedNoTrafficMinutes
X	measurementSiteNumberOfLanes	AV	carriageway

Table 4.1: List of the attributes that can be found in the traffic data files. The columns that represent metadata for each measuring location are shown in blue.

Figure 4.1 shows a short example of the data. By exploring the rows we can see that they are first sorted by measuring period (columns D and E) and after that by measuring location name (column A). The *periodStart (D)* and *periodEnd (E)* columns specify the aggregation period of the measurements. NDW uses this aggregation step to create the measure values (columns Q,R and S). Hence by looking at the data we can answer for example the question *"How many vehicles have passed during the period (aggregation step)?"* but not *"At what time has each vehicle passed?"*. This creates some issues with the accuracy and consistency of the data itself. E.g. if we have 10 vehicles which have passed through the measuring location in a period of 5 minutes, this cannot be interpreted as if 2 vehicles have passed every minute. The developed application was made to work with this limitation, but it should also be kept in mind by the user itself. To solve this issue, we can use a smaller aggregation step (the smallest one allowed by NDW is 1 minute), but this increases linearly the size of the downloaded data. Ofcourse in some cases, aggregated data is easier to interpret and bigger aggregation step is preferable. Our application can also aggregates data depending on the step, set by the user. Because of that we still recommend to download raw data with as lowest aggregation step as possible because it can lead to more thorough analysis.

Raw data files store all available data in one big table. This can make it slow and difficult to find and extract the information which is needed. There is also lots of duplicated data which uses lots of memory. Because of this, some processing is done with the raw data, before it is used by our application.

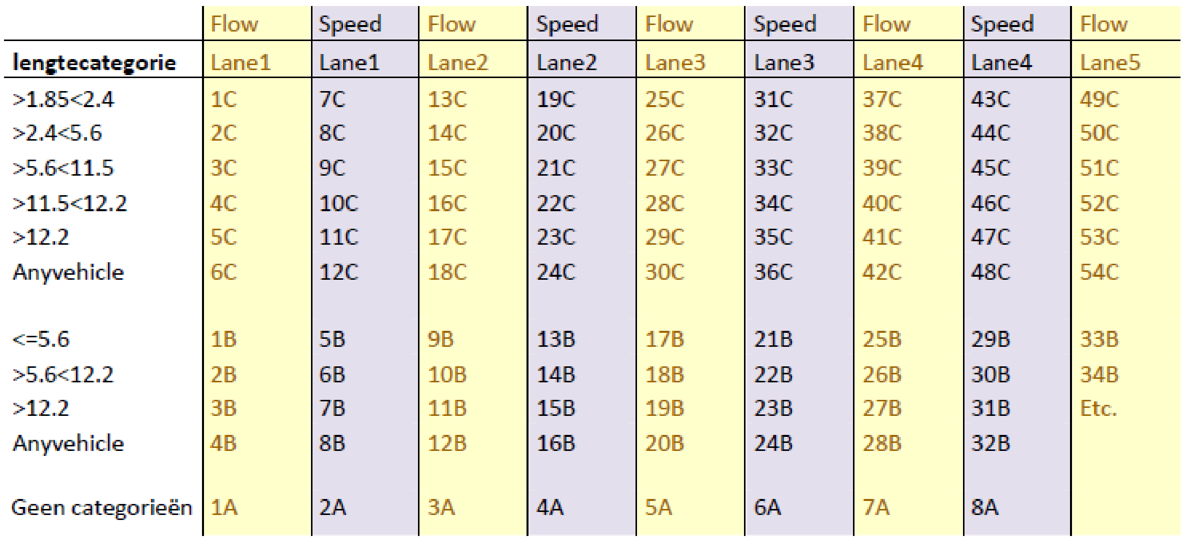
For each measuring period and for each measuring location we have several entries in the table: rows 2 to 5 for location GEO01_SRETI043r, 6-17 for GEO01_SRETI13r, 18-29 for GEO01_SRETI40, etc.. These entries for each location differ in the *index* (column C) field. The meaning of the indexes is explained further in this section. As we mentioned already, each entry stores all meta data for the measuring location. From here comes the first data processing step - store the meta data for each location in a separate table keeping only one entry for each location. The table is named *PointsMetaData* (Table 4.2).

column	column title	column	column title
A'	measurementSiteReference	AE'	startLocatieForDisplayLat
B'	measurementSiteVersion	AF'	startLocatieForDisplayLong
C'	indexes	AG'	LocationCountryCode
D'	periodStart	AH'	LocationTableNumber
E'	periodEnd	AI'	LocationTableVersion
T'	computationMethod	AJ'	alertCDirectionCoded
U'	measurementEquipmentTypeUsed	AK'	specificLocation
V'	measurementSiteName1	AL'	offsetDistance
W'	measurementSiteName2	AM'	LOC_TYPE
X'	measurementSiteNumberOfLanes	AN'	LOC_DES
Y'	measurementSiteIdentification	AO'	ROADNUMBER
Z'	measurementSide	AP'	ROADNAME
AA'	accuracy	AQ'	FIRST_NAME
AB'	period	AR'	SECND_NAME
AC'	specificLane	AV'	carriageway

Table 4.2: List of the column names that can be found in *PointsMetaData* table.

Apart from the meta data column we also kept the columns *periodStart*, *periodEnd* and added *indexes*. The first two keep the timestamps of the first and the last entry for the corresponding measuring location, i.e., we know the period for which we have measurements. The column *indexes* keeps all the indexes that are used for that location, e.g., if we consult with Figure 4.1 we can conclude that the *indexes* field (column C') for location *GEO01_SRETI043r* will have value *"1B 2B 3B 4B"*. The indexes for each location depend on number of lanes of the road there and on the number and on the number of vehicle classes that are recognized by the measuring hardware. Similar to the meta data, they are constant for every measuring point.

Index values encode more information about the measurement itself. They can give information about the location of the measurement (which lane of the road), the type of measurement (flow of vehicles or speed) and the type of vehicles for which the measurement is made (vehicles are divided into groups by their length). Figure 4.2 shows the indexes for regular lanes. The indexes for the special lines like *leftHandTurningLane, rightHandTurningLane, busLane etc.* are not used and shown in this thesis. This decision was a result of consultations with the future users from the OPAC group. It turned out that the traffic in such lanes was not of interest to them. Another reason is that during this project we did not find any entry in the raw data that has an index different from the ones shown in the figure.



	Flow	Speed	Flow	Speed	Flow	Speed	Flow	Speed	Flow
lengtecategorie	Lane1	Lane1	Lane2	Lane2	Lane3	Lane3	Lane4	Lane4	Lane5
>1.85<2.4	1C	7C	13C	19C	25C	31C	37C	43C	49C
>2.4<5.6	2C	8C	14C	20C	26C	32C	38C	44C	50C
>5.6<11.5	3C	9C	15C	21C	27C	33C	39C	45C	51C
>11.5<12.2	4C	10C	16C	22C	28C	34C	40C	46C	52C
>12.2	5C	11C	17C	23C	29C	35C	41C	47C	53C
Anyvehicle	6C	12C	18C	24C	30C	36C	42C	48C	54C
<=5.6	1B	5B	9B	13B	17B	21B	25B	29B	33B
>5.6<12.2	2B	6B	10B	14B	18B	22B	26B	30B	34B
>12.2	3B	7B	11B	15B	19B	23B	27B	31B	Etc.
Anyvehicle	4B	8B	12B	16B	20B	24B	28B	32B	
Geen categorieën	1A	2A	3A	4A	5A	6A	7A	8A	

Figure 4.2: Index values at regular lanes. Image is taken from "Handleiding Historische gegevens". Dictionary NL-EN: lengtecategorie - length category, Geen categorieën - uncategorized

To ease our work, organize the data in a better way and to decrease the time required to find an entry in the raw data we decided to do a second preprocessing step: create a new table for each separate measuring location and in each new table store all measurement results for this location. Each table is assigned the name of the location and has the columns shown in Table 4.3.

column	column title
A"	periodStart
B"	periodEnd
C"	Index 1
...	Index N

Table 4.3: List of the column names that can be found in each location data table.

If we turn our attention back to Figure 4.1 and column Q, which shows the intensity of the traffic for the measuring period, we will notice a strange anomaly. The first four rows of the table contain all measurements made at *GEO01_SRETI043r* location. The first three rows show the number of vehicles with length "$\leq$ 5.6 meters", "$<$ 5.6 $\leq$ 12.2 meters" and "$>$ 12.2 meters". The last row contains the total number of vehicles that have passed through that location. The common rule is (4.1):

$$N^{\underline{o}}vehicles(\leq 5.6) + N^{\underline{o}}vehicles(< 5.6 \leq 12.2) + N^{\underline{o}}vehicles(> 12.2) = N^{\underline{o}}All\_vehicles \quad (4.1)$$

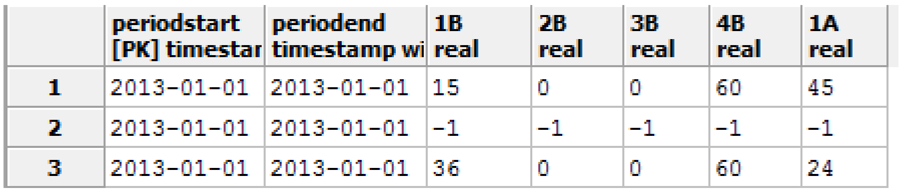
In our case for the number of passing vehicles we have 15, 0, 0 and 60 but $15+0+0 \neq 60$. This anomaly happens quite often in the data. After sending a question to NDW we got the following explanation:

Vehicles for which no length could be determined (because they were too long or because they were changing lanes over the loop, or any number of reasons) are counted for anyvehicle but not for any vehicleclass.

Hence the extra vehicles that are counted should belong to the "Uncategorized" group. In our research we did not find any entry that has any of the "Uncategorized" indexes, that is why we decided to make the best use of them by calculating the "extra" vehicles using

$$\begin{aligned} N^{\underline{o}}Uncategorized = N^{\underline{o}}All\_vehicles - N^{\underline{o}}vehicles(\leq 5.6) \\ - N^{\underline{o}}vehicles(< 5.6 \leq 12.2) - N^{\underline{o}}vehicles(> 12.2) \end{aligned} \quad (4.2)$$

and save the result in the locations data table along with the information for the other indexes. The resulting table *"GEO01_SRETI043r"* is shown below (Figure 4.3).



	periodstart [PK] timestar	periodend timestamp wi	1B real	2B real	3B real	4B real	1A real
1	2013-01-01	2013-01-01	15	0	0	60	45
2	2013-01-01	2013-01-01	-1	-1	-1	-1	-1
3	2013-01-01	2013-01-01	36	0	0	60	24

Figure 4.3: Table GEO01_SRETI043r loaded in a PostgreSQL database and viewed with pgAdmin III

These two preprocessing steps give a huge reduction in file size. Putting it into numbers, a 2013 year traffic intensity data for all measuring locations in the city of Eindhoven takes 53.8 GB of storage space. After optimization it needs only 3.04 GB, which is 5.6% of the original data and we do not have any important data loss.

Although NDW provides different types of traffic information, the focus of this thesis is on data about traffic intensity. The reason for this is that only a small number of the measuring locations provide more than traffic intensity data, in practice too few to be useful. NDW is constantly developing and improving, so we expect that in future the number of these locations will increase. In consultations with members from the OPAC group, we agreed that until this happens, they cannot make good use of the non-intensity data.

4.3 Roads Network Data

Road authorities have selected pathways that are important for the accessibility of the region. They called that network RVM-network (Regionaal VerkeersManagement - Regional Traffic Management). NDW is using this network to place measurement equipment and collect traffic data. This information is useful to us and can be used it to visualize the traffic data.

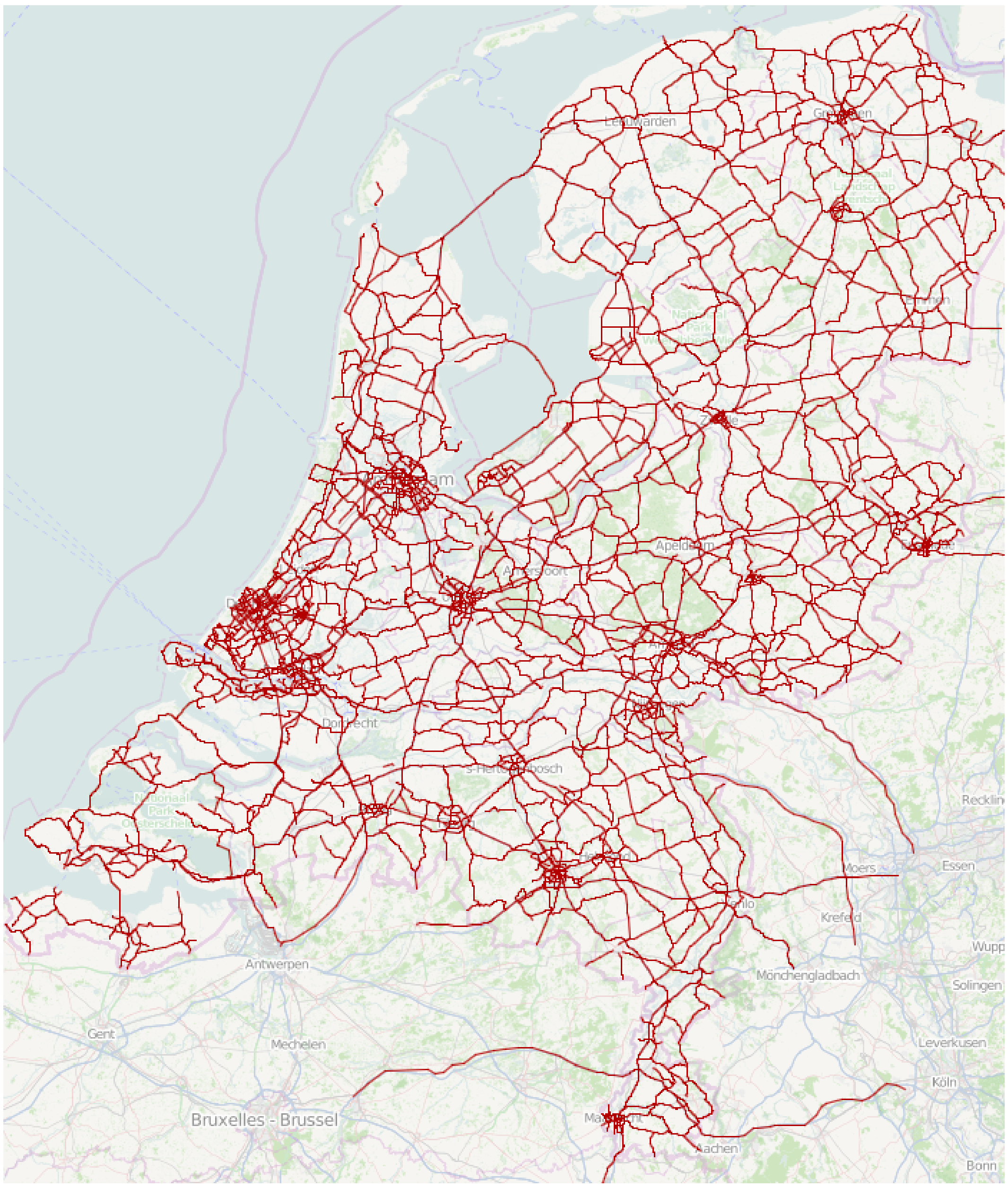


Figure 4.4: RVM network from June 2014

The RVM network is available for download on the NDW website in the form of a Shapefile (shp). For our application we have to change the format of the file to make it easier to read and edit. This was achieved with the help of a PostgreSQL database with PostGIS extension.

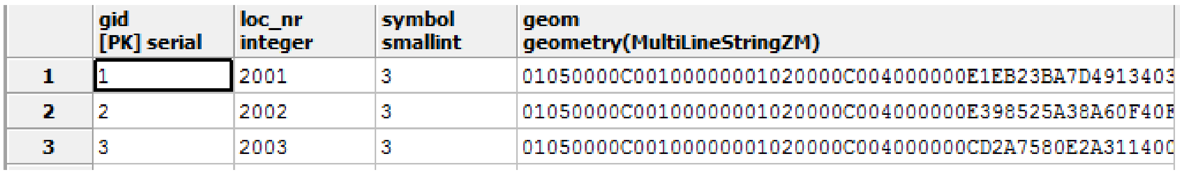
More technical details about the database used are available later in this thesis (Chapter 6 and Appendix A). PostGIS adds extra types (geometry, geography, raster and others) to the PostgreSQL database. It also adds functions, operators, and index enhancements that apply to these spatial types. With its help we can translate Shapefiles into a Structured Query Language (SQL) file and then import this file into a database. After that we can do some more processing to shape the data in the format most useful for us. The whole process is as follows:

Step 1: Translate Shapefiles into SQL file:
shp2pgsql -s <SRID><shapefile><tablename><gisdatabase>>filename.sql

Step 2: Import the SQL file into the database:
psql -d <gisdatabase>U <username>h <hostname>p <port>-f filename.sql

Step 3: CREATE TABLE <tablename>AS
SELECT <column1,...,columnN>,ST_AsGeoJson(<columnX>) FROM dataTableName

If pgAdmin III is used to access the database, then it is easier to use the graphical user interface to directly perforn Steps 1 and 2. We perform this operation on the RVM network Shape file. After the first two steps we have the table presented in Figure 4.5.



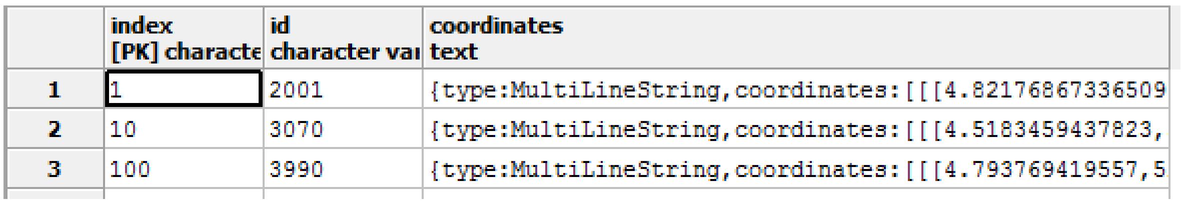
	gid [PK] serial	loc_nr integer	symbol smallint	geom geometry(MultiLineStringZM)
1	1	2001	3	01050000C00100000001020000C004000000E1EB23BA7D4913403
2	2	2002	3	01050000C00100000001020000C004000000E398525A38A60F40E
3	3	2003	3	01050000C00100000001020000C004000000CD2A7580E2A311400

Figure 4.5: The result table from steps 1 and 2 performed on the RVM network Shapefile.

Each row stores the available information about one road from the RVM network. Columns *gid* and *loc_nr* present a unique number used for identification of the entry. The first one is added by the database upon import of the SQL file. The second one is proovided by NDW and the information there was stored in the Shapefile. After step 3 we decided to keep both of them as they can be useful for future identification of the roads. Column *symbol* does not provide any information of interest to us, that is why we drop it.

The last column *geom* keeps information about the path of the road on the map. After Step 2, this column is in geometry(MultiLineStringZM) format. We decided to transform it to JSON format. This has the following advantages: first, it will be more easy for the user to read and understand what is written in that column and second, we can use libraries that are specially developed to work with JSON string, which makes their reading and processing much easier. After Step 3 we have the final result (Figure 4.6). We name this table *tmc_line*.

Columns *gid*, *loc_nr* and *geom* are renamed respectively to *index*, *id* and *coordinates*. Each road is given as a series of locations through which it is passing. Each location is given with its three-dimensional coordinates (*longitude*, *latitude*, *hight*). The height of all of them is 0, so in our work we only use the latitude and longitude.



	index [PK] characte	id character vai	coordinates text
1	1	2001	{type:MultiLineString,coordinates:[[[4.82176867336509
2	10	3070	{type:MultiLineString,coordinates:[[[4.5183459437823,
3	100	3990	{type:MultiLineString,coordinates:[[[4.793769419557,5

Figure 4.6: *tmc_line* table. The final result after performing Step 3.

4.4 Other Data

NDW provides two more pieces of information, again stored in the form of Shape files. The first one contains the outlines of the municipalities in Netherlands (*gemeente* in Dutch), the second piece the coordinates of all measuring locations. Both Shapefiles have the same form as the one presented in the above section. The municipality outlines are given as a sequence of coordinates, while the locations are given with only one set of coordinates.

Although both files provide us with usable information we decided not to use it for the following reasons. Municipality outlines can be useful but we do not need them for something more than just showing them on the map. In our application we use as background a map on which they are already drawn, hence we can avoid using this file. The locations file gives us only the coordinates of the locations and no other useful information. Hence we prefer to construct the *PointsMetaData* table described in Section 4.2, Table 4.2.

Chapter 5

Data Visualization

In Chapter 2 was stated why it is important to have a good visualization of city traffic data. Most of the existing tools and applications do not provide the full functionality which is required for this project. Following the requirements we had to improve the old solutions and develop some new ideas and techniques that make a better use of NDW data and suit *Smart Logistics* researchers' needs.

Following the requirements for temporal and spatial views (**R3, 4, 5, 7, 9, and 10**) we designed two types of views that we believe to satisfy of them. The first one visualizes data over a geographical map. The second one visualizes data as time series, depending on the selected data and time periods. Both views produce different type of images, but result from projecting the same city model and controllers for filtering the data.

5.1 City Model

The city model consists of roads, measuring locations and some extra meta information which is added by the user and can be used later for further exploring. The model aims to aid the visualization process by first constraining the number of data sources that are used (take and process data only from locations that we want to explore); reduce the size of the road network, which we want to visualize; and combining measuring locations into groups. In this section we explain the model in more detail. More description about our tool functionality and detailed description how to create a city model can be found in Appendix B.

Figure 5.1 shows the difference between all the available data for the measuring locations and RVM network, and the trimmed data which is used to create a model. The created model helps us to focus only on the central part of Eindhoven, especially the ring and the roads that lead to it from outside. We have removed the unnecessary roads and the roads for which we have no measuring locations and the locations which are not in the city. Immediately we can notice a more clear picture because of the reduced visual clutter caused by drawing unnecessary road lines.

Because we want to visualize data for a whole road rather than a single location we need to identify which measuring location relates to which road of the RVM network. NDW does not provide us with clear information for this. Measuring locations are given with their geographical coordinates and name of the street on which they are located, but the roads are given only with their coordinates and not a name. We tried to automate the process using the shortest distance from a measuring location to a road but this gave us some issues

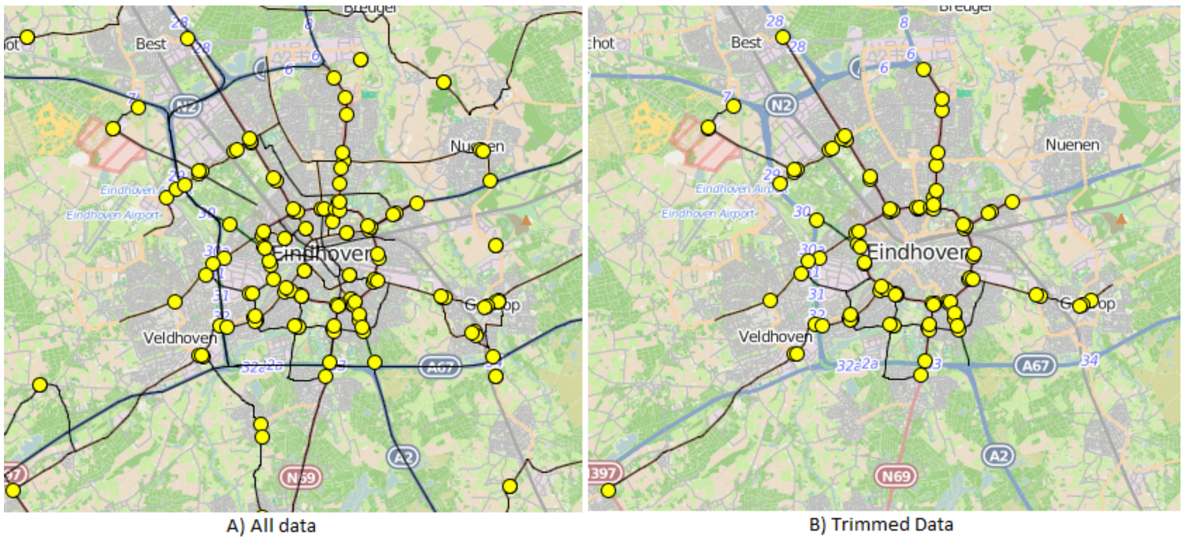


Figure 5.1: Model of the city of Eindhoven. Image A) displays all the available information which we have about road network and measuring points locations. Image B) shows only the information which is needed to create a sample model of Eindhoven. Black lines show the RVM road network and the yellow dots represent the location of each measuring station. Measuring locations are positioned using the coordinates that comes from *PoiintsMetaData* (Table 4.2). Roads are drawn using the coordinates from *tmc_line*(Figure 4.6) as a guideline and simply connecting them with a line.

(Figure 5.2).

For automating the process we calculated the distance between each measuring location and each road in our model, next we linked the location to the road it was closest to. As Figure 5.2 explains, this resulted in linking some locations to the wrong road. The problem occurred because some locations were positioned closer to one road although they were measuring the traffic on another road. To solve this problem the user has to first identify the problematic areas and then either redraw some of the roads so that they go further from some locations or change the drawn position of the points so that they are closer to the road they belong to. Unfortunately this would take too much time and efforts for the user to correct everything and it is also likely that he might miss some of the errors. Because of that we decided to drop this solution and let the user manually assign location to road. Although this can also require a lot of effort, it also has the advantage that each linking is inspected by the user upon building the model. Another advantage is that the user can choose which locations and roads he wants to link together. The unused objects will not be viualized, hence reduce clutter in the image.

Figure 5.3 gives an example with a few measuring locations. Each measuring location provides data only for traffic going in the same direction. This is important information to be taken into account. Hence upon binding location to a road, the user should also provide information about the direction of the traffic that is measured (Figure 5.4). We implemented a semi-automated function which can save the user some time. More details about it can be found in Appendix B.

The last piece of information that is specified in the model is the information about labels

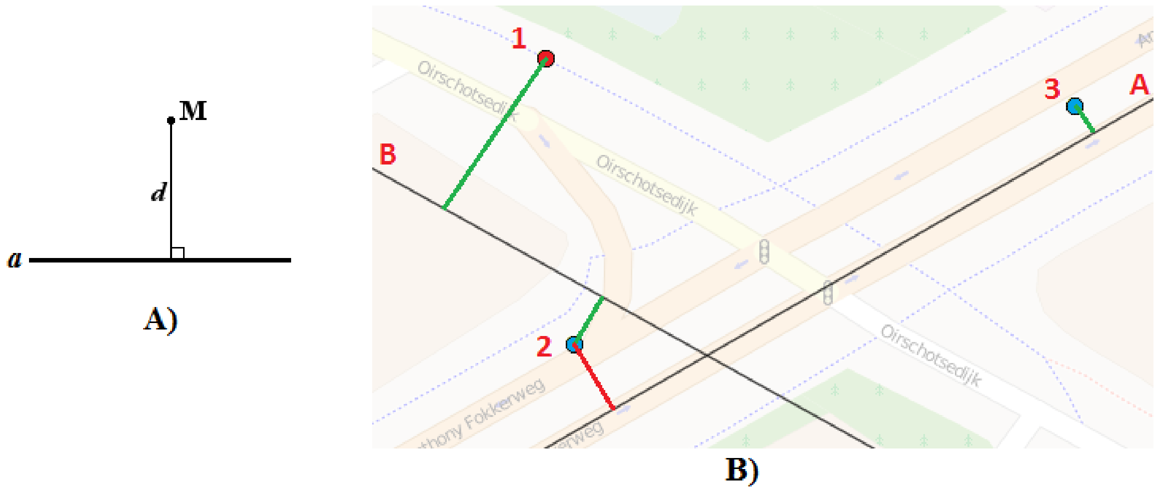


Figure 5.2: A) Shortest distance from point M to line a can be found by dropping a perpendicular line from M to a and measure its distance d. B) Actual image taken from our tool presenting the distance between three measuring locations (1,2,3) and two roads (A,B). We know for fact that point 1 should be positioned on road B, while points 2 and 3 on road A. Green lines show the shortest distance between the points and any road. The red line shows the distance between point 2 and road A. Positioning locations using shortest distance to a road would result an incorrect positioning of point 2 on road B. The green and the red lines are only annotations and are not part of our application.

of measuring locations. Requirement **R8** states that user should be able to identify different type of traffic i.e. there must be some method for grouping measuring locations with similar properties. This is done by enabling the user to select a group of points and apply a label to them. It is practically possible that one or more locations fit in more than one group (e.g. "Traffic nf Boschdijk street" and "Traffic entering the RING"), thus we allow applying several labels to a measuring point. The user can also elects a color to represent each label. On Figure 5.5 two labels are applied to the measuring locations. Label *ENTRANCE* (green) groups the locations that measure the incoming to the city traffic and label *EXIT* (red) the locations measuring outgoing traffic. We tried to make the selection carefully and identify the correct label for each location based on our knowledge of the city. We believe that the quality of our selection is sufficient and we use the specified labels in some of our demonstrations later in this thesis.

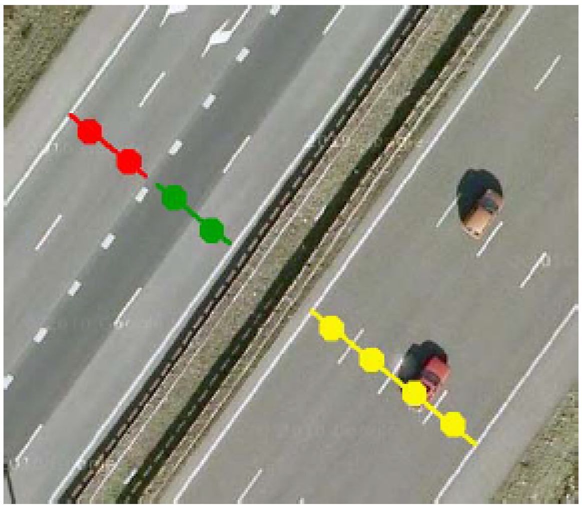


Figure 5.3: Example of three measuring locations - yellow, green and red. Each location consist of several measuring points, each measuring point collecting information for one line of the road. Each measuring location is combining traffic information about the same type of lanes. Yellow location - four lane road, all lanes are provided for traffic going forward. Green location - four lane road, the two lines are provided for traffic going forward. Red location - four lane road, the two lanes are for traffic leaving on the right side. Image from *NDW Interface beschrijving*.

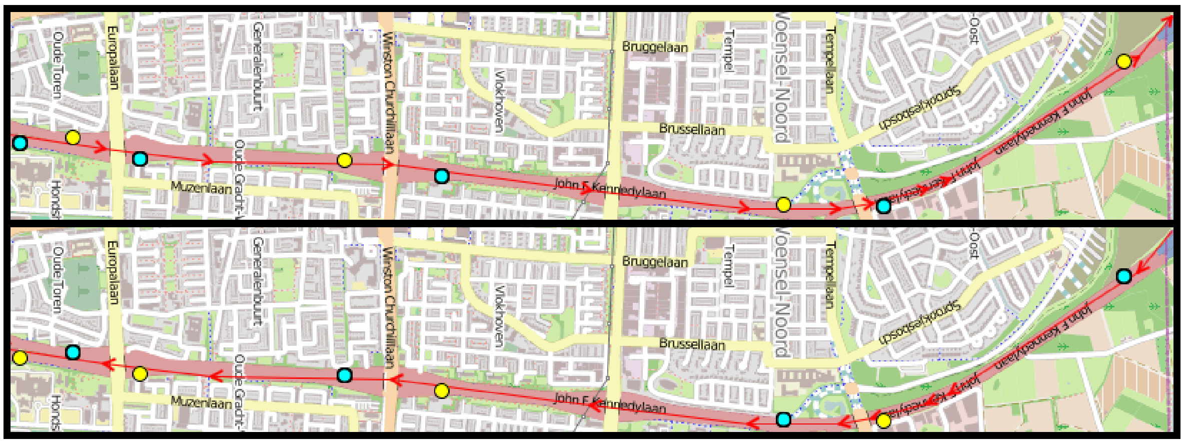


Figure 5.4: Measuring locations linked to *John F. Kennedylaan.* The arrows on the road show the traffic direction and the dots in blue indicate the locations that measure the traffic in that direction.

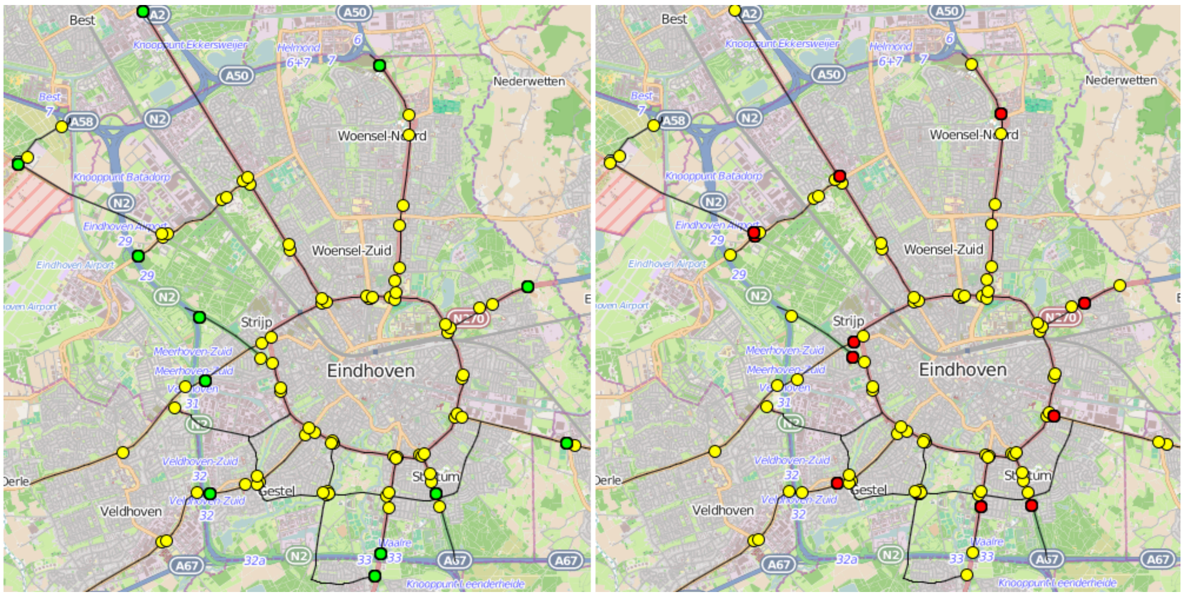


Figure 5.5: Labeling measuring locations in the city of Eindhoven. Label *ENTRANCE* (green) is used for incoming traffic locations and label *EXIT* (red) - for outgoing traffic locations.

5.2 Basic controllers

Following the problem description and requirements, users should be able to apply filtering on the data. The three main filters which were identified and applied are:

- Filter based on vehicle type;
- Filter based on measuring location or group of measuring locations (label);
- Filter based on time.

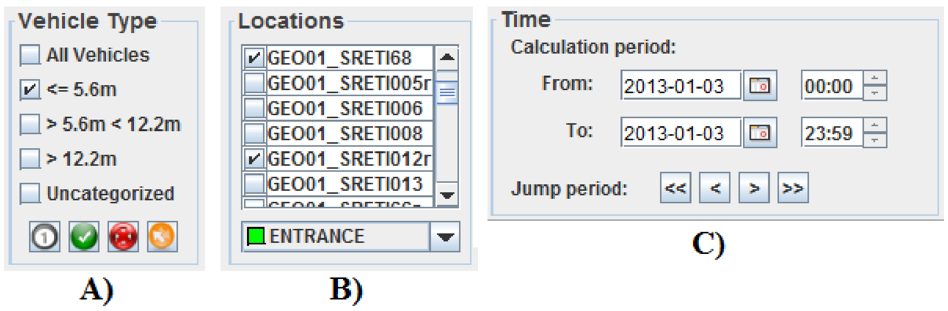


Figure 5.6: User interface for applying basic filtering. A) Filter for vehicle type; B) Filter for measuring locations; C) Filtering based on time period;

We tried to design the interface to be simple, easy to use and time-saving. All controllers that are responsible for one filter are put together in a separate panel (Figure 5.6). In the vehicle type panel the user can select the class of vehicles for which he wants to create a visualization. He can select more than one class as this will allow him to make combinations, e.g., if he is interested in all vehicles longer than 5.6m, then he has to select "*>5.6m <12.2m*" and "*>12.2m*". The four buttons at the bottom of the panel allow to respectively select only one class, select all, deselect all, and invert the selection. When the *select only one class* button is pressed, then the user can use only one class for his selection. If he select a new class, then the old one gets deselected. This saves some time as a user can quickly jump between subsets without losing his attention to the image. *Select all* allows the selection of all classes of vehicles at once. The button selects all but the *All vehicles* check boxes. As it was shown in Chapter 4, the sum of these vehicle groups makes the total sum of all vehicles. We decided that selecting all but *All vehicles* is the better solution than selecting only this check box, as it will allow the user to make faster selection of at least three vehicle classes together, e.g., if we are interested in all vehicle but the unknown, then we press *select all* button and then deselect the one which we do not need. *Deselect all* allows us to quickly clear our selections. The *invert selection* button helps to quickly invert a selection, e.g., unselected classes get selected and selected classes get deselected.

The locations panel helps the user to select the locations from which he wants to obtain data. It allows the selection of either individual locations or group of locations having the same label.

The time controllers panel is inspired by the work of Andrienko [6]. They propose the idea to add buttons (bottom of the panel) that enable the user to quickly jump forward and backward in time. This can be used to simulate an animation and allows spotting time variances in the data. The exact jumping intervals are explained in the next sections. Date selectors have a button next to them which opens a calendar. This allows quick and accurate selection of a date.

These three groups of controllers compose the main part of our user interface. The other part of the interface varies depend on the type of visualization chosen by the user. The new parts are explained in next sections of this chapter. The three main groups of controllers also have minor variations depending on the type of visualization.

5.3 Visualization of traffic on a map

Visualizing traffic data on a map is one of the main requirements which we have for this project. We use the already created city model to create views. Before we create any visualization, we need an appropriate map. We decided to use the *standard layer* Open Street Map[1]. This layer provides information about roads, buildings, area borders, etc. Every different type of objects has it own color which makes the map fast and easy to understand and follow. Being able to distinguish the separate objects like specialized buildings for example can prove useful in the later analysis of the data.

All map based views are designed to show the spatial properties of the data. For that we use the location of each measuring station. Before generating any image we have to make the appropriate calculations based on the user preferences and applied filters. We use algorithm 1 to obtain the final results and then we keep the numbers for each location.

Algorithm 1 Pseudo code for generating values for each location

```
1:  Input: List of locations L, List of vehicle classes V, Time period $T[t_{begin}, t_{end}]$,
2:  and flag $vehiclesPerHour$
3:  Output: Value $result$
4:  for location $l$ in $L$ do
5:      Obtain data $D$ for location $l$, vehicle classes $V$ and time period $T$
6:      $sum \leftarrow 0$
7:      for line $d$ in $D$ do
8:          for Vehicle class $v$ in $V$ do
9:              $sum \leftarrow sum + d_v$ Note: $d_v$ is the number of vehicles for class $v$ in line $d$.
10:     if $vehiclesPerHour$ then
11:         $result \leftarrow sum/totalLines$
12:     else
13:         $result \leftarrow (sum/totalLines) * (T/60)$
14:     Save $result$ in $l$.
```

The pseudo code presents shortly the algorithm which we use for calculating the final result for each location. The inputs for the algorithm are list of measuring locations L, list of the selected vehicle classes V, the selected time period T and boolean flag *vehiclesPerHour*. The location list L contains only the locations that will be used for the visualization e.g. locations that are selected, locations linked to roads. This helps us to decrease the total computation time. The time period T and the list of vehicle classes V is used to reduce the size of the data that is need for the calculations. The data is obtained from the locations

[1] http://www.openstreetmap.org

tables described in Chapter 4. The boolean flag *vehiclesPerHour* shows the form of the final result. The result can be given in either *vehicles per hour* or *total number of vehicles.* According to NDW *vehicles per hour (veh/h)* is the standard measurement unit for traffic intensity. Because of that the data provided is given in this unit. Hence in our locations data tables we also store the traffic intensity in veh/h. If the user selects this option then he gets the aggregated average intensity (line 10 in the algorithm).

$$I_{aggr} = \frac{\sum_{i=1}^{N} I_i}{N} \tag{5.1}$$

Where I_{aggr} is the average intensity for the whole period, N is the number of measurements used, and I_i is the intensity for measurement i. Our target group of users is mainly interested in the total number of vehicles that are passing through some point. Because of that we added this option for the final calculations result (line 12).

$$V_{total} = \frac{\sum_{i=1}^{N} I_i}{N} * \frac{T}{60} \tag{5.2}$$

Where V_{total} is the total number of vehicles for the whole period, $N$is the number of measurements used, I_i: Intensity for measurement i, and T is the length of the time period (in minutes). Calculated results are stored for each measuring location in order to be used for the visualization.

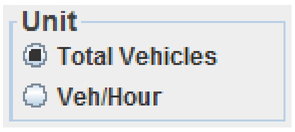


Figure 5.7: Unit panel used for selecting the measurement unit for the final result.

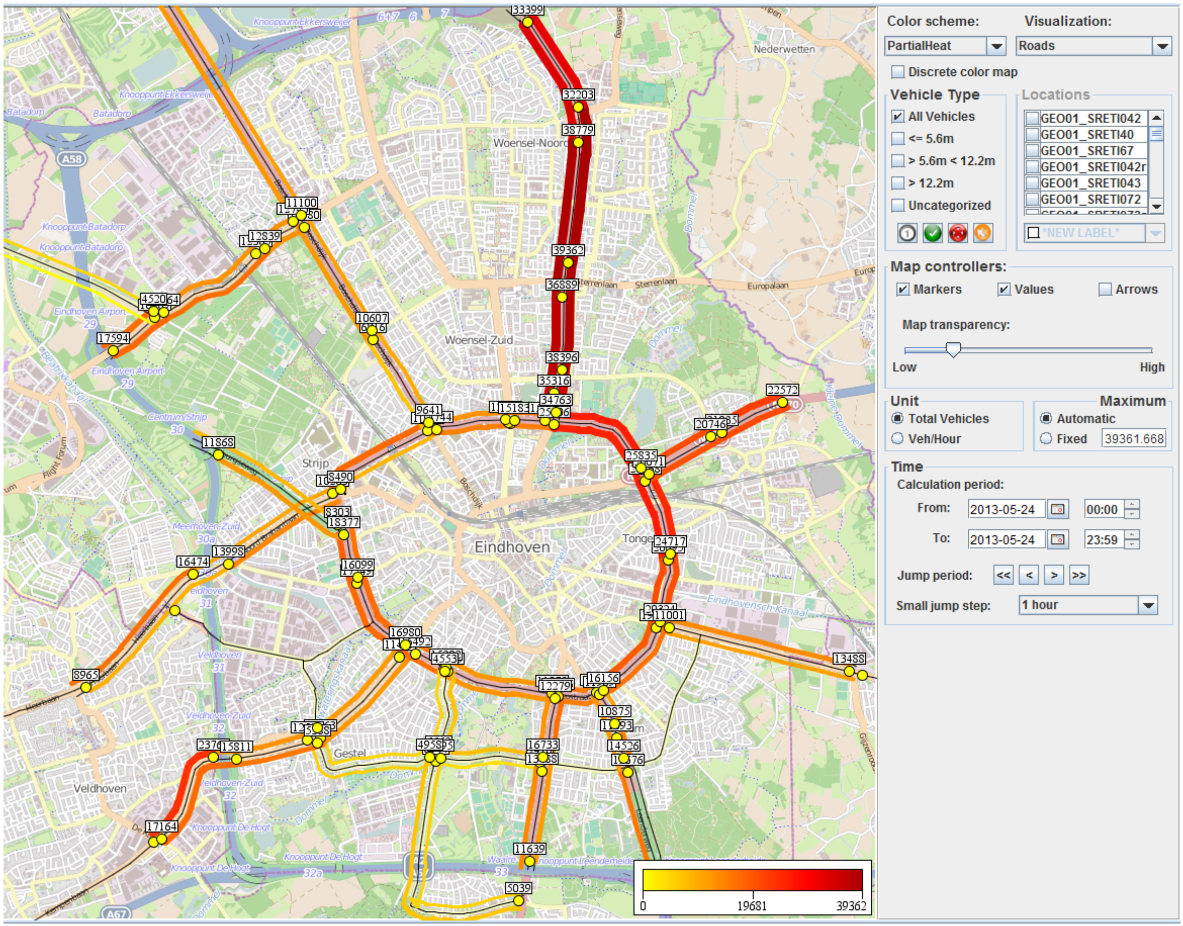
5.3.1 Visualization of traffic over the road network

The first map view which we implemented is a traffic view that colors the road network depending on the traffic intensity (Figure 5.8).

The visualization image is generated in four steps:

1. Calculate the values for each location;
2. Calculate parallel lines on both sides of each road, which are used to display the traffic in each direction;
3. Draw the roads using colors, respective to the visualized values;
4. Draw locations, values and arrows if selected.

Step 1 is explained before. As shown in Figure 5.8, we use two parallel polylines on a distance d on each side of every road to present the traffic in each direction. To calculate then each segment ($\overline{AB}$) of the polyline (P) that describes the main road is taken (Figure 5.9).



Color scheme: PartialHeat
Visualization: Roads
☐ Discrete color map

Vehicle Type
☑ All Vehicles
☐ <= 5.6m
☐ > 5.6m < 12.2m
☐ > 12.2m
☐ Uncategorized

Locations
☐ GEO01_SRETI042
☐ GEO01_SRETI40
☐ GEO01_SRETI67
☐ GEO01_SRETI042r
☐ GEO01_SRETI043
☐ GEO01_SRETI072
☐ *NEW LABEL*

Map controllers:
☑ Markers ☑ Values ☐ Arrows
Map transparency: Low — High

Unit
● Total Vehicles
○ Veh/Hour

Maximum
● Automatic
○ Fixed 39361.668

Time
Calculation period:
From: 2013-05-24 00:00
To: 2013-05-24 23:59
Jump period: << < > >>
Small jump step: 1 hour

Figure 5.8: All traffic intensity for the city of Eindhoven for 24 May 2013. The image is created with our tool.

Then we find points *A'*, *A"*, *B'*, *B"* such that $\overline{AA'}$, $\overline{AA''}$, $\overline{BB'}$ and $\overline{BB''} \perp \overline{AB}$ and $|AA'|$, $|AA''|$, $|BB'|$ and $|BB''| = d$. This ensures that $\overline{A'B'}$ and $\overline{A''B''}$ are parallel to $\overline{AB}$, have the same length as $\overline{AB}$ and are on a distance d from $\overline{AB}$. The newly calculated points *A'* and *B'* are added to the list of points that describes the polyline used for traffic in positive direction, and *A"* and *B"* respectively to the list of points that describes the polyline used for traffic in negative direction (assuming that $\overrightarrow{AB}$ is the positive direction). This can cause some clutter problems as we can see on Figure 5.10. As the road makes a curve, the polyline which is on the inner side of the curve makes a small triangle. The sharper the curve is, the more obvious this triangle is.

We can see that the triangles are relatively small compared to the length of the roads and in practice these are hardly noticable (Figure 5.8). Nevertheless, a better soutions should be implemented.

The next step in generating the image is to color the new parallel polylines to show the values calculated for the measuring locations. In the construction of the city model we linked each location to a road and indicated in which direction that location is recording the traffic. Each of the two parallel polylines has a direction. To draw a colorful polyline we need the locations that measure traffic on that road and in the same direction indicated by the polyline.

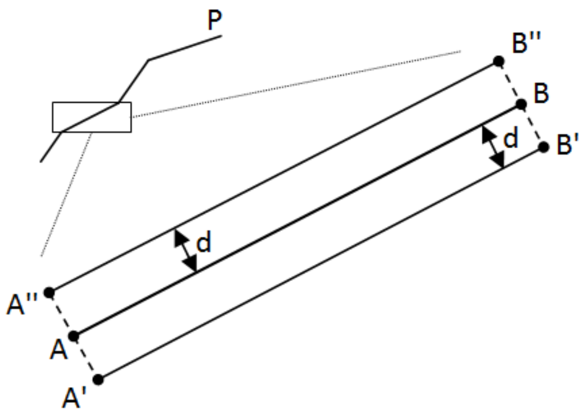


Figure 5.9: Drawing parallel lines $\overline{A'B'}$ and $\overline{A''B''}$ lines on both sides of segment $\overline{AB}$ at distance d from it and with the same length. The main segment $\overline{AB}$ is part of the polyline P.

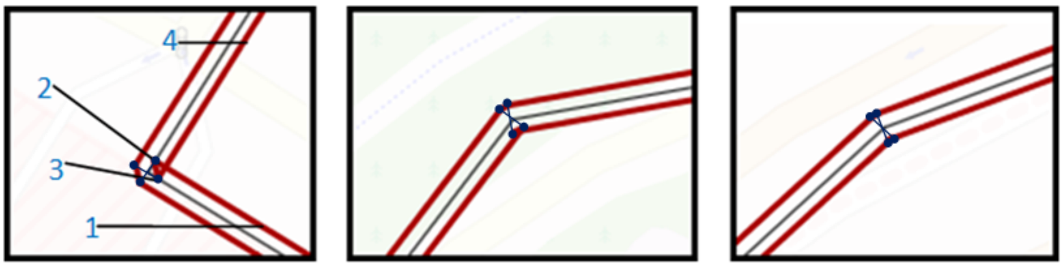


Figure 5.10: Cluttering triangle at the curve of the road, caused by the order of adding points to the polyline points list. The sharper the road curve is, the bigger the triangle is. The numbers indicate the order in which the inner polyline is drawn.

For each location we find its projection onto the polyline. The segment of the polyline located between two projections *R'* and *B'* (Figure 5.11) is drawn in the color determined by location *R*, assuming that we draw the polyline in direction from *R'* to *B'*.

One of our aims is to draw the user's attention to areas with heavy traffic. Therefore we also change the thickness of the line segment depending on the traffic there. Roads with heavier traffic are drawn with thicker lines and roads with lighter traffic - with thinner. For drawing each road, a polyline with 1 pixel thickness is used. It is made as thin as possible, because we don't want it to distract the user. The thickness of colored polylines can vary from 5 pixels to 10 pixels. Even the thinnest segments are significantly thicker than the road, because we want them to be big enough in order for the user to be able to determine their color. The maximal thickness was chosen such that it is enough to create a readable image but in the same time small enough so that the roads on the map are still visible.

The color schemes for the roads are designed to be easy to understand and to draw the attention to the important areas. The main questions which are to be answered with the help of our tool involve identifying heavy traffic. Hence we need color schemes which high-value colors are easily distinguishable and more noticeable. We tried to solve one of the problems which we mentioned about Google Maps color scheme - light colors are located somewhere in the middle of the scheme. Here they are put at the beginning so that the higher the value,

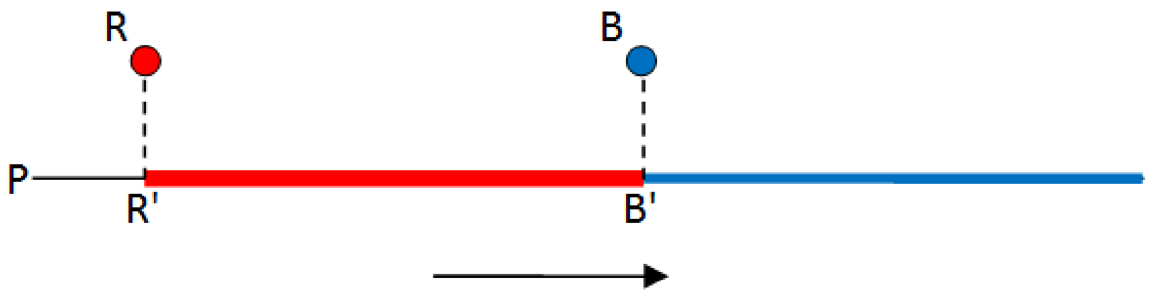


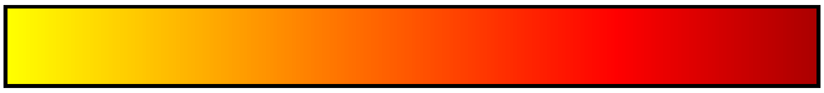
Figure 5.11: Drawing polyline *P* in different colors and thickness. The polyline has two measuring locations associated with it - R and B. First the projections R' and B' of these locations onto the line are found. The arrow indicates the direction of drawing of the polyline. The polyline is drawn from the beginning using the default color and thickness. After each projection the parameters are changed depending on the value of the respective measuring location. The colors and thickness used in this figure are for demonstrative purposes only. In our application location points are drawn in the same default color.

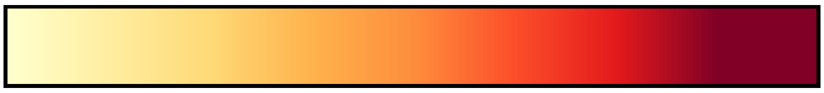
the darker the color gets. The high end colors should also be clearly distinguishable from the colors used in background map. We also want color schemes to make a transition between at most three different leading colors. The small number of colors makes the visualization easier to understand and more clear. Furthermore, the human eye can easily distinguish between shades of the same color than between shades of different colors.

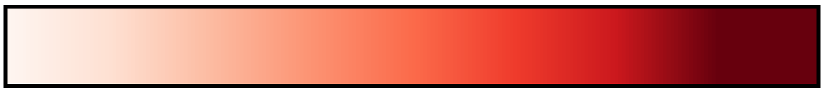
Most of the schemes which we designed and used are inspired by ColorBrewer[2]. This website gives many discrete color schemes, specifically designed to be used on geographical maps. The schemes suggested there, depend on the type of data visualized: sequential, diverging, and qualitative. We need to visualize sequential data here. Schemes for sequential data are given in discrete form with up to 9 colors. To make a continuous scheme out of them we use a 9 color scheme as a guideline and do a smooth transition between them. The result can be seen on Figures 5.12b 5.12c 5.12d 5.12e and 5.12f. On Figure 5.12a the partial hot body color scheme is shown. The hot body scheme is relatively well accepted within the visualization community. It provides smooth transitions between the neighboring shades without many sudden jumps in the colors. The scheme goes from white to yellow, orange, red, dark red and black. For our needs we dropped the white and black color: white because it is widely used in the background map, and black because it is too dark a color and attracts user attention less than the red color shades. Thus we ended up with a color scheme very similar to Yellow-Orange-Red scheme by ColorBrewer. The main difference is that our colors look brighter.

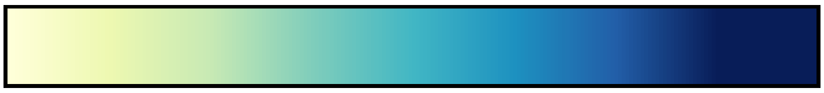
The next scheme designed by us is the Cyan-Magenta scheme (Figure 5.12g). It progressively changes color from cyan to magenta. The reason to pick only two colors is that in this way we have smooth transition between the neighboring shades without sudden jumps. Cyan and magenta are two of the main colors in the CMYK color model. This makes them easily distinguishable from each other. Another reason for exactly these colors is that they are not used in the background map and that they attract user attention more easily. We choose our scheme to start with cyan because the magenta color is darker and hence more intuitively

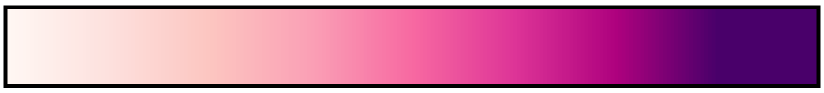
[2] http://colorbrewer2.org/

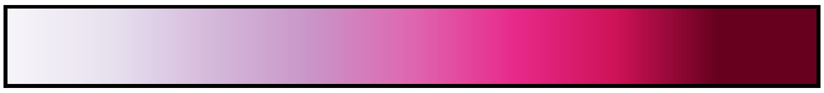

(a) Partial Hot Body

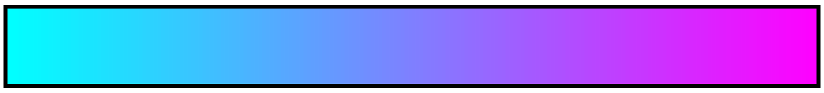

(b) Yellow-Orange-Red


(c) Reds


(d) Yellow-Green-Blue


(e) Red-Purple


(f) Purple-Red


(g) Cyan-Magenta

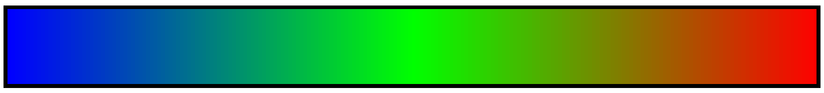
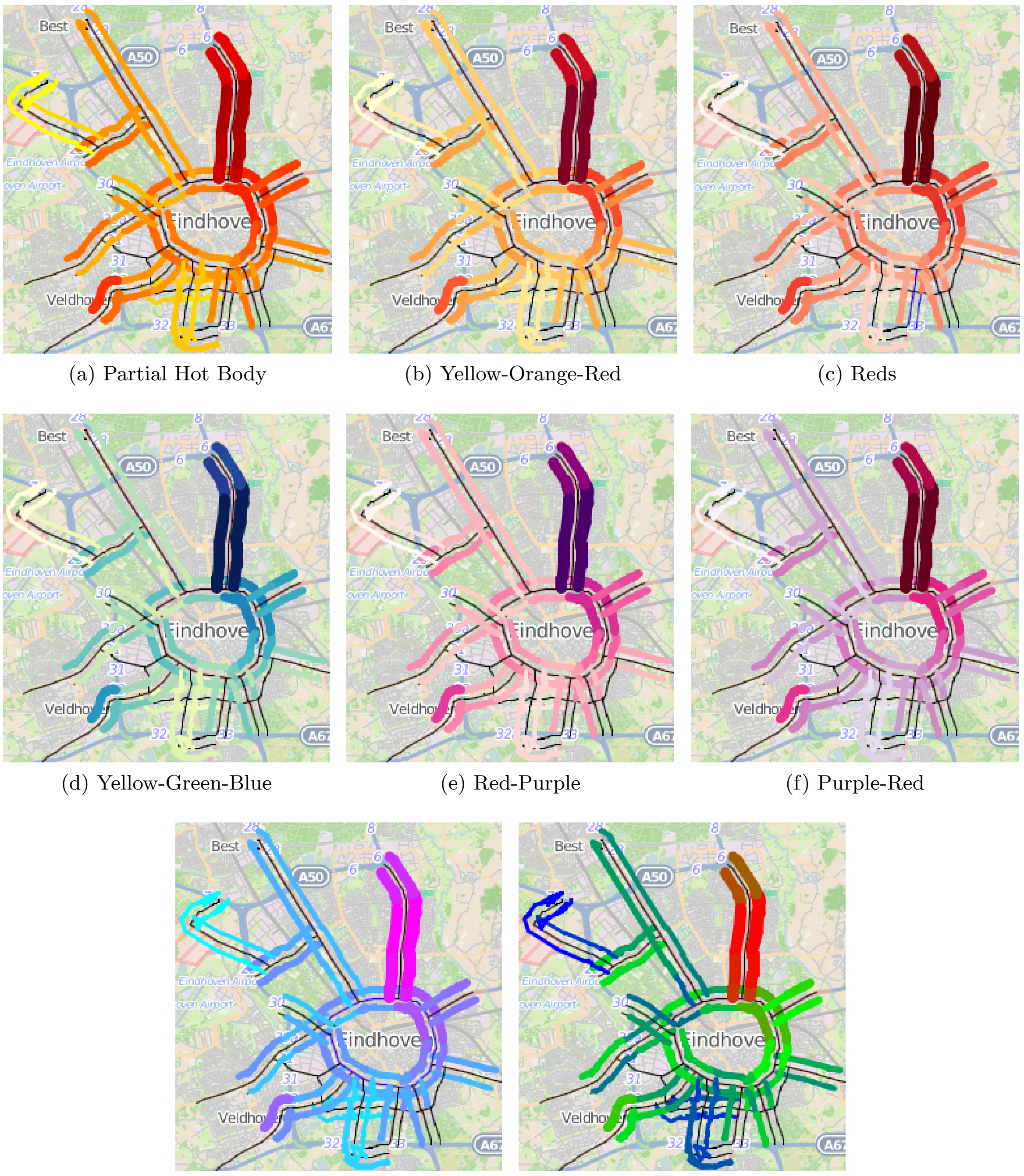

(h) Blue-Green-Red

Figure 5.12: Color schemes that are used in the tool.

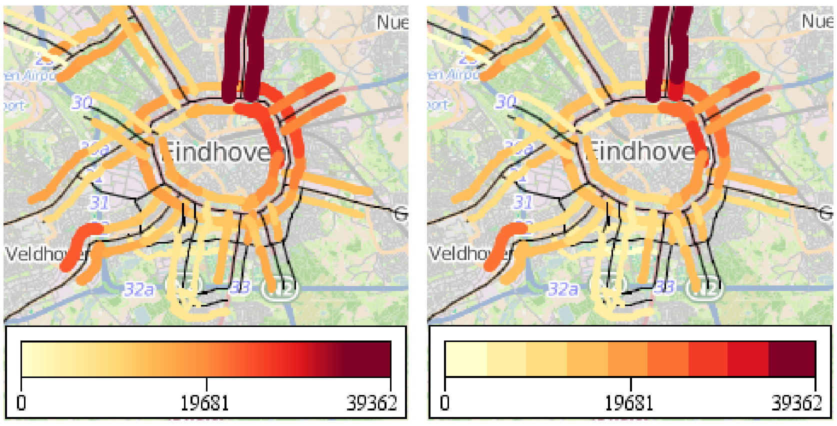


(a) Partial Hot Body (b) Yellow-Orange-Red (c) Reds

(d) Yellow-Green-Blue (e) Red-Purple (f) Purple-Red

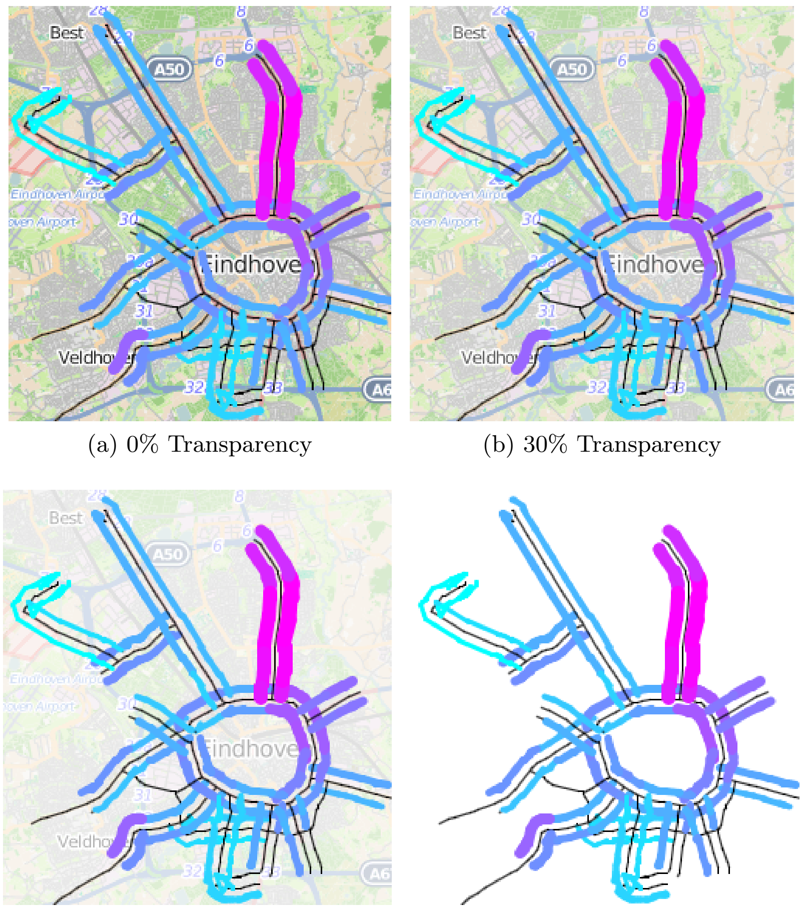
(g) Cyan-Magenta (h) Blue-Green-Red

Figure 5.13: Visualizing the same data as on Figure 5.8 using different color schemes.



(a) Continuous (b) Discrete

Figure 5.14: Traffic intensity visualization using continuous and discrete version of the *Yellow-Orange-Red* color scheme.



(a) 0% Transparency (b) 30% Transparency

(c) 60% Transparency (d) 100% Transparency

Figure 5.15: Traffic intensity visualization using the Cyan-Magenta color scheme and four levels of transparency for the background map.

connected with a higher value. Some of the shades in the middle of the scheme can be considered similar to the background map colors used for highways (light purple) and buildings (shades of gray). This issue can be avoided by adjusting the brightness of the background map (explained later in this chapter). Nevertheless, high value colors cannot be mistaken and clearly identify high traffic intensity (Figure 5.13g).

The last color scheme implemented is Blue-Green-Red (Figure 5.12h). Our initial idea was that since it makes a transition between all main colors of the RGB model it will be easy to see the difference between the remote color shades. If we check the image on figure 5.12h and pay more attention to the colors which are slightly left and slightly right from the middle we can conclude that they are very similar although there is a significant distance between them. This can mislead the user while analyzing the image. According to us this is the least useful scheme.

Mapping values to a color happens in the following way. First, we find the highest calculated value. Then it is mapped to the last color in the color scheme and zero is mapped to the list color. The rest of the colors in the color scheme are mapped evenly to the values between 0 and *max value*.

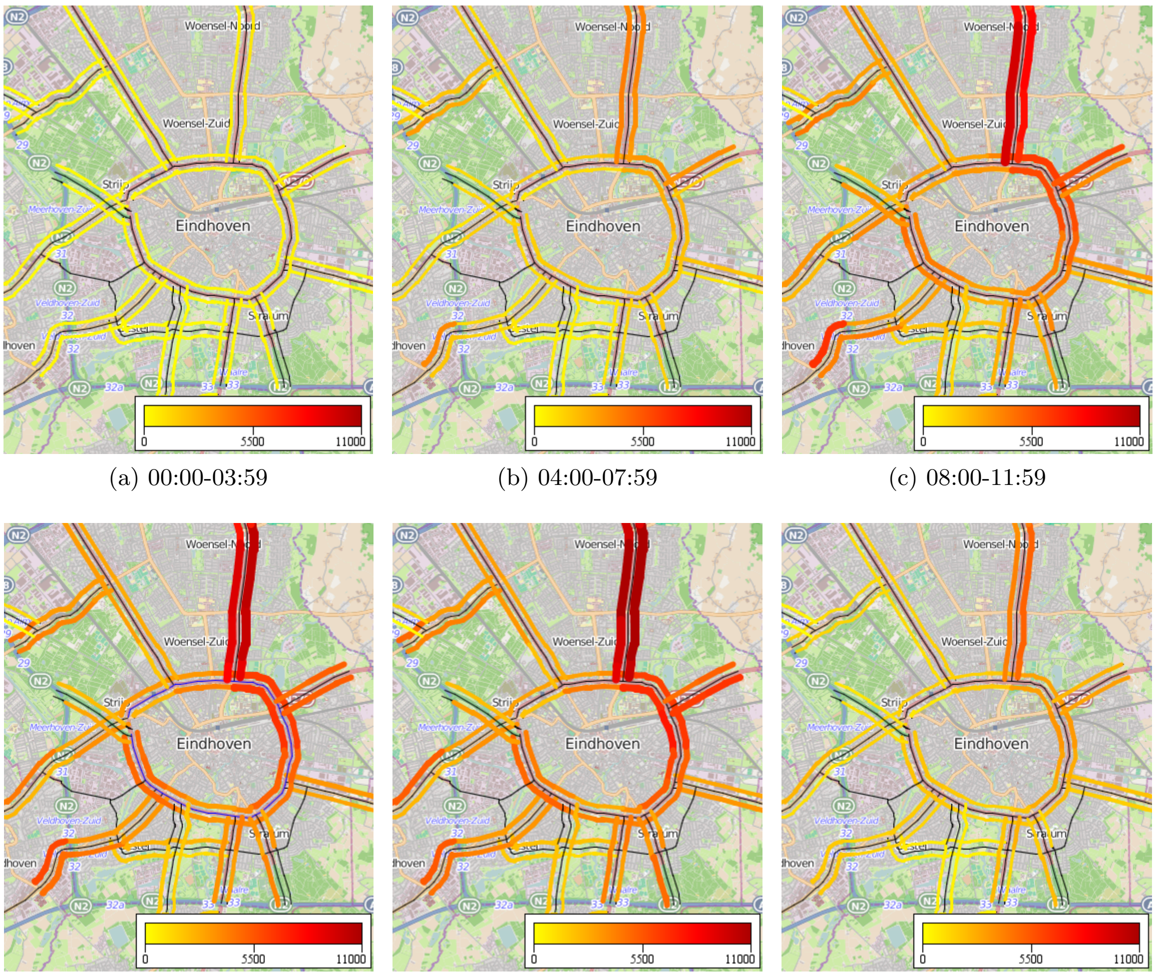
We conducted a small research amongst members and researchers from *Smart Logistics* lab as our main target group, and also some students from other work groups. We showed them images similar to figures 5.12 and 5.13 asking them for their opinion about:

1. *if the color scheme is useful;*
2. *if you see the difference between colors/traffic intensity easily;*
3. *which scheme is the best one?*

We did not get a clear winner about the best one, although Cyan-Magenta and Yellow-Green-Blue were least preferred. Two groups of users were identified - one that recognizes the more colorful schemes as being more useful and one that finds less colorful schemes more useful. Subjects from the first group even pointed Blue-Green-Red as being the best scheme despite its flaws. It seems that different users have different preferences and different ways of reading colors. Because of this reason and because all ColorBrewer schemes used are designed to be more printer and photocopy friendly, none of the proposed color schemes was dropped and all of them were implemented.

Discrete versions of each color scheme are also available. Discrete schemes provide a smaller number of colors for drawing the roads and hence streets with similar intensity are colored the same way. This can help in the traffic analysis by making it easier to identify sections with close traffic (Figure 5.14). In our tool, discrete color schemes have nine colors. More colors makes neighboring shades difficult to distinguish. ColorBrewer also propose discrete schemes with at most nine colors.

User is allowed the option to manually set the maximum value which is associated with the darkest (most right) color on our color schemes. Colors are mapped evenly to values between 0 and *set maximum*. If later we need to obtain a color for value greater than the set maximum then this values is mapped to the last color of the scheme. Figure 5.16 presents traffic intensity for Eindhoven for a whole day, divided into periods of 4 hours. We can see how traffic increases in the morning and decreases in the evening and also see intensity variations for individual streets during the day. Fixing the maximum value can be used to compare traffic hourly, daily, weekly.



(a) 00:00-03:59 (b) 04:00-07:59 (c) 08:00-11:59

(d) 12:00-15:59 (e) 16:00-19:59 (f) 20:00-23:59

Figure 5.16: Traffic intensity for the city of Eindhoven for 24 May 2013 divided into periods of 4 hours. Maximum traffic intensity for the color scheme is fixed at 11,000 vehicles.

As it was mentioned above, some of the colors of the schemes might blend with the colors used on the background map. It is also possible that a user might want a more clean view with a background map being either less visible or missing at all. To solve this we made the background with adjustable transparency. This is done by covering the background map with a white screen and changing its transparency and thereby makes the impression as the background map is disappearing. User can change this parameter to make the desired image (Figure 5.15).

Using colors to encode values proves to give a very good overview of the traffic situation, but analyzing only the colors makes it practically impossible to determine the exact value encoded. In a more thorough analysis it is important to know the numbers that are visualized. This is done by displaying the value which is used for coloring the road just above the measuring locations marker. One small feature which was added upon user request is to draw small arrows on the roads, pointing the direction of traffic. The arrows are drawn on top of the parallel polylines which we color according to traffic intensity. For every section $\overline{AB}$ of

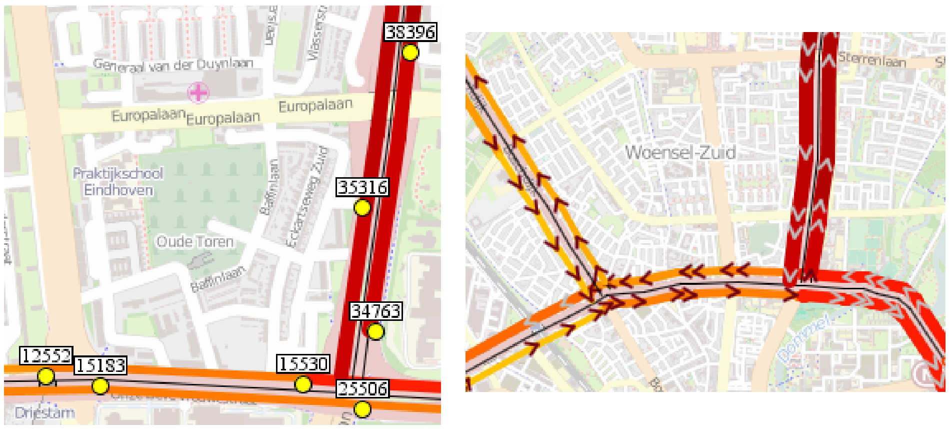


Figure 5.17: Showing the visualized values (left) and drawn arrows on the roads (right).

polyline P an arrow is added on the location of point B, pointing in direction $\overrightarrow{AB}$. To make arrows more visible their color is determined based on the brightness of the color of the street they are drawn over. If it is a light color then arrows are drawn in dark brown and if the color is dark then arrows are in light gray. Colors are selected to be in good contrast with the colors used for the streets.

$$\begin{aligned} brBack = 0.213 * backR + 0.715 * backG + 0.072 * backB \\ if(brBack < 0.5) \\ then \text{ Background is DARK} \\ else \text{ Background is BRIGHT} \end{aligned} \tag{5.3}$$

where $backR$, $backG$, and $backB$ are the Red, Green and Blue components of the color in the range [0 , 1].

The brightness of the background color is determined by summing its RGB components. Because the green component increase color brightness more than red and blue, weight coefficients, adjusted accordingly to RGB components brightness were used.

Drawing measuring locations markers, values and arrows on the image can make it cluttered and difficult to read. Because of that we presented the user with an option to select if he wants them or not (Figure 5.17).

There is one major issue with our visualization approach, illustrated on figure 5.18. If we focus on the top lane (left going traffic) on the section between measuring locations A and B, we can see that this street is uniformly colored. The next color change happens after location B (darker orange, indicating that traffic intensity is larger). One can make the wrong conclusion that the traffic between locations A and B has the same intensity for the whole stretch. It is wrong because we color the street according to the traffic measured in location A which is at the beginning of the section. The red arrows indicate possible entrance and exit locations for the traffic moving on the top lane. As can be seen on the figure, all these entry-exit locations result in increased traffic intensity on *Boschdijk* in location B. Hence users should keep in mind that a stretch from a road is colored according to the measured traffic at its beginning and the result of all vehicle movement on that stretch can be seen on the next measuring location.

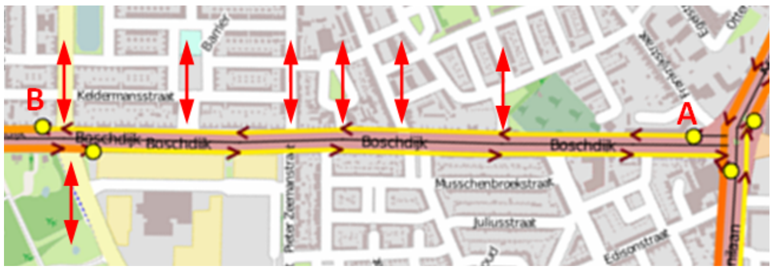


Figure 5.18: Traffic intensity of *">5.6m <12.2m"* vehicle class on a section of *Boschdijk* street. The red double arrows show the possible entrance-exit locations for the traffic on the top lane (going left) between measuring locations A and B.

5.3.2 Visualization of traffic at the measuring locations

The next view which was created for presenting data on a map allows visualization of the collected data on the locations where it was collected (Figure 5.19). Instead of using the calculated values for coloring the road network, here it is used to draw a small bar which parameters depend on the visualized value. The bars are used in a separate view because we want to use colors to present the values at the exact location, where the measurement took place. Using colored bars, over colored roads would result in a messy image.

Calculating the values and mapping them to colors is done the same way as described in the previous section. All visualization bars have the same size, defined by black outlines (Figure 5.20). This makes adjacent bars comparable to each other. The inner solid rectangle (colored orange) is used to visualize a value. Its height and color are variable and depend on the value that is visualized. The color depends on the selected color scheme and if a maximum displaying value is set. The height is calculated via

$$\begin{aligned} &cHeight = \frac{val}{max} * tHeight \\ &if(cHeight > tHeight) \\ &then\ cHeight = tHeight \end{aligned} \tag{5.4}$$

where, *cHeight* is the height of the inner rectangle, *val* is the visualized value, *max* is the maximum value, and *tHeight* is the total height of the Visualization bar.

Drawing the colored bar inside a white rectangle with black borders has several advantages. Firstly the white rectangle provides the same background for all visualized values. This eliminates some optic illusions that occur with identifying a color when it is drawn on different backgrounds. Hence colors that are used for visualization of similar values should be recognized as almost the same. Secondly, as color is used to encode a value, the height of the colored rectangle also encodes the value. Knowing that the total height of the visualization bar is linked to the maximum visualized value (or manually set maximum), the user can see clearly how close this value is to the maximal one. Because it is difficult to determine the exact value from the shades of the color scheme or the height of the inner bar, a number displaying the exact value was added above each visualization bar. Each bar is connected to its corresponding measuring location (Figure 5.20).

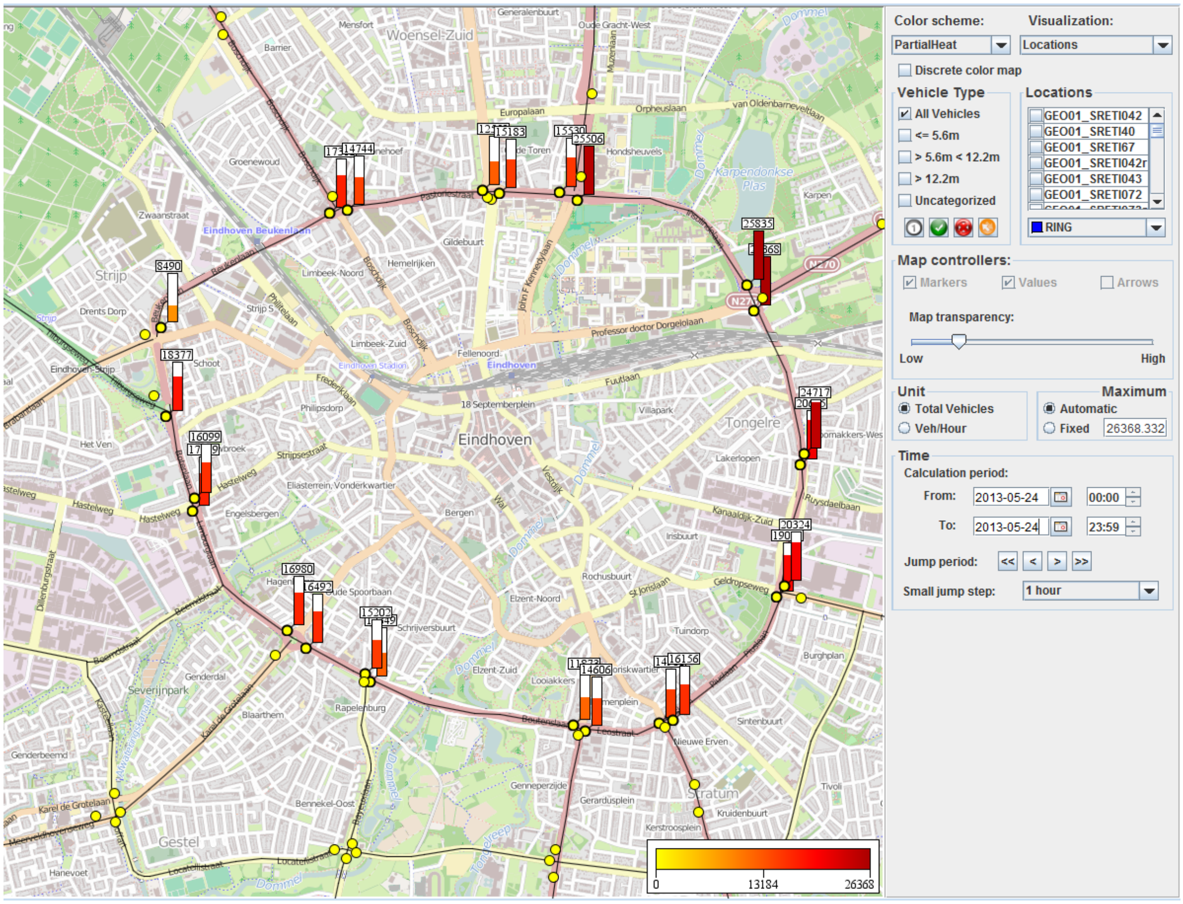

Color scheme: PartialHeat
Visualization: Locations
[ ] Discrete color map
Vehicle Type
[✓] All Vehicles
[ ] <= 5.6m
[ ] > 5.6m < 12.2m
[ ] > 12.2m
[ ] Uncategorized
Locations
[ ] GEO01_SRETI042
[ ] GEO01_SRETI40
[ ] GEO01_SRETI67
[ ] GEO01_SRETI042r
[ ] GEO01_SRETI043
[ ] GEO01_SRETI072
RING
Map controllers:
[✓] Markers [✓] Values [ ] Arrows
Map transparency: Low – High
Unit: [✓ Total Vehicles] Veh/Hour
Maximum: [✓ Automatic] Fixed 26368.332
Time
Calculation period:
From: 2013-05-24 00:00
To: 2013-05-24 23:59
Jump period: << < > >>
Small jump step: 1 hour

Figure 5.19: All traffic intensity for the RING in Eindhoven for 24 May 2013.

5.3.3 Map visualization user interface

Both types of visualization described above can be controlled using the same user interface. Thus users have to learn to work with only one interface saving time and efforts. The interface on figure 5.21 is designed to be organized, simple, clean and reusable. All controllers which have a similar type function or are relevant for making a selection are grouped in panels, shown with a red rectangle and capital letter. Panels C, D and H have already been described in section 5.2. One can notice that Time panel (H) has been modified compared to the one in Section 5.2. We added a *Small jump step* selector. It determines the jump step of "<" and ">" buttons. The other two time jump buttons "<<" and ">>" make a jump step forward and backwards of a complete time interval [$fromTime$; $toTime$]. Location controllers (D) are available for use only during measuring locations data visualization.

Color scheme controllers (A) allow selection of a color scheme and if it has to be discrete or continuous. From the visualization panel (B) the type of visualization can be selected. Because it has a major impact on the generated image and because it unlocks the use of some other controllers it is located at the top of the user interface panel. Panel E allows the selection of additional components on the map and also has a slider that can adjust the transparency of the background map. With the three check boxes the user can select if measuring locations markers, visualized values and direction arrows are shown on the map.

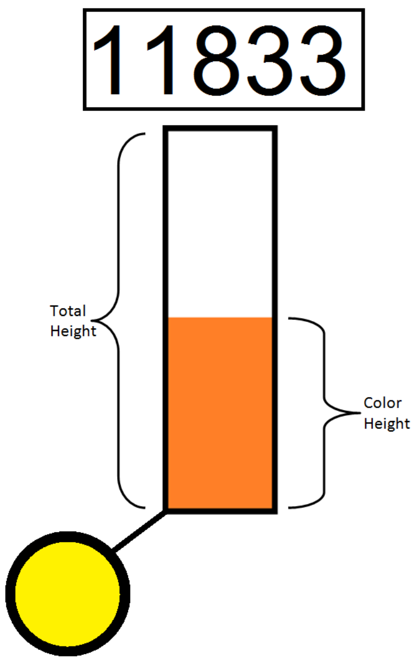
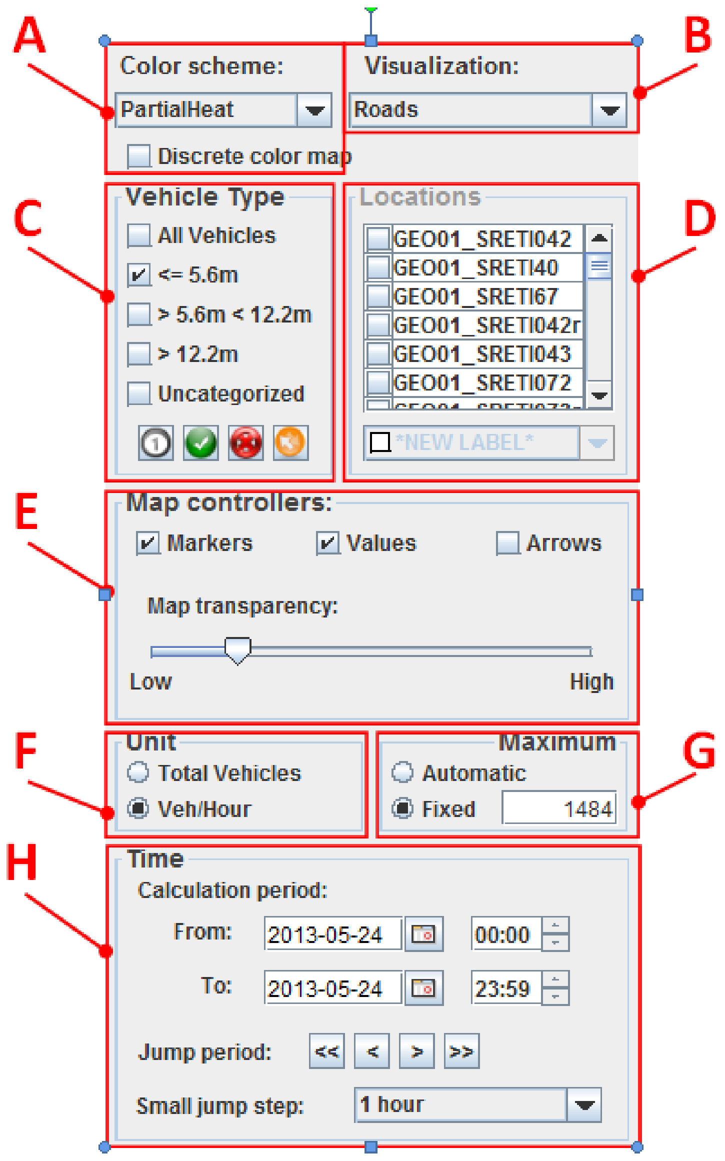


Figure 5.20: Visualization bar

The check boxes are active only during road network data visualization.

The last two presented panels F and H allow the selection of the measuring unit for the displayed values and the maximum for the color the used color scheme. If the option for automatic maximum is selected, then the text field displays the current highest calculated value. If a fixed maximum is selected, then the scheme is adjusted according to the number in the field.



Color scheme: PartialHeat
Discrete color map: ☐
Visualization: Roads

Vehicle Type
☐ All Vehicles
☑ <= 5.6m
☐ > 5.6m < 12.2m
☐ > 12.2m
☐ Uncategorized

Locations
☐ GEO01_SRETI042
☐ GEO01_SRETI40
☐ GEO01_SRETI67
☐ GEO01_SRETI042r
☐ GEO01_SRETI043
☐ GEO01_SRETI072
☐ *NEW LABEL*

Map controllers:
☑ Markers ☑ Values ☐ Arrows
Map transparency: Low — High

Unit
○ Total Vehicles
● Veh/Hour

Maximum
○ Automatic
● Fixed 1484

Time
Calculation period:
From: 2013-05-24 00:00
To: 2013-05-24 23:59
Jump period: << < > >>
Small jump step: 1 hour

Figure 5.21: Map visualization user interface - A) Color scheme controllers, B) Visualization type panel, C) Vehicle type panel, D) Measuring locations panel, E) Map controllers, F) Measuring unit panel, G) Maximum panel, H) Time controllers panel

5.4 Charts visualization

Visualizing information with the help of various type charts is a standard approach in presenting data and giving better understanding. Although most of the charts cannot present the spatial property of data, they can show temporal aspects) in small detail. As most of the researchers in our target group need to answer questions concerning time, we included this type of visualization in our tool. Being one of the most common type of charts, line and bar charts were adapted to suit our needs. Most of our users are experienced in working and reading these type of charts and we believe that this will help our tool being easier to learn and use. Both types of charts require a built city model to use their full functionality.

5.4.1 Line charts data visualization

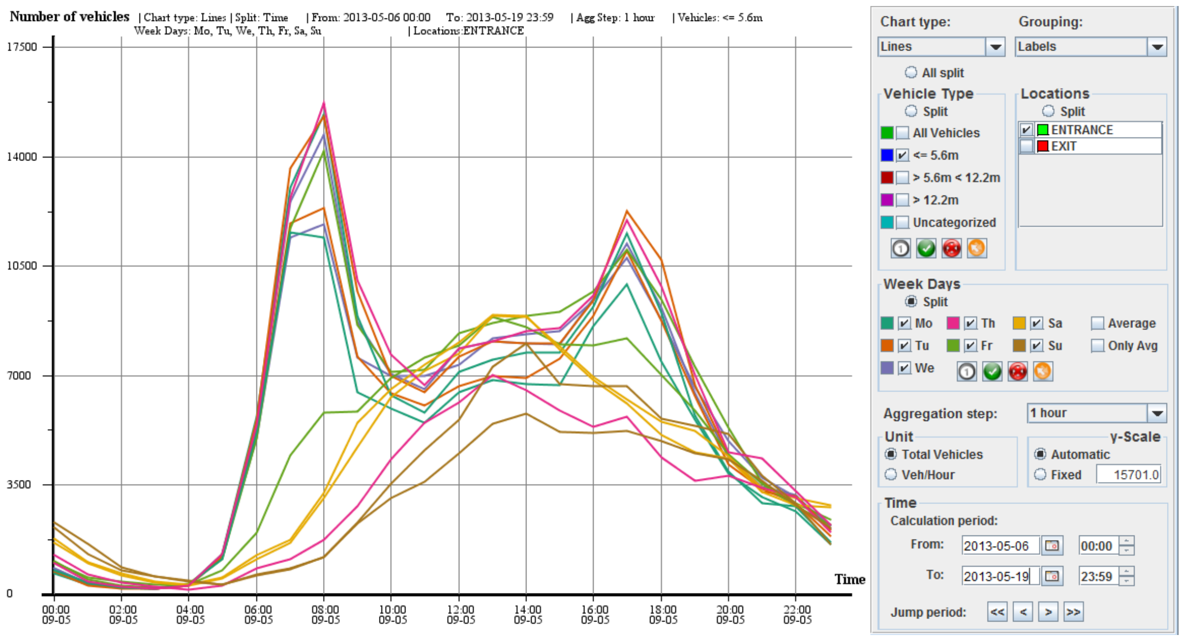

Number of vehicles | Chart type: Lines | Split: Time | From: 2013-05-06 00:00 To: 2013-05-19 23:59 | Agg Step: 1 hour | Vehicles: <= 5.6m
Week Days: Mo, Tu, We, Th, Fr, Sa, Su | Locations:ENTRANCE

Chart type: Lines
Grouping: Labels
All split
Vehicle Type: Split; All Vehicles; ✓ <= 5.6m; > 5.6m < 12.2m; > 12.2m; Uncategorized
Locations: Split; ✓ ENTRANCE; EXIT
Week Days: ● Split; ✓ Mo; ✓ Tu; ✓ We; ✓ Th; ✓ Fr; ✓ Sa; ✓ Su; Average; Only Avg
Aggregation step: 1 hour
Unit: ● Total Vehicles; Veh/Hour
y-Scale: ● Automatic; Fixed 15701.0
Time
Calculation period:
From: 2013-05-06 00:00
To: 2013-05-19 23:59
Jump period: << < > >>

Figure 5.22: Daily variation of traffic intensity of $<= 5.6m$ vehicle class for all measuring locations with label *ENTRANCE* for the period from 6 May 2013 to 19 May 2013. Each line shows data for a day. Lines are colored depending on the day of the week they show.

Line charts in our tool present the variation of one or multiple data streams in time. Each line in the chart represents a list of values. Figure 5.22 show daily traffic intensity data for all *ENTRANCE* locations combined. Each line shows data for a different day.

All map visualizations described in the previous section allow the user to display only one stream of data. This does not provide any possibility to compare different streams and types of data, e.g., we cannot compare traffic intensity between small and medium length vehicles. Charts does not have this issue. To make comparisons it is important to identify on which property the data has to be compared. Our data has three main properties: vehicle class, location and time. Hence the user should be enabled to split the data on any of these properties. Splitting the data on the first two is pretty straight forward as there are a fixed number of classes and locations (either measuring locations or labels) and they are clearly defined. Making a split on the time dimension proved to be more delicate. Data can be divided into hourly data, daily data, monthly, etc... Based on our consultations of *Smart Logistics* lab researchers and based on the requirements which we have derived, we concluded that it is important to make a split based on days and also to be able to identify which day of the week corresponds to each data stream. Hence in our tool we focused on improving the data visualization with a daily time split. Splitting on longer periods of several or more days is left for future work. Despite that, we believe that with the current implementation we are still able to give an answer to almost all of the asked questions. Each visualization image is generated in the following steps:

1. Calculate lists of values that are to be visualized;
2. Find the location of each value on the chart plot;
3. Connect consecutive values from each list with a colored line.

As with the data visualizations on a map here we also start with calculating the values that are to be visualized. The algorithms used for that are very similar to algorithm 1 and are modified versions of it. They differ depending on the selected split dimension. The user is allowed to make a split based on vehicle class, location, day of the week or both vehicle class and day of the week combined. All these four cases have their own specifications which will be mentioned below.

Algorithm 2 Pseudo code for generating values lists making split on vehicle class

```
1: Input: List of locations L, List of vehicle classes V, Time period $T[t_{begin}, t_{end}]$,
2: aggregation step $tAgg$ and flag $vehiclesPerHour$
3: Output: List of list of values "vLists" that are to be visualized.
4: List $vLists$
5: for Vehicle class $v$ in $V$ do
6:     List $bufList$
7:     for location $l$ in $L$ do
8:         Obtain data $D$ for location $l$, vehicle class $v$ and time period $T$
9:         $time = t_{begin}$; $sum = 0$; List $resList$
10:        for line $d$ in $D$ do
11:            if $d_{time} \in [t_{begin}; t_{begin} + tAgg]$ then
12:                $sum \leftarrow sum + d_v$ Note: $d_v$ is the record for vehicle class $v$ in line $d$.
13:            else
14:                if $vehiclesPerHour$ then
15:                    $result \leftarrow sum/totalLines$
16:                else
17:                    $result \leftarrow (sum/totalLines) * (tAgg/60)$
18:                Add result to resList; $t_{begin} = t_{begin} + tAgg$; $sum = 0$
19:        Sum elements from resList to the corresponding elements from bufList
20:    Add vehicle class information to list $bufList$
21:    Add bufList to vLists
```

Calculating the values lists for splitting on vehicle class is done using algorithm 2. It calculates one list for each individual vehicle class selected. In this case we sum the data from the individual locations together. The extra information that is added to the list (line 20) is later used for our visualization purposes (e.g. determine the color of the line). *tAgg* is the aggregation step which is used to aggregate our data.

The algorithms that are used when the user wants to have a split based on location or both vehicle class and location are similar to algorithm 2. The main difference is that when split on location is needed then we sum the data for each of the selected vehicle classes together and if split on both criteria is needed then we do not sum it.

Algorithm 3 Pseudo code for generating values lists making split on days of the week

```
1: Input: List of locations L, List of vehicle classes V, Time period $T[t_{begin}, t_{end}]$,
2: List of week days $W$, aggregation step $tAgg$ and flag $vehiclesPerHour, avg, avgOnly$
3: Output: List of list of values "vLists" that are to be visualized.
4: List<List<Value>> $vLists$
5: for week day $w$ in $W$ do
6:     $t_{buf} = t_{begin}$
7:     List<List<Value>> $wLists$
8:     while $t_{buf} <= t_{end}$ do
9:         if $t_{buf}.getDay() \neq w$ then $t_{buf} = t_{buf} + 1day$; $continue$;
10:        $List < List < Value >> bufList =$ Algorithm 2$(L, V, T, tAgg, vehiclesPerHour)$
11:        $List < Value > resultList = \sum$ corresponding elements of each list in bufList
12:        Add day of week information to list $resultList$
13:        Add resultList to wLists
14:        $t_{buf} = t_{buf} + 1day$
15:    if $avgOnly = false$ then Add all lists from wLists to vLists
16:    if $Avg = true$ then
17:        $n =$ number of elements in wLists
18:        $List < Value > avgList = (\sum$ corresponding elements of each list in wLists$)/n$
19:        Add $avg$ information to list $avgList$
20:        Add avgList to vLists
```

Algorithm 3 calculates one list of values for each selected day in the selected interval. It uses algorithm 2 for some intermediate calculations (line 10). Flag *avg* indicate user's preference if he wants to see also the average values for each group of days e.g. average traffic for all Mondays in a month (lines 16-20). Flag *avgOnly* indicates if the user needs only the average values.

To visualize our calculated lists of values on a line chart we used many of the advices given by Stephen Few in his papers and book (see Bibliography).We paid particular attention to chart size, line size (lines on the plot, axis lines, background lines) and color, number of labels on X and Y axis and legend.

Some of the visualized values have a small label below them (and below the x-Axis) showing the time stamp of the visualized value. For convenience we try to put labels for as many values as possible. Having in mind that there is only limited number of labels that can be located along the xAxis without overlapping we use formula 5.5. We put a label under the first visualized value and then we check for the rest of the values. If there is enough room for a new label we put one. This can result in our last point to be left without a label. We believe that this will not be a big issue in reading the graph as the user knows that the last value is always at the end of the selected time interval.

$$\begin{aligned} &space = \frac{xWidth}{n-1} * (j-i)\ , i \in [1,n], j \in [2,n] \\ &if(space > labelWidth) \\ &then\ Put\ label\ below\ point\ p_j. \end{aligned} \tag{5.5}$$

Where n is the number of values in a list, i: is the index of the last point p_i which has a label,

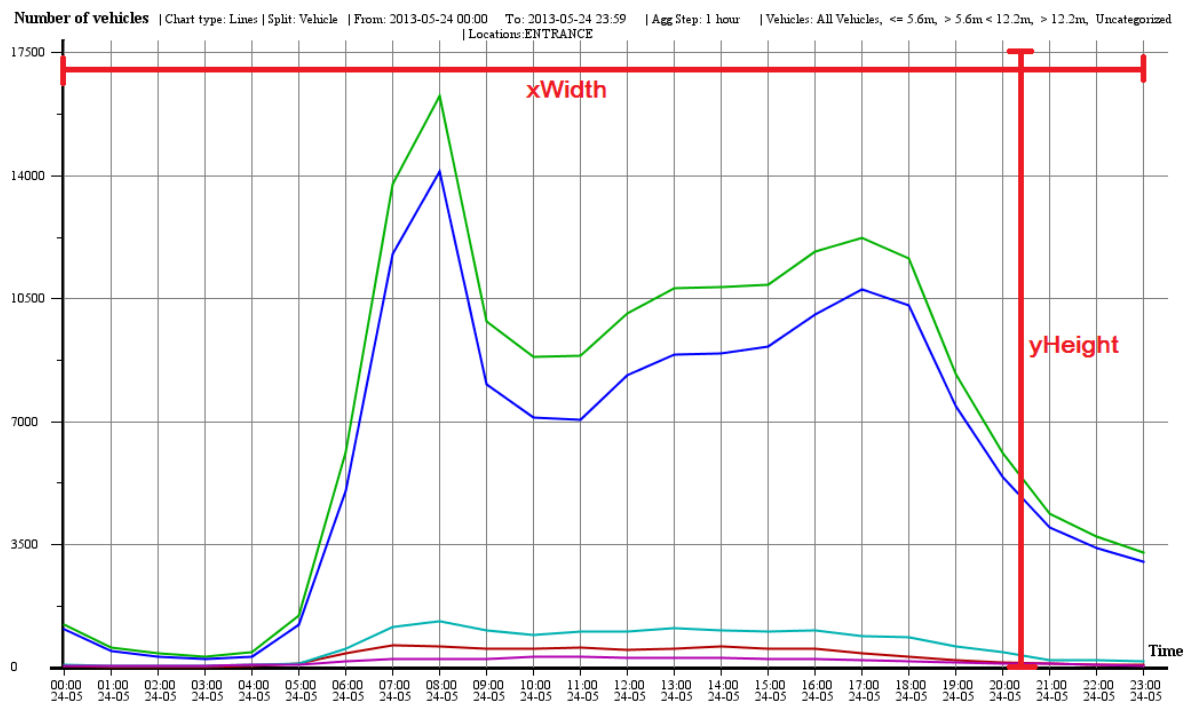


Figure 5.23: Traffic intensity at all locations with label *ENTRANCE* for 24-05-2014. Split is made on vehicle class type. Green line represents *All vehicles* class, blue - $<= 5.6m$, dark red - $> 5.6m <= 12.2m$, purple - $> 12.2m$, and cyan - $> 5.6m <= 12.2m$. *xWidth* and *yHeight* show the size of the working plot, used for drawing the visualization polylines.

and $xWidth$ is the width of the drawing plot.

Putting labels on y-Axis of the chart is done in a different way. We use a fixed number of labels, evenly spaced, presenting values from the [0,max] interval. In our tool six labels are used (Figure 5.23). We believe that this number is sufficient to give a good overview about the values used for drawing the polylines. If the exact value in each point is needed, then other mechanisms are available for that. To determine the max value that can be presented on the chart we use a slightly different approach than just finding our largest calculated value. We apply algorithm 4 to find it and also determine the y-Axis labels.

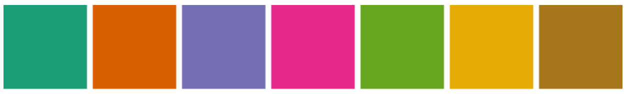
The algorithm calculates max value in such a way that the interval between two neighboring labels has a size multiple of 5. This makes it easier to obtain a value from the chart and also produces a better looking and easier to understand image. To make it easier to identify the value and the time stamp from the lines on the graph, thin gray lines were added at the background, each one running from a x or y-Axis label (Figure 5.23).

To paint the lines on the chart we need to use colors that are easily distinguishable from each other. Steven Few recommends the following twelve colors: red, green, yellow, blue, black, white, pink, cyan, gray, orange, brown and purple. If more are needed then it is possible to use darker or lighter versions of the colors. From these colors we already use gray, white and black to draw the background and axis lines. ColorBrewer also gives examples of several qualitative color schemes. Based on these recommendations we designed three color schemes that we use depending on where the user makes a data split (Figure 5.24). Colors

Algorithm 4 Calculating y-Axis labels

```
1: Input: Number of labels $nrLabels, minValue, maxValue$
2: Output: $List\ yLabels,\ max$
3: $range = maxValue - minValue$
4: $unrTickS = \frac{range}{nrLabels-1}$
5: $x = \lceil log_{10}(unrTickS) - 2 \rceil$
6: $pow10x = 10^x$
7: $rTickS = \lceil \frac{unrTickS}{pow10x*5} \rceil * pow10x * 5$
8: $max = (nrLabels - 1) * rTickS$
9: for i=0 : nrLabels-1 do
10:     label = minValue+i*rTickS
11:     Add label to yLabels;
```

(a) Split on vehicle class


(b) Split on day of the week

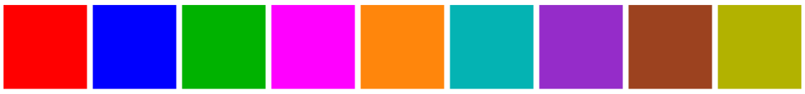
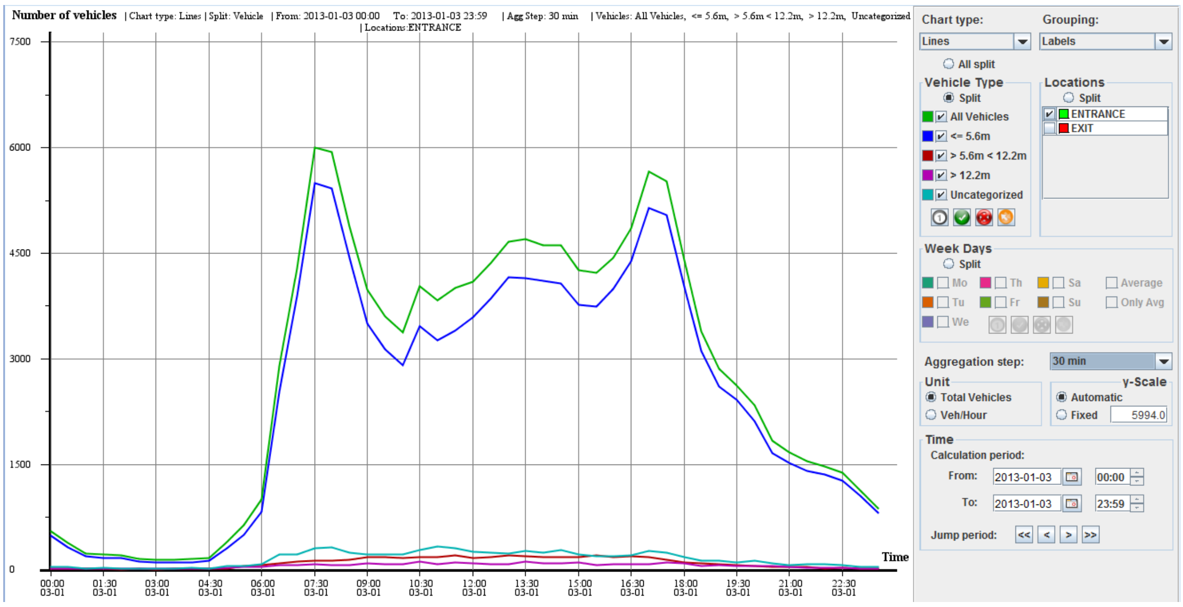

(c) Split on location

Figure 5.24: Default color schemes that are used for coloring the lines in our line chart.

from scheme a) are assigned to each vehicle class and the colors from scheme b) to each day of the week.

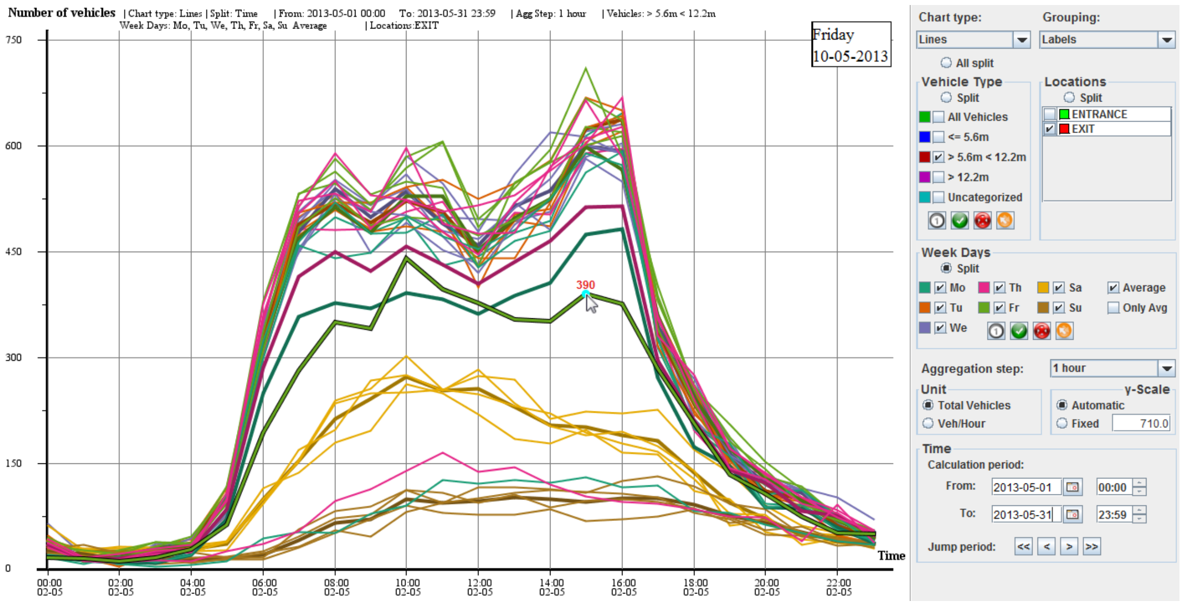
If a split is made based on vehicle class, then lines are drawn using the colors assigned to each class (Figure 5.25), if a split is made based on days of the week then the colors assigned to the days are used (Figure 5.26). The user can also select to see the lines presenting average traffic data for each week day (e.g. average traffic intensity for Mondays). The lines that show the average traffic are drawn using a darker version of the same colors used for drawing each day. These lines are also made thicker in order to be more noticeable. On figure 5.26 Saturday traffic is shown in yellow and the average traffic in dark yellow. We distinguish two cases in splitting data based on location. In the first one, each measuring location is used to create a line for the chart. In the second case we use aggregated data from several locations, assigned with the same label. Each group (label) of locations is used for one line on the chart. When we split on measuring location, then each line is assigned color from scheme c) from figure 5.24. If there are more locations selected than colors in the scheme, then the colors are reused over and over again. Figure 5.27a has twelve lines, while the scheme has only nine colors. In this case red, blue and green are used for two lines. If the user splits on label, then the color assigned to each label is used. Assigning a color to a label is done in the creation of the city model. It is important for the user to select colors such that lines



Number of vehicles | Chart type: Lines | Split: Vehicle | From: 2013-01-03 00:00 To: 2013-01-03 23:59 | Agg Step: 30 min | Vehicles: All Vehicles, <= 5.6m, > 5.6m < 12.2m, > 12.2m, Uncategorized | Locations:ENTRANCE

Chart type: Lines
Grouping: Labels
All split
Vehicle Type: [✓ Split]
🟩 ✓ All Vehicles
🟦 ✓ <= 5.6m
🟥 ✓ > 5.6m < 12.2m
🟪 ✓ > 12.2m
🟦 ✓ Uncategorized
Locations: Split
✓ 🟩 ENTRANCE
🟥 EXIT
Week Days: Split
Mo, Tu, We, Th, Fr, Sa, Su, Average, Only Avg
Aggregation step: 30 min
Unit: [✓ Total Vehicles] Veh/Hour
y-Scale: [✓ Automatic] Fixed 5994.0
Time
Calculation period:
From: 2013-01-03 00:00
To: 2013-01-03 23:59
Jump period: << < > >>

Figure 5.25: Traffic intensity at all locations with label *ENTRANCE* for 03-01-2013. A split is made on vehicle class type. Each polyline represent a vehicle class.



Number of vehicles | Chart type: Lines | Split: Time | From: 2013-05-01 00:00 To: 2013-05-31 23:59 | Agg Step: 1 hour | Vehicles: > 5.6m < 12.2m
Week Days: Mo, Tu, We, Th, Fr, Sa, Su Average | Locations:EXIT

Friday
10-05-2013

Chart type: Lines
Grouping: Labels
All split
Vehicle Type: Split
🟩 All Vehicles
🟦 <= 5.6m
🟥 ✓ > 5.6m < 12.2m
🟪 > 12.2m
🟦 Uncategorized
Locations: Split
🟩 ENTRANCE
✓ 🟥 EXIT
Week Days: [✓ Split]
✓ Mo, ✓ Tu, ✓ We, ✓ Th, ✓ Fr, ✓ Sa, ✓ Su, ✓ Average, Only Avg
Aggregation step: 1 hour
Unit: [✓ Total Vehicles] Veh/Hour
y-Scale: [✓ Automatic] Fixed 710.0
Time
Calculation period:
From: 2013-05-01 00:00
To: 2013-05-31 23:59
Jump period: << < > >>

Figure 5.26: Traffic intensity at all locations with label *EXIT* for May 2013. A split is made on days of the week. Each line represents traffic data for one day. Thicker darker lines show the average data for the same week days. Positioning the mouse over a line results in highlighting it, showing a box with information about the line. Positioning the mouse over a point with a value results in showing the value there.

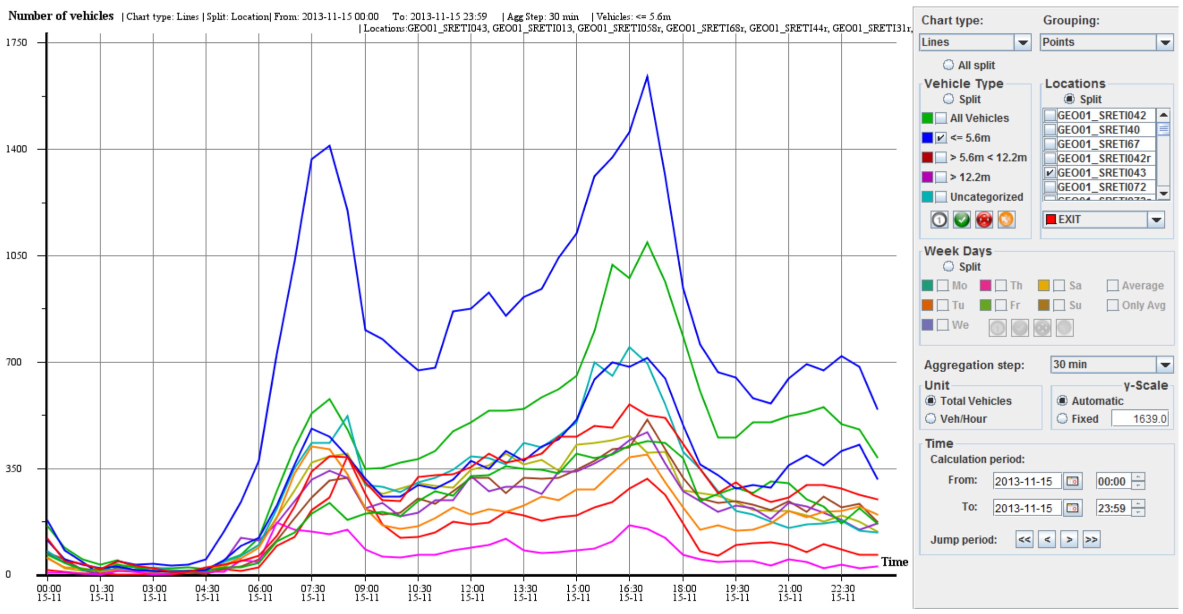

Number of vehicles | Chart type: Lines | Split: Location| From: 2013-11-15 00:00 To: 2013-11-15 23:59 | Agg Step: 30 min | Vehicles: <= 5.6m
| Locations:GEO01_SRETI043, GEO01_SRETI013, GEO01_SRETI058r, GEO01_SRETI68r, GEO01_SRETI44r, GEO01_SRETI31r,

Chart type: Lines
Grouping: Points
Vehicle Type: [✓ <= 5.6m]
Locations: Split [✓ GEO01_SRETI043]
EXIT
Aggregation step: 30 min
Unit: [✓ Total Vehicles]
y-Scale: [✓ Automatic] Fixed 1639.0
Time
Calculation period:
From: 2013-11-15 00:00
To: 2013-11-15 23:59

(a) Each line presents a data from one measuring location.

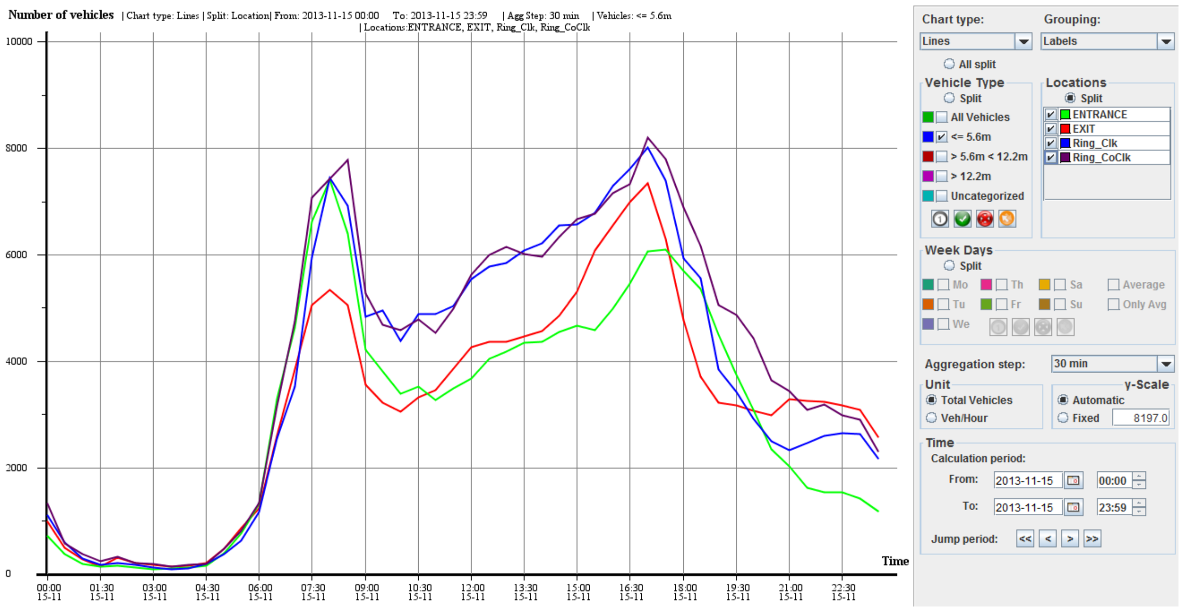
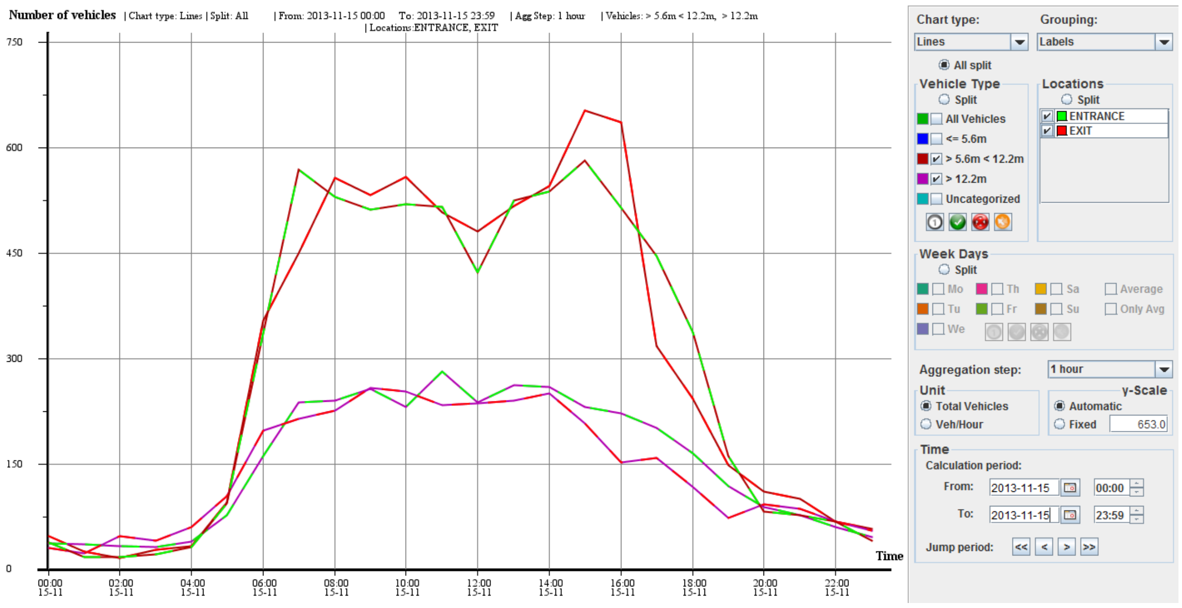

Number of vehicles | Chart type: Lines | Split: Location| From: 2013-11-15 00:00 To: 2013-11-15 23:59 | Agg Step: 30 min | Vehicles: <= 5.6m
| Locations:ENTRANCE, EXIT, Ring_Clk, Ring_CoClk

Chart type: Lines
Grouping: Labels
Vehicle Type: [✓ <= 5.6m]
Locations: Split [✓ ENTRANCE] [✓ EXIT] [✓ Ring_Clk] [✓ Ring_CoClk]
Aggregation step: 30 min
Unit: [✓ Total Vehicles]
y-Scale: [✓ Automatic] Fixed 8197.0
Time
Calculation period:
From: 2013-11-15 00:00
To: 2013-11-15 23:59

(b) Each line presents data for all measuring locations having the same label.

Figure 5.27: Traffic intensity at A) all locations with label *EXIT*, B) labels *ENTRANCE*, *EXIT*, *Ring_Clk*, and*Ring_CoClk* for 15-11-2013. Split is made on location.

are easily distinguishable. The last case is when the split is made on combination of vehicle class and location. In this case we draw one line for each combination of the selected classes and locations. Here each line is painted in dashed style with two colors (Figure 5.28): one color comes from the vehicle class and one color from the selected location. Here it is also important to select colors that make the chart easy to understand. On the figure can be seen that the combination from red($EXIT$) and dark red ($> 5.6m <= 12.2m$) is not a good one.



Number of vehicles | Chart type: Lines | Split: All | From: 2013-11-15 00:00 To: 2013-11-15 23:59 | Agg Step: 1 hour | Vehicles: > 5.6m < 12.2m, > 12.2m | Locations:ENTRANCE, EXIT

Chart type: Lines
Grouping: Labels
[✓ All split]

Vehicle Type
Split
🟩 All Vehicles
🟦 <= 5.6m
🟥 ✓ > 5.6m < 12.2m
🟪 ✓ > 12.2m
🟦 Uncategorized

Locations
Split
✓ 🟩 ENTRANCE
✓ 🟥 EXIT

Aggregation step: 1 hour

Unit: [✓ Total Vehicles] Veh/Hour

y-Scale: [✓ Automatic] Fixed 653.0

Time
Calculation period:
From: 2013-11-15 00:00
To: 2013-11-15 23:59

Figure 5.28: Traffic intensity for $> 5.6m <= 12.2m$ and $> 12.2m$ vehicle classes at all locations with label *ENTRANCE* and *EXIT* for 15-11-2013. Split is made on vehicle class type and location. Each polyline shows data for one vehicle class and one location.

The proposed color schemes are used by default, but the user can always change the colors assigned to the vehicle classes, week days and labels.

To increase the readability of the chart and to aid data analysis, some user interaction is implemented. If the user positions the mouse cursor over a line, this will result in drawing the line highlighted: thicker line with black outlines, and also to be positioned on top of the other lines. A small box is drawn, showing information about the line. Depending on the selected split, information can vary between the vehicle class that is represented, the day of the week or the location name/label. If the cursor is positioned over a value point then the value is shown above the point (Figure 5.26).

5.4.2 Bar charts data visualization

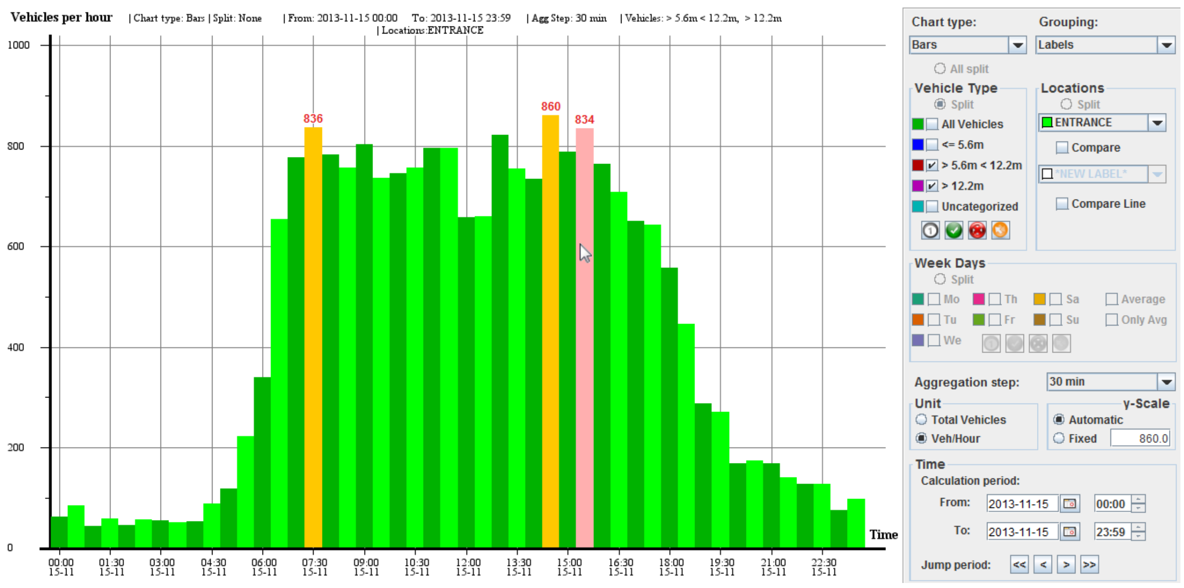

Vehicles per hour | Chart type: Bars | Split: None | From: 2013-11-15 00:00 To: 2013-11-15 23:59 | Agg Step: 30 min | Vehicles: > 5.6m < 12.2m, > 12.2m | Locations:ENTRANCE

Chart type: Bars — Grouping: Labels
Vehicle Type: [✓ Split]; All Vehicles; <= 5.6m; [✓ > 5.6m < 12.2m]; [✓ > 12.2m]; Uncategorized
Locations: ENTRANCE; Compare; *NEW LABEL*; Compare Line
Week Days: Mo, Tu, We, Th, Fr, Sa, Su, Average, Only Avg
Aggregation step: 30 min
Unit: Total Vehicles [✓ Veh/Hour]
y-Scale: [✓ Automatic] Fixed 860.0
Time — Calculation period: From: 2013-11-15 00:00 To: 2013-11-15 23:59
Jump period: << < > >>

Figure 5.29: Bar chart presenting daily traffic intensity for vehicles longer than 5.6 meters at *ENTRANCE* labeled locations on 15-11-2013.

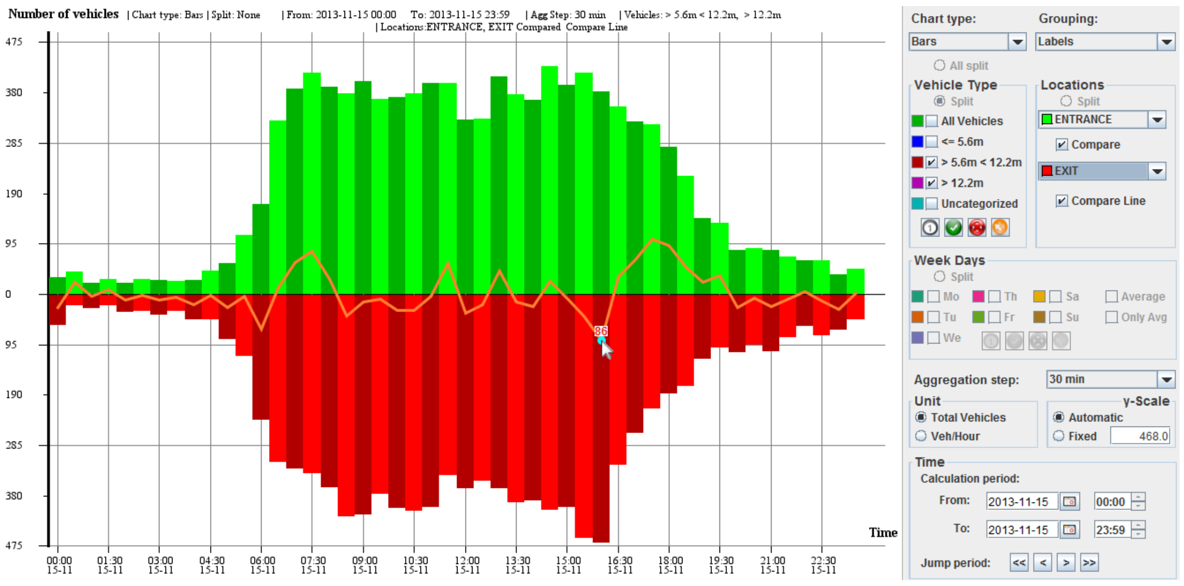

Number of vehicles | Chart type: Bars | Split: None | From: 2013-11-15 00:00 To: 2013-11-15 23:59 | Agg Step: 30 min | Vehicles: > 5.6m < 12.2m, > 12.2m | Locations:ENTRANCE, EXIT Compared Compare Line

Chart type: Bars — Grouping: Labels
Vehicle Type: [✓ Split]; All Vehicles; <= 5.6m; [✓ > 5.6m < 12.2m]; [✓ > 12.2m]; Uncategorized
Locations: ENTRANCE; [✓ Compare]; EXIT; [✓ Compare Line]
Week Days: Mo, Tu, We, Th, Fr, Sa, Su, Average, Only Avg
Aggregation step: 30 min
Unit: [✓ Total Vehicles] Veh/Hour
y-Scale: [✓ Automatic] Fixed 468.0
Time — Calculation period: From: 2013-11-15 00:00 To: 2013-11-15 23:59
Jump period: << < > >>

Figure 5.30: Bar chart comparing traffic intensity for vehicles longer than 5.6 meters between *ENTRANCE* and *EXIT* labeled locations on 15-11-2013. The orange polyline shows the difference between the top and bottom bars.

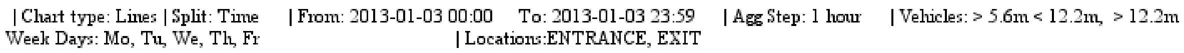
The second type of charts available in our tool is bar chart. By design it shows only one or two lists of values. Calculating the values for display is done in the same way as calculating the values for line charts. The only difference is that user cannot split the data on any criteria. Figure 5.29 shows the daily traffic intensity for vehicles longer than 5.6 meter at *ENTRANCE* locations. The color of the bars is determined by the color assigned to the visualized label data. If we choose to show traffic from randomly selected measuring locations then bars are painted in one of the color from scheme 5.24c. To make neighboring bars easier to distinguish we gave every other bar a darker shade. If the user wants to see the exact value represented in a bar, then the mouse cursor should be positioned over it. If the user wants to keep seeing the value, then he can click on the bar.

Bar charts allow the option to compare traffic data from two different groups of measuring points (Figure 5.30). The main advantage over our line charts is that it draws a line showing the difference between the corresponding values. This is for instance useful to compare incoming and outgoing traffic.

5.4.3 Charts visualization user interface

The user interface for charts visualization is to a great extent similar to the one used for map visualizations (Figure 5.32). This can help the user to adapt faster and make it easier to use the application. Panels C, D, F, G and H have the same functionality as in the map visualizations user panel. There are a few differences though. The small *split* radio buttons allow the user to select where he wants to make the data split. The two versions of panel D are used respectively for line charts and bar charts. The bar charts panel has also controllers which allow the selection of the compare mode. Panel E allows the selection of week days. It is active only if the user select to make split there and in line charts visualization. Both panels C and E has small colored buttons in front of every check box. The colors of the button represents the color binded to that option. Showing the colors there can help the user interface panel act as a sort of legend for the chart. The two top panels A and B allow the selection of chart type (lines or bar) and grouping of the data - either using labeled locations or individual locations. The *aggregation step* allows the selection of a time interval that is used to aggregate the data. The aggregation step also determines the jump period of the "<" and ">" button of time panel H.

Every selection made by the user is shown in a small legend on the top of each chart. It can be used firstly to understand what is shown on the image but also to recreate it if needed in the future.


| Chart type: Lines | Split: Time | From: 2013-01-03 00:00 To: 2013-01-03 23:59 | Agg Step: 1 hour | Vehicles: > 5.6m < 12.2m, > 12.2m
Week Days: Mo, Tu, We, Th, Fr | Locations:ENTRANCE, EXIT

(a) Example of legend in lines type chart.

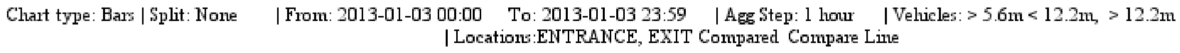


Chart type: Bars | Split: None | From: 2013-01-03 00:00 To: 2013-01-03 23:59 | Agg Step: 1 hour | Vehicles: > 5.6m < 12.2m, > 12.2m
| Locations:ENTRANCE, EXIT Compared Compare Line

(b) Example of legend in bar type chart.

Figure 5.31: Examples of line and bar chart legends

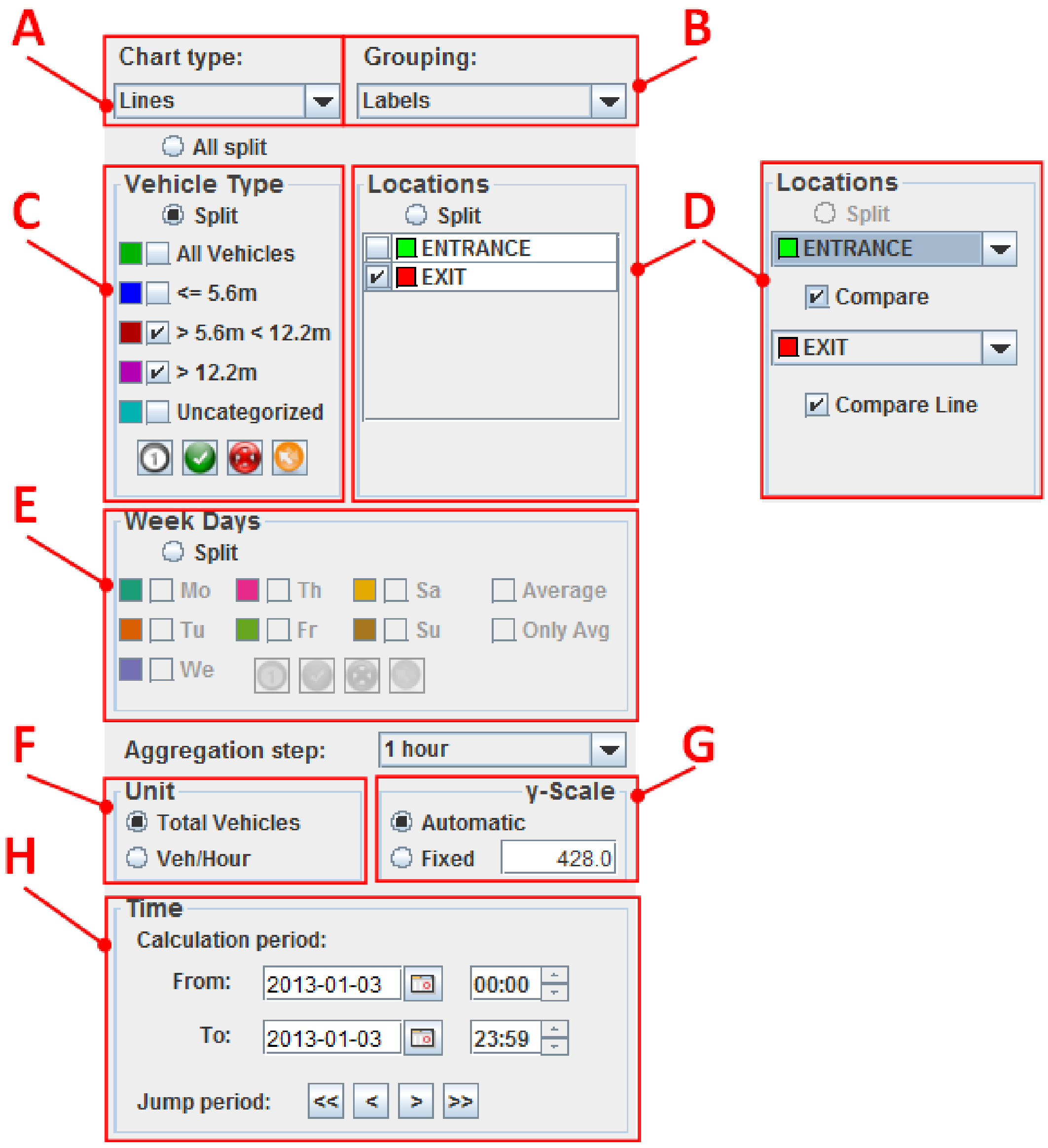


Chart type: Lines

Grouping: Labels

All split

Vehicle Type
[✓ Split]
🟩 All Vehicles
🟦 <= 5.6m
🟥 ✓ > 5.6m < 12.2m
🟪 ✓ > 12.2m
🟦 Uncategorized

Locations
Split
🟩 ENTRANCE
✓ 🟥 EXIT

Locations
Split
🟩 ENTRANCE
✓ Compare
🟥 EXIT
✓ Compare Line

Week Days
Split
Mo Th Sa Average
Tu Fr Su Only Avg
We

Aggregation step: 1 hour

Unit
[✓ Total Vehicles]
Veh/Hour

y-Scale
[✓ Automatic]
Fixed 428.0

Time
Calculation period:
From: 2013-01-03 00:00
To: 2013-01-03 23:59
Jump period: << < > >>

Figure 5.32: charts visualization user interface - A) Charts type, B) Visualization type panel, C) Vehicle type panel, D) Measuring locations panels, E) Week days panel, F) Measuring unit panel, G) Maximum y-Scale panel, H) Time controllers panel

Chapter 6

System architecture

In this chapter we describe the architecture of our system and how the separate components are connected together. One of our goals is to make things as simple as possible and to automate the process as much as possible, hence reducing the extra work for the user.

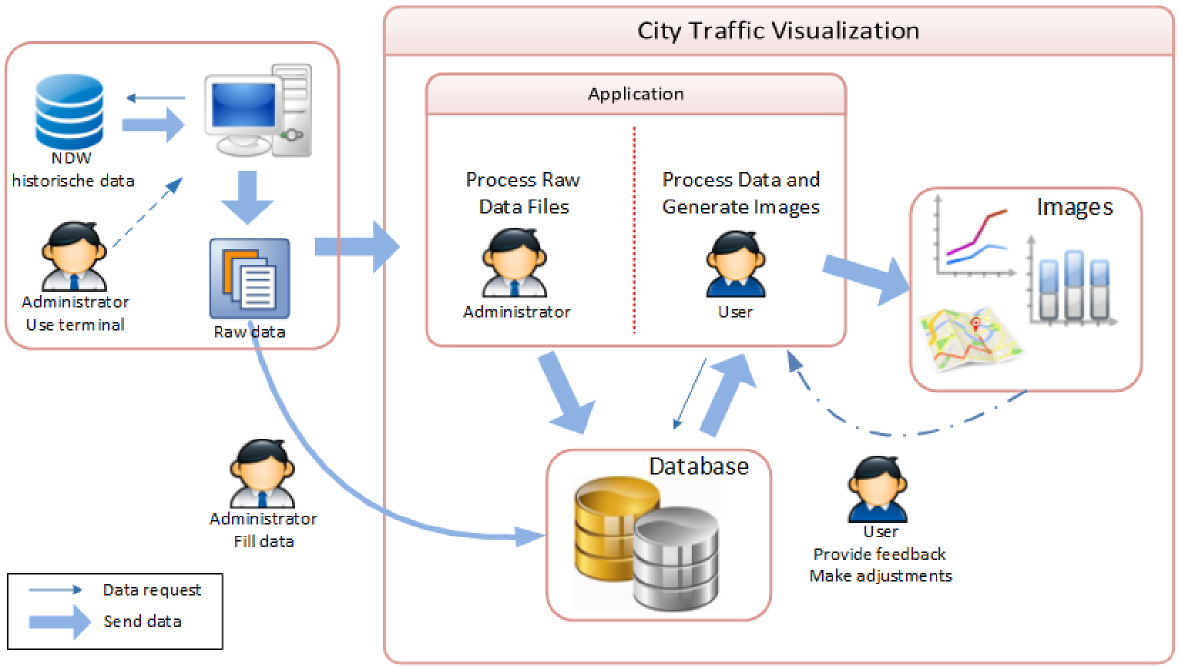


Figure 6.1: System architecture and data flow diagram.

Downloading raw data from NDW is the only process that has to be done manually. At the time of writing this thesis NDW historical traffic data servers[1] do not provide an API (Application Programming Interface) for automating the task of obtaining data. The system administrator has to use working station with internet access to log in to the NDW system. Using their provided web interface, the administrator has to specify what data he wants to download. After the request has been processed, the administrator can download the data

[1]http://ndw.dysi.nl

back to his working station. Here we have to note that with the use of the term *system administrator* we mean any user who has the privileges and credentials to access and work with NDW servers.

The city traffic visualization system has two main components: a data storage module (database) and data processing module (software application). The data storage component is used to store the processed traffic data (described in Chapter 4). In our final implementation we set up a PostgreSQL database for our data storage module. The original implementation was done using .csv files to store the data. Each table described in Chapter 4 is saved in a separate file. The main advantage is that it is simpler to set up, as the user does not have to install and tune a database and also that the files are stored directly on the users machine, hence always available. Unfortunately, the increasing size of the files leads to a decrease in performance of our application. Because of that we considered a database approach. A few speed tests using both a file and a database storage system were done. We processed two different size chunks of raw data: the first one containing traffic data from all measuring locations in the city of Eindhoven for May 2013 and the second one with the same type of data but for the whole 2013. The task was to generate the image on Figure 5.8. Using the file storage system with monthly data, calculating the image was done in between 3 and 4 seconds. With a yearly data image was done in between 26 and 30 seconds. Running the same test using a database located on the testing machine resulted in presenting the image in around 3 seconds, regardless of the raw data size. We set up a database on a remote server and this slowed down the system as the visualization was done in 12 seconds, no matter the size of the database. In this case the performance speed depends on the size of the requested data and the connection speed between the database server and the user's work station. Hence using a database as storage system does not always result in a major speed up. Despite this, we kept this solution because of another advantage. Once a database server is created, it can be used by many users simultaneously, each one having his own workstation.

The application is developed using Java and has two main functions: one is to process raw data downloaded from NDW and store it in the database, and the other one is to visualize the stored data using various techniques. The processing part (chapter 4) is intended to be used by the system administrator or any other user who has credentials to write in the database. As mentioned before, some of the data files has to be manually uploaded to the database.

The visualization part of our application allows the researcher to use the techniques from Chapter 5 to generate clear and meaningful images. Researchers can inspect these pictures and change visualization settings to offer more insight.

Chapter 7

Conclusions

In this thesis we presented our interactive prototype for visualizing NDW historical data. Proper visualization images can be used by researchers to understand and improve the traffic of the cities in the Netherlands. Below we present our conclusions and some recommendations for future work and improving our work.

We started this project with a few general requirements and several research questions that had to be answered. We tried to specify our goals and address each challenge. Every design decision which has been taken is done to make our application meet with the requirements.

Working with big data always brings some limitations to the developed products. The main problems come from limitation in the memory storage capacity, performance speed and limitations of visualization techniques. NDW provide data with lots of duplicating records. As shown in chapter 4, the useful and unique information of the raw data downloaded from the databank can be less than 6%. We used some simple techniques to remove the extra data and summarize the remainder into tables. Our approach makes it easier to access, search and fetch the needed information.

Visualizing all available data in one image is practically impossible. Hence we designed a number of views that address various properties of the data in more details. Visualizing traffic data over a road map allows the user to get a spatial view of the traffic status. Analyzing this type of images make it fast and easy to identify the main traffic arteries and their state during the different periods of the day or year.

A number of charts has been developed for visualizing the traffic. They can present the time variance of the data in detail. Researchers can use them to compare the traffic in different locations or compare the traffic for different time periods. The user is provided with the ability to select and filter the data which he wants to see. We followed some of the advices from related work to create clear and meaningful visualizations.

As our data can be explored in various ways and the usefulness of the visualization depends on the research questions, we leave it to the user to make adjustments, and we provided several options to control time periods, chart types, colors and other map properties. These encourage users to explore the data with our tool. Our work has been focused in visualizing NDW data and we optimized our techniques to show its properties. Therefore our approach might not be applicable to other types of data.

The developed system is divided into a number of modules. Using a database to store the data leads to a scalable solution, allowing several users to work simultaneously with the same data.

7.1 Future work

After the completion of our work, there are several issues that remained open for future work. Some of them have higher priority than others. These are:

- Increase the performance speed of the application. We tried to speed up some of our computations by parallelizing algorithm 3, but we believe that more optimizations can be done in this area. The second major issue concerns the quality of the fetch data requests which are sent to the database. As we stated, the speed depends on the size of the fetched data and the connection speed. We believe that moving some of the calculations towards the database by improving the request can speed up the whole process.
- Deal with incompleteness of the data. There are many "holes" in the data caused by malfunctioning of the measuring stations. At the moment we do not apply any data mining techniques to clean it.
- Apart from cleaning the data, there are other clustering algorithms to identify patterns in the data. This can be used to find days with similar traffic.
- We wanted to implement several more visualization techniques, mainly connected with comparing data, making a calendar view (van Wijk [22]). Using other type of charts can also provide new insights for the user.

Finally, our current visualization approach can be improved by providing the user more interaction and options to better tune the data selection. Moreover, implementing a search option would help users to find faster measuring locations or roads, based on the search criteria and the meta data they provide (location name, street name, number of lanes, etc...).

Bibliography

[1] Onderweg, http://www.onderweg.nl/

[2] Aigner, W., Miksch, S., Schumann, H., Tominski, C.: Visualization of Time-Oriented Data. Springer, London Dordrecht Heidelberg New York (2011)

[3] Andrenko, G., Andrenko, N.: Interactive Maps for Visual Data Exploration. International Journal of Geographical Information Science pp. 355–374 (1999)

[4] Andrenko, G., Andrenko, N., Demsar, U., Dransch, D., Dykes, J., Fabrikant, S., Jern, M., Kraak, M., Schumann, H., Tominski, C.: Space, Time and Visual Analytics. International Journal of Geographical Information Science pp. 1577–1600 (2010) 8

[5] Andrenko, G., Andrenko, N., Wrobel, S.: Visual Analytics Tools for Analysis of Movement Data. ACM SIGKDD Explorations Newsletter - Special issue on visual analytics pp. 38–44 (December 2010) 7, 8

[6] Andrenko, N., Andrenko, G., Gatalsky, P.: Supporting Visual Exploration of Object Movement. In: Proceedings of the working conference on Advanced visual interfaces AVI '00. pp. 217–220 (2000) 28

[7] Andrenko, N., Andrenko, G., Voss, H., Bernardo, F., Hipolito, J., Kretchmer, U.: Testing the Usability of Interactive Maps in CommonGIS. Cartography and Geographic Information Science pp. 325–342 (2002)

[8] Andrienko, G., Andrienko, N., Bak, P., Keim, D., Wrobel, S.: Visual Analytics of Movement. Springer, Heidelberg New York Dordrecht London (2013) 7

[9] Andrienko, N., Andrienko, G.: Exploratory Analysis of Spatial and Temporal Data. Springer, Germany (2006)

[10] Colorbrewer 2.0: Color advice for cartography, http://colorbrewer2.org/

[11] Few, S.: Table and Graph Design for Enlightening Communication, perceptual Edge 8

[12] Few, S.: Effectively Communicating Numbers: Selecting the Best Means and Manner of Display (November 2005), perceptual Edge 8

[13] Few, S.: Show Me the Numbers: Designing Tables & Graphs to Enlighten. Analytics Press, Oakland (2012) 8

[14] Few, S., Edge, P.: Line Graphs and Irregular Intervals: An Incompatable Partnership. Visual Business Inteligence Newsletter (November/December 2008) 8

[15] Few, S., Edge, P.: Practical Rules for Using Color in Charts. Visual Business Inteligence Newsletter (February 2008)

[16] Google: Live Traffic, http://en.wikipedia.org/wiki/Google_Traffic 9

[17] INRIX: Inrix Traffic, http://www.inrixtraffic.com/

[18] Jankowski, P., Andrenko, N., Andrenko, G.: Map-centered Exploratory Approach to Multiple Criteria Spatial Decision aking. International Journal of Geographical Information Science pp. 100–127 (2001)

[19] Karnick, P., Cline, D., Jeschke, S., Razdan, A., Wonka, P.: Route Visualization using Detail Lenses. In: IEEE Transactions on Visualization and Computer Graphics. pp. 235–247 (March-April 2010) 9

[20] Salvatore, R., Pedreschi, D., Nanni, M., Giannotti, F., Angrenko, N., Andrenko, G.: Visually Driven Analysis of Movement Data by Progressive Clustering. In: Information Visualization. pp. 225–239 (September 2008)

[21] Smart Logistics: Smart Mobility - Intelligent and Productive Mobility and Transport, https://www.youtube.com/watch?v=rC7JFijh_YI 3

[22] Wijk, J., Selow, E.: Cluster and Calendar based Visualization of Time Series Data. In: Information Visualization, 1999. (Info Vis '99) Proceedings. 1999 IEEE Symposium on. pp. 4–9 (October 1999) 58

[23] Willems, N.: Visualization of Vessel Traffic. Ph.D. thesis, Eindhoven University of Technology (2011) 9

Appendix A

Administrator Manual

This manual is intended to help system administrators set up the database, fill up the raw data and prepare everything for work. Some of the steps here are optional and can be skipped or done using a different approach. What we describe here is an exemplary manual for getting started.

A.1 Installing and setting up a database

The first thing to do is to download and install the required software. For our implementation we choose to work with the PostgreSQL object-relational database management system with PostGIS extension pack.

1. Download the latest version of PotgreSQL from `http://www.postgresql.org/download/`. Make sure that you got the package suitable for your operating system.

2. Run the downloaded file and install PostgreSQL.

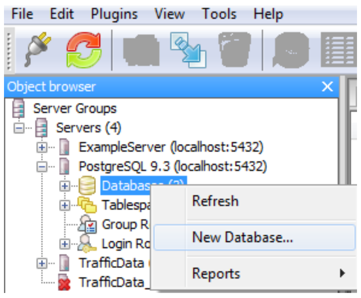
3. During the installation, use StackBuilder to include the PostGIS add-on package.

4. (Optional) PostGIS included in StackBuilder might not be the latest version. We recommend to get and install the latest PostGIS from `http://postgis.net/install`.

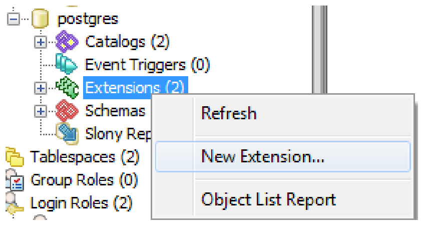
5. The installer for PostgreSQL includes also pgAdmin III (`http://www.pgadmin.org/`) - a graphical tool for managing and developing databases. Locate pgAdmin III and open it.

6. Create a new server or add an existing one by clicking File $\rightarrow$ Add Server. Fill the required information and press "OK". For more information and instructions see the pgAdmin III documentation (`www.pgadmin.org/docs/dev/index.html`) or PostgreSQL wiki page `https://wiki.postgresql.org/wiki/Main_Page`.

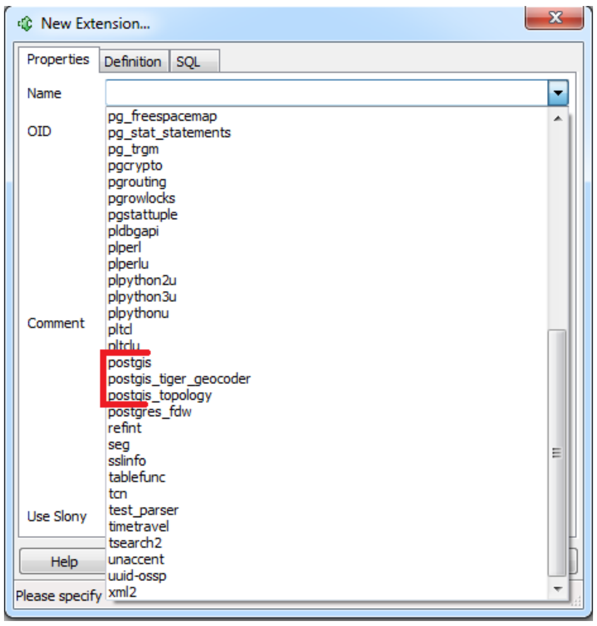
7. Create a new database in the server by right clicking on "Databases" in the server menu and select "New Database".



8. Add PostGIS extensions to the database. Right click on "Extensions" option of the database and select "New Extension".

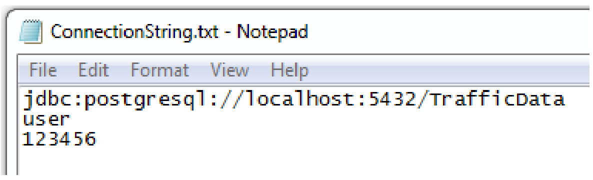


Add all postgis extensions. For more information visit also `http://www.bostongis.com/printerfriendly.aspx?content_name=postgis_tut01`.



9. Set up "Login Roles" on the server. A login role is used by the application to connect and work with the database.

10. Create a connection string text file. The file should have three lines. First jdbc:postgresql://*Host*:*port*/*DatabaseName*
Second and third line contain *Username* and *Password*.

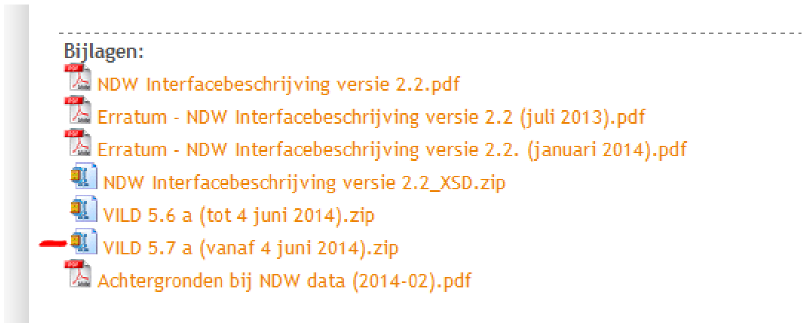


```
jdbc:postgresql://localhost:5432/TrafficData
user
123456
```

A.2 Download and import data from NDW to the database

There are several types of files provided by NDW. In our application we used only RVM-network data file and traffic intensity data files. The first one has to be imported manually into the database, the second type of files are imported with the help of our tool.

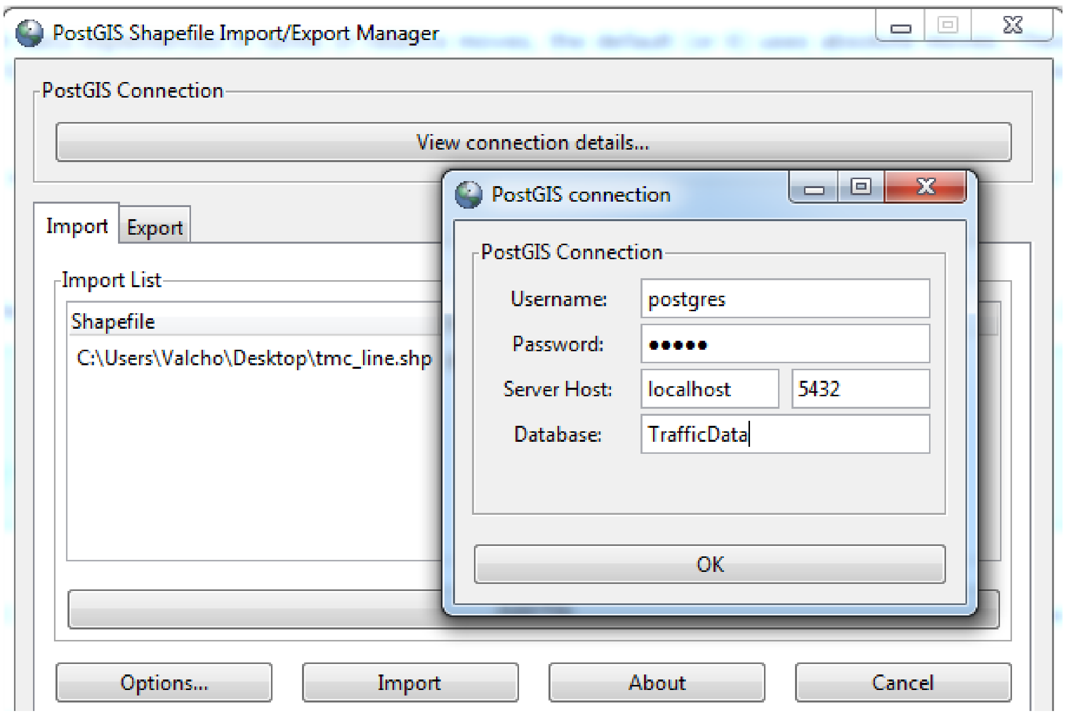
1. Go to NDW website (`http://www.ndw.nu/`) → Databank → Actuele verkeersgegevens (or click `http://www.ndw.nu/pagina/nl/4/databank/31/actuele_verkeersgegevens/`) and download the latest version of the VILD file (bottom of the page).



Bijlagen:
- NDW Interfacebeschrijving versie 2.2.pdf
- Erratum - NDW Interfacebeschrijving versie 2.2 (juli 2013).pdf
- Erratum - NDW Interfacebeschrijving versie 2.2. (januari 2014).pdf
- NDW Interfacebeschrijving versie 2.2_XSD.zip
- VILD 5.6 a (tot 4 juni 2014).zip
- VILD 5.7 a (vanaf 4 juni 2014).zip
- Achtergronden bij NDW data (2014-02).pdf

2. Locate the downloaded file and open it with the help of WinRAR, 7-Zip or other archiving software. Locate "tmc_line.dbf", "tmc_line.prj", "tmc_line.shp" and "tmc_line.shx" files and extract them from the archive.

3. Import the shape files into the database. This can be done through the SQL editor or use the graphical user interface. From the pgAdmin III menu select Plugins → PostGis Shapefile and DBS loader. Select "tmc_line.shp" file. Make sure that you have entered the correct database and credentials and press Import.

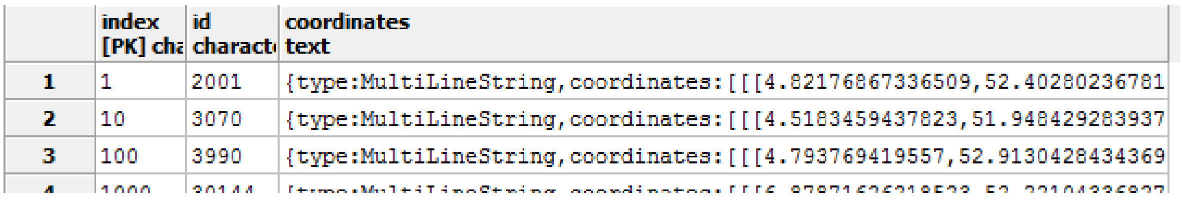


C:\Users\Valcho\Desktop\tmc_line.shp

PostGIS Connection

Username:	postgres	
Password:	●●●●●	
Server Host:	localhost	5432
Database:	TrafficData	

4. Locate *tmc_line* table in the database and rename it to *tmc_line2*. From the SQL editor run the following query:
   CREATE TABLE tmc_line AS
   SELECT gid,loc_nr,geom ,ST_AsGeoJson(geom) FROM tmc_line2
   Locate the newly created *tmc_line* table and rename its columns: *gid* to *index*, *loc_nr* to *id*, and *geom* to *coordinates*.

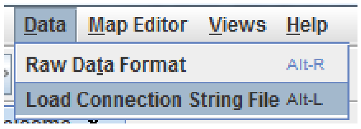


	index [PK] cha	id characte	coordinates text
1	1	2001	{type:MultiLineString,coordinates:[[[4.82176867336509,52.40280236781
2	10	3070	{type:MultiLineString,coordinates:[[[4.5183459437823,51.948429283937
3	100	3990	{type:MultiLineString,coordinates:[[[4.793769419557,52.9130428434369
4	1000	30144	{type:MultiLineString,coordinates:[[[6.87871626218523,52.22104336827

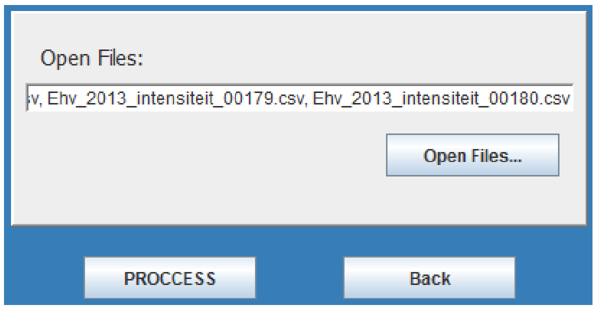
5. Go to http://ndw.dysi.nl and login using your credentials.

6. Use the interface to download raw data from NDW and extract files from the archive. Download only traffic intensity data (Intensiteit) for the required locations. We recommend to download data for *5 minuut* aggregation step (aggregatietijd). For more details about using the user interface, once logged in go to Handleiding → Historische Exporttool.

7. Open City Traffic Visualization and load a connection string file (Data → Load Connection String File).



8. Select Data → Raw Data Format option from the menu. In the new window select all extracted raw data files and press PROCESS. Depending on the size of the raw data this can take from several minutes to a few hours.

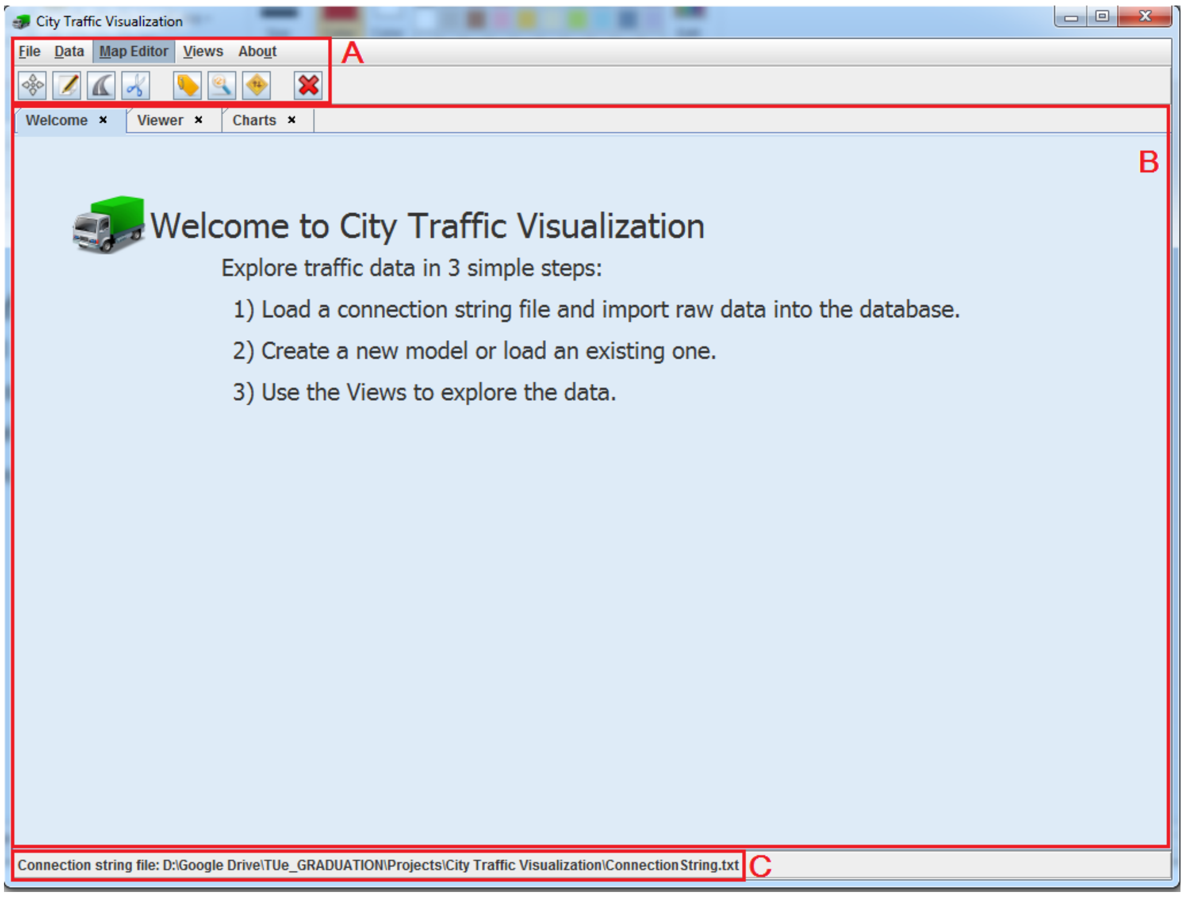

Open Files:
v, Ehv_2013_intensiteit_00179.csv, Ehv_2013_intensiteit_00180.csv
Open Files...
PROCCESS
Back

9. After the import is complete check your database. If the work was done successfully, the database should now have many tables - one named after each measuring location, one table named *pointsmetadata* and one table *tmc_line*.

Depending on their privileges, users can also use City Traffic Visualization to perform steps 6 to 8 and import raw data into the database.

Appendix B

User Manual



Welcome to City Traffic Visualization

Explore traffic data in 3 simple steps:

1) Load a connection string file and import raw data into the database.

2) Create a new model or load an existing one.

3) Use the Views to explore the data.

Connection string file: D:\Google Drive\TUe_GRADUATION\Projects\City Traffic Visualization\ConnectionString.txt

Figure B.1: A) Menu, B) Working panels, C) Connection string path

In the main application window frame three areas can be recognized. In top is the menu (A); in the middle are situated the working plot panels (B), where every opened view is put; on the bottom the path and the name of the currently loaded connection string file are shown (C).

B.1 Menu

B.1.1 File Menu

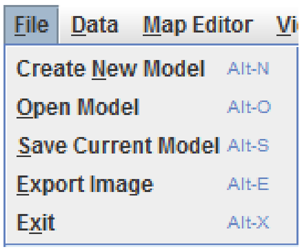


Figure B.2: File menu

1. Create New Model - Start building a new model;
2. Open Model - Open an already existing model;
3. Save Current Model - Saves the current model;
4. Export Image - Saves a picture of the current view to the computer;
5. Exit - Closes the application.

B.1.2 Data Menu

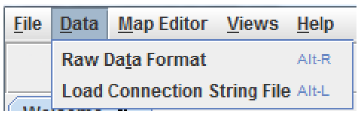


Figure B.3: Data menu

1. Raw Data Format - Process and upload raw data to the database;
2. Load Connection String File - Selects a connection string file for use.

B.1.3 Map Editor Menu

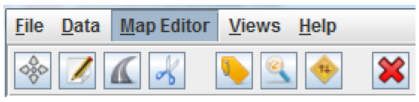


Figure B.4: Map Editor menu. Buttons from left to right: move marker, edit marker data, draw roads, edit roads, create a label, see label's markers, bind markers to a road, delete.

The options in this menu are used to create a new city model and edit an existing one if needed. Steps how to create a model can be found in section B.2.

1. Move Marker button: allows adjusting the display position of a measuring location marker;
2. Edit Marker Data button: allows editing the meta data for each measuring location;
3. Draw Roads button: allows custom drawing of roads on the map;
4. Edit Roads button: allows editing of a road by either cutting it or combining it with another;
5. Create A Label button: allows creation of a new label and allocating it to markers;
6. See Label's Markers button: shows the markers assigned with the selected label;
7. Bind Markers To A Road: binds a selection of measuring locations to a selected road;
8. Delete button: removes markers and roads from the model.

The **Move Marker** option can be used to adjust the display location of a measuring location marker (Figure B.5). Changing it is done by clicking and holding on the marker with the left mouse button and while holding it move the mouse around to drag the marker to the desired position. The selected marker is painted in blue color. The gray "ghost" point on the map shows the original marker position. The marker and its position are connected with a thin gray line. If the user wants to return the marker to its original position he can press the RESET button.

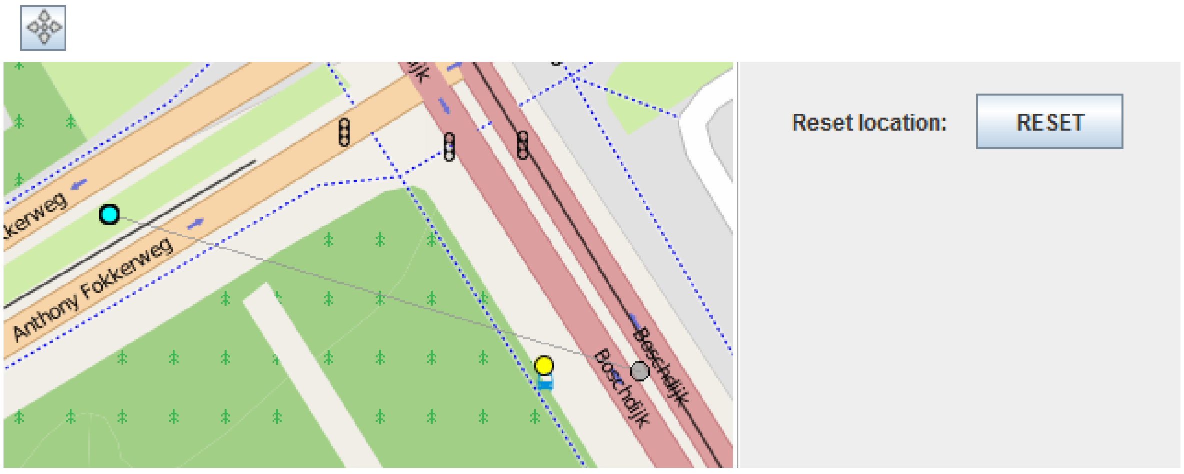


Figure B.5: Map marker with adjusted display location. The blue point represents the new display location, the connected gray point shows its original location.

Edit Marker Data allows the user to alter the original meta data for each marker (Figure B.6). First select the marker by clicking with the left mouse button on it. The marker gets colored in blue and its data is loaded to the text fields on the right. Make the required adjustments of the data and then save them by pressing the SAVE button. WARNING: By pressing SAVE, the data from the fields is saved directly to the database. Be careful with any changes as they are irreversible. Pressing the CANCEL button discards any changes that have not been saved and closes the menu.

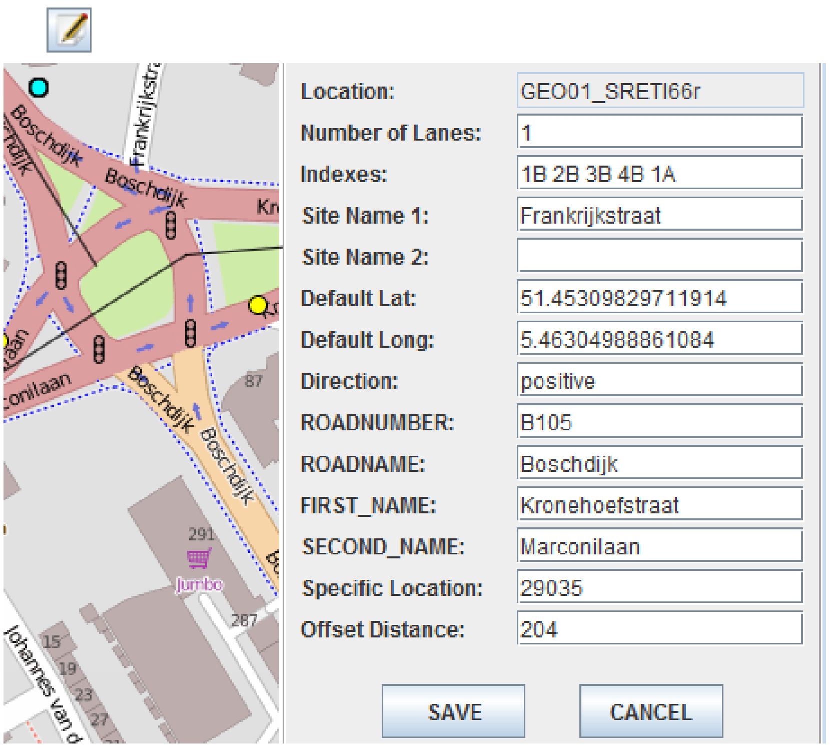

Location:	GEO01_SRETI66r
Number of Lanes:	1
Indexes:	1B 2B 3B 4B 1A
Site Name 1:	Frankrijkstraat
Site Name 2:	
Default Lat:	51.45309829711914
Default Long:	5.46304988861084
Direction:	positive
ROADNUMBER:	B105
ROADNAME:	Boschdijk
FIRST_NAME:	Kronehoefstraat
SECOND_NAME:	Marconilaan
Specific Location:	29035
Offset Distance:	204

Figure B.6: Meta data editor. Selected location's data (blue point) is shown in the text fields on the right where it can be altered.

The **Draw Roads** function allows the user to add a road to the road network. Clicking with the left mouse button on the map sets a start point for the road. Every left mouse button click after that adds the location of the mouse cursor to the points which the road is going through. The mouse cursor is always connected to the last set point of the road. To remove a point, click with the middle (scroll) mouse button anywhere on the map. This removes the last point of the road. To finish drawing the road, click anywhere with the right mouse button.

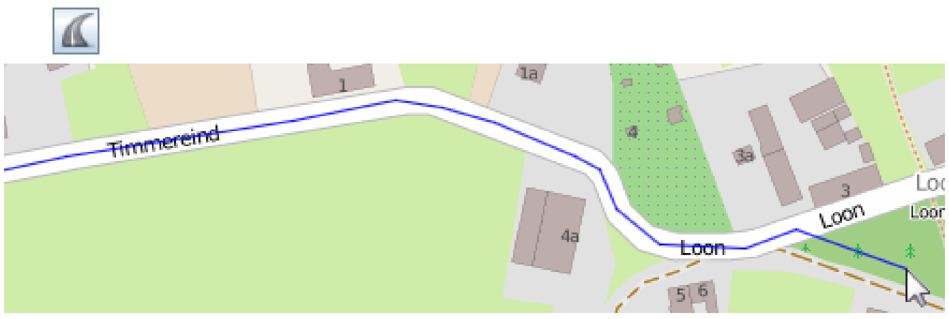


Figure B.7: Draw Roads function.

The **Edit Roads** function allows the user to cut and combine roads. To cut a road into two pieces, first select the Cut option from the menu on the right. Then click on a road to cut it at the location of the marker. To combine two paths together, select the Combine option. Then select the two roads that you want to combine. Selected roads are drawn in red. Click Combine button (Figure B.8).

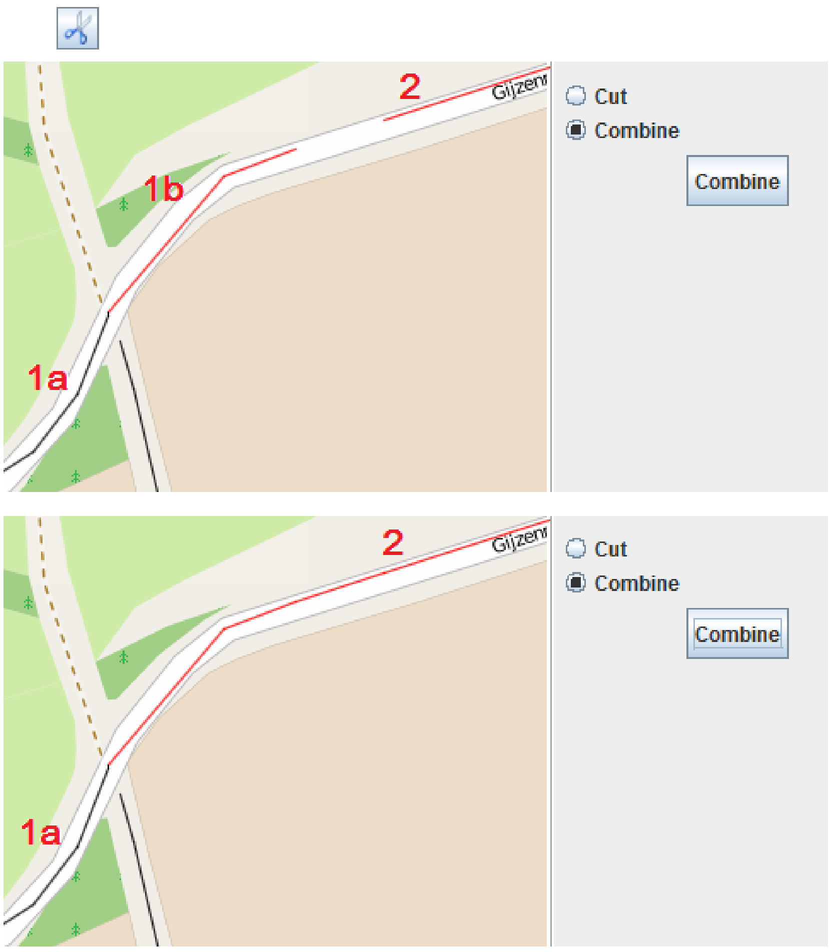


Figure B.8: Edit Roads function. In the top image road 1 has been cut into two parts - road 1a and 1b. In the bottom image road 1b has been combined with road 2.

The **Create A Label** functions allows binding of several markers into a group by assigning them the same label. To create a new label select the *NEW LABEL* option from the combo box. Then write the name of the new label in the ”Name” field. Click on the small square next to ”Color” to select and assign a color for this label. Select locations that will be tagged with the new label. Selecting locations can be done by either clicking on them or holding the left mouse button and draw a selection rectangle around them. The selected locations are painted in blue. Press Put Label to assign the label to the locations (Figure B.9).

If you want to assign more locations with an existing label, first select the locations. After that select the label from the combo box and press Put Label.

To see the locations with a given label simply select the label from the combo box and press Show Markers. Locations with that label are painted with the color assigned to the label.

To remove a label from one or several locations, first select the locations. Then select the label from the combo box and press Remove Label.

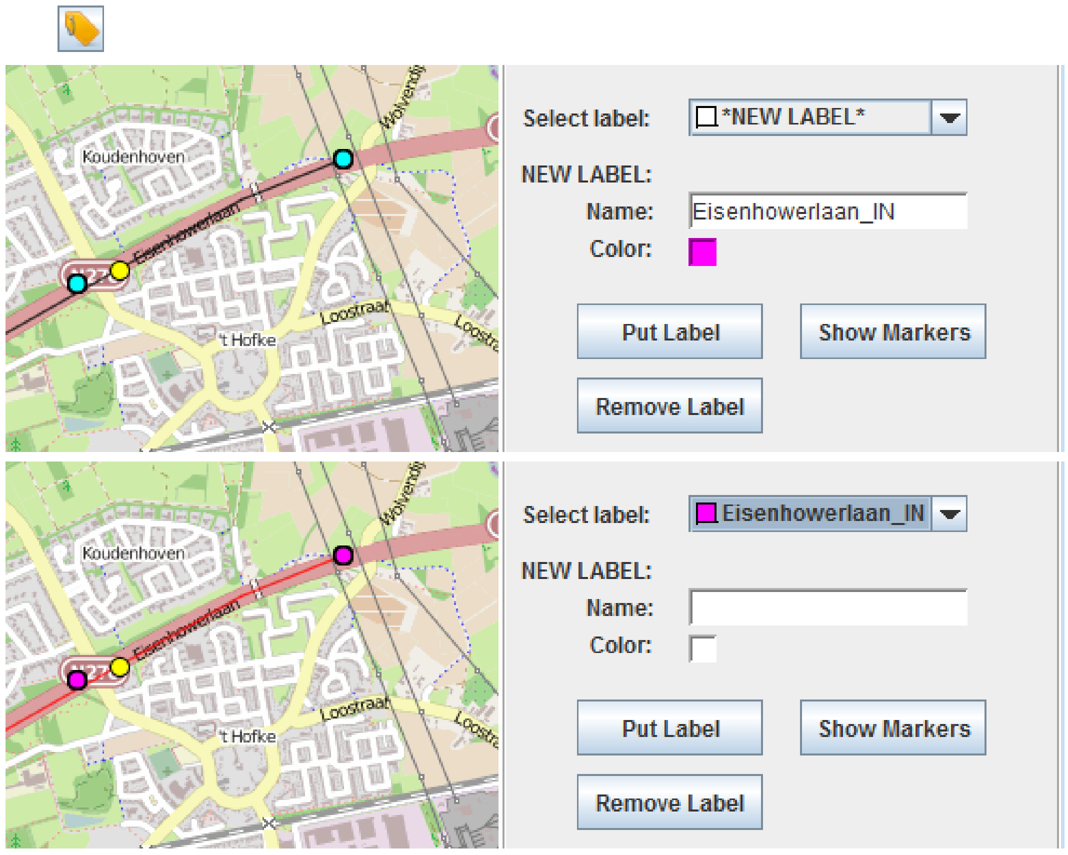


Figure B.9: Creating a label and assigning it to markers. Top image - label *Eisenhowerlaan_IN* is being created and is about to be assigned to the selected markers (blue). Bottom image - all markers with label *Eisenhowerlaan_IN* are shown, painted with the color of the label.

See Label's Markers is similar to the Show Markers option from the Create A Label menu. Simply select a label and press Show Markers to see the markers with that label.

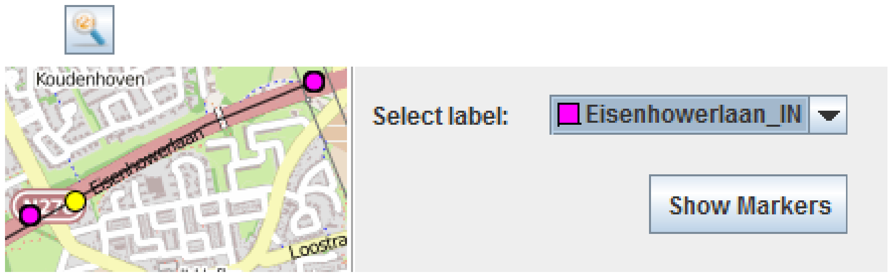
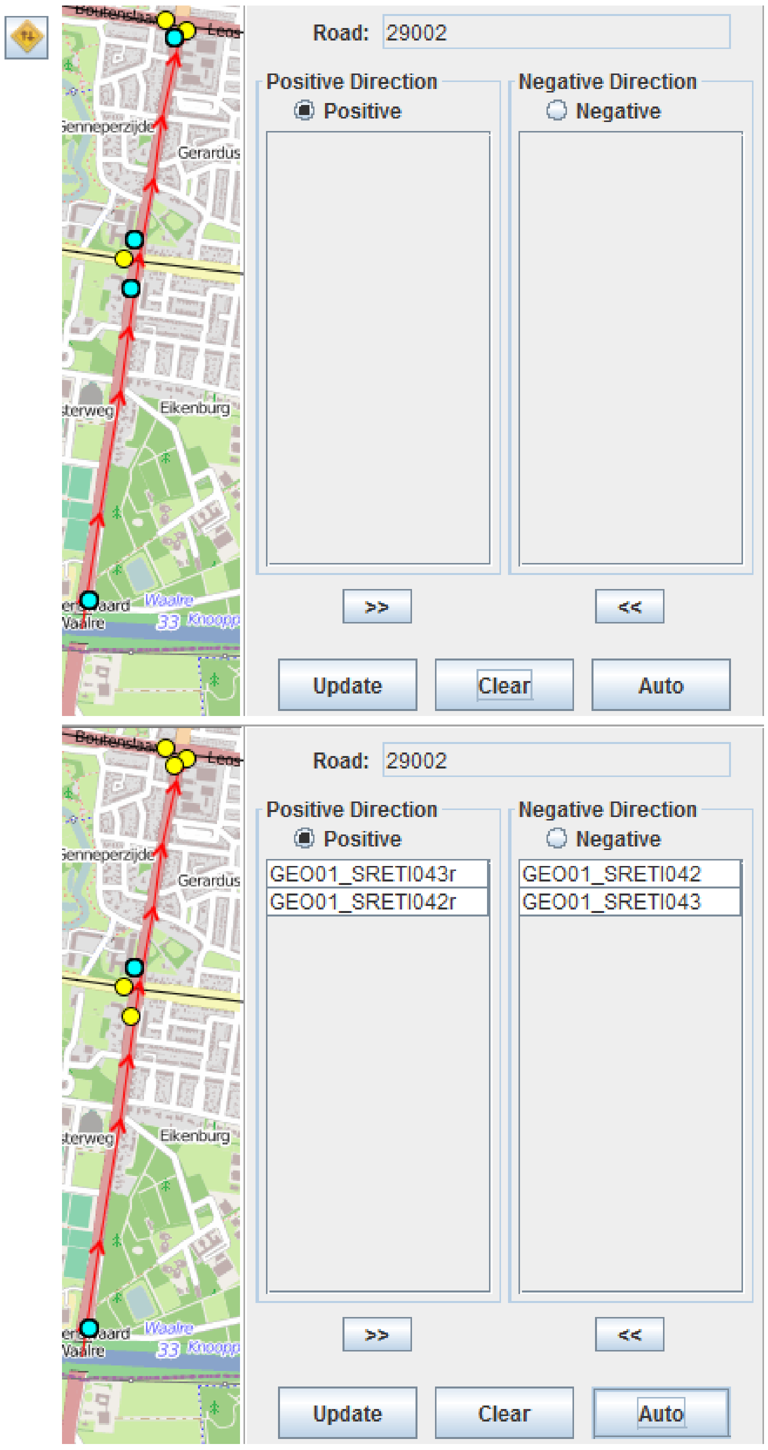


Figure B.10: See Label's Markers option. Markers assigned with *Eisenhowerlaan_IN* label are painted in purple.

Bind Markers To A Road is one of the most important functions in the Map Editor menu. It allows the user to bind measuring locations to a road.



Road: 29002

Positive Direction	Negative Direction
GEO01_SRETI043r	GEO01_SRETI042
GEO01_SRETI042r	GEO01_SRETI043

Figure B.11: Bind Markers To A Road function. Markers are assigned to road 29002.

Binding a measuring location to a road can be done in two ways. The first is semi automatic and the second one is fully manual.

For the semi automatic way, first select the road you want to assign markers(locations) to. The selected road is painted in red and its name is shown in the "Road" text box on the right. Second, select all markers that are to be binded to this road. Press the Auto button to make the application automatically determine the direction of the traffic at the marker and bind the marker for the right direction of the road. Markers assigned to the positive direction of the road are shown in the "Positive Direction" table and the ones in negative direction to "Negative Direction". Positive direction is the direction in which the road is drawn. In almost all of the cases this direction matches the direction in "Direction" field of the meta data for the marker (See figure B.6). Pressing the Auto button just makes the application compare the directions. After assigning has been finished carefully inspect the result. Make sure that the locations are binded to the right direction. Pressing "Positive" and "Negative" radio buttons makes small red arrows appear on the road indicating respectively positive or negative direction and markers from that table are painted in blue. If a marker needs to be transfered from one direction table to the other simply select this marker and press ">>" or "<<" buttons.

Assigning locations manually is done in a similar way. First, select a road. Second, select a direction for which you want to bind markers by pressing either "Positive" or "Negative" radio buttons. Select the markers for the chosen direction. Press the Update button.

If you want to remove all markers assigned to some direction of a road, select the road and the direction and press the Clear button. This removes all markers from the corresponding table.

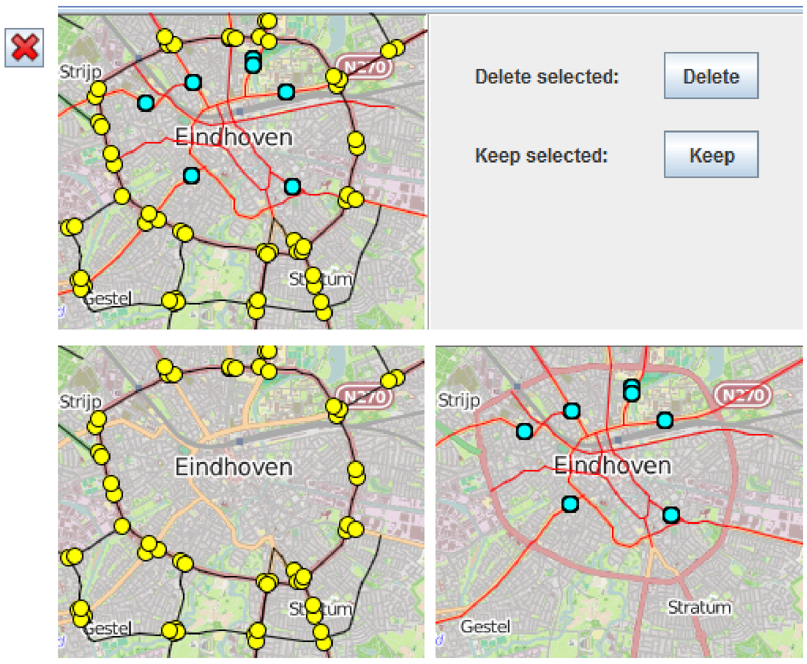


Figure B.12: Deleting unnecessary objects. Top image: selected objects are painted in red for the roads and blue for markers. Bottom left image: the selected objects are deleted. Bottom right image: only selected objects are kept.

If you want to see the locations bound to a road click on the road with the left mouse button. The tables and the fields on the right are automatically filled with the road data.

The **Delete** option allows you to remove objects from the map. First, select objects by clicking on them or drawing a selection rectangle around them. Selected roads are painted in red and selected markers in blue. Pressing the Delete button removes the selected objects from the model. Pressing the Keep button deletes every object but the selected one (Figure B.13).

B.1.4 Views Menu

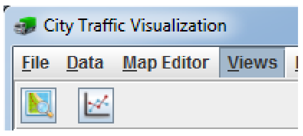


Figure B.13: Views menu. Buttons from left to right - map view, charts view.

1. Map View button: allows visualization of geographic traffic data on the map;
2. Charts View button: allows visualization of temporal traffic data with charts.

B.1.5 About Menu

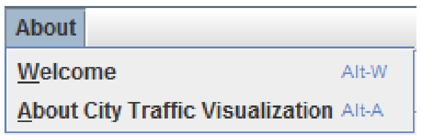


Figure B.14: About menu

1. Welcome - opens the welcome screen;
2. About City Traffic Visualization - Opens a window with information about City Traffic Visualization.

B.2 Creating a model

A city model is the base on which our visualizations are build. It is required for the proper functioning of *City Traffic Visualization*. The creation of a model is done with the help of the Map Editor menu options and is recommended to be done in the following order:

1. Load a connection string file if not done so.
2. From File menu select Create New Model. If the user wants to continue working on an existing model then select Open Model and load the model file.
3. With the help of Delete option remove all roads and markers that are not going to be part of the model. This reduces the clutter of the generated image.
4. If needed use Draw Roads and Edit Roads options to shape the road network in the model. More accurate road network makes more accurate road traffic visualization.
5. Select Move Marker option to adjust the display position of the markers. The display position is important for the accurate visualizations on the map. Try to position the markers close to the roads or close to the location of the point at which they measure traffic.
6. Use Bind Markers To A Road to allocate measuring locations to a road. Make sure that you have allocated each location to the right direction of the road. Use the metadata for each point to determine its correct location. The traffic on each road is determined by the traffic from the measuring locations linked to it. This step should be done carefully and with utmost attention.
7. If needed use Edit Marker Data to correct the metadata associated with the measuring location.
8. Use Create A Label to assign labels to the measuring points. This allows you to visualize traffic from a group of points.
9. Save your model for future use by selecting Save Model from File menu.

B.3 Create a view

Views allow you to get a visualization of the selected information. Both types of visualization have a similar user interface. Before starting, make sure that you have loaded a connection string file. Secondly, create a new city model or load an existing one.

Map visualizations present traffic data over geographical road map.

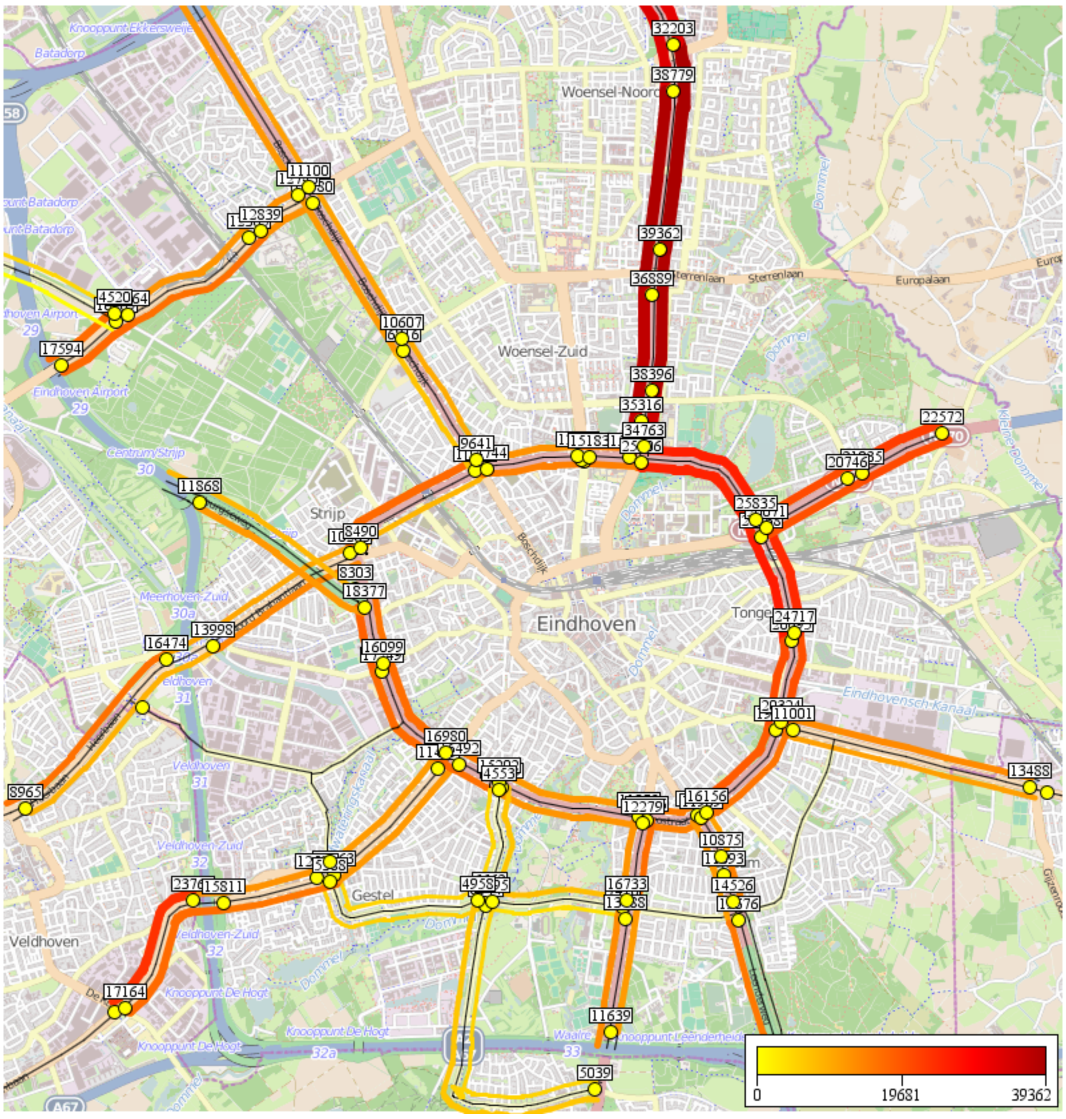


Figure B.15: Traffis data visualization over a road network

Figure B.17 shows the user interface with all available controllers that are used to create and tune these images.

From combo box 3 the type of visualization can be selected: visualizing data over the road network (figure B.15) and visualizing data over measuring locations (figure B.16).

Combo box 1 allows the selection of a color scheme (bottom right corner of the image). User can select between Partial Hot Body (*PartialHeat*), Yellow-Orange-Red (*YlOrRd*), Red shades (*Reds*), Yellow-Green-Blue (*YlGnBl*), Red-Purple (*RdPu*), Purple-Red (*PuRd*), Cyan-Magenta (*CyMa*), and Blue-Green-Red (*BlGnRd*). Selecting check box 2 makes each color scheme discrete.

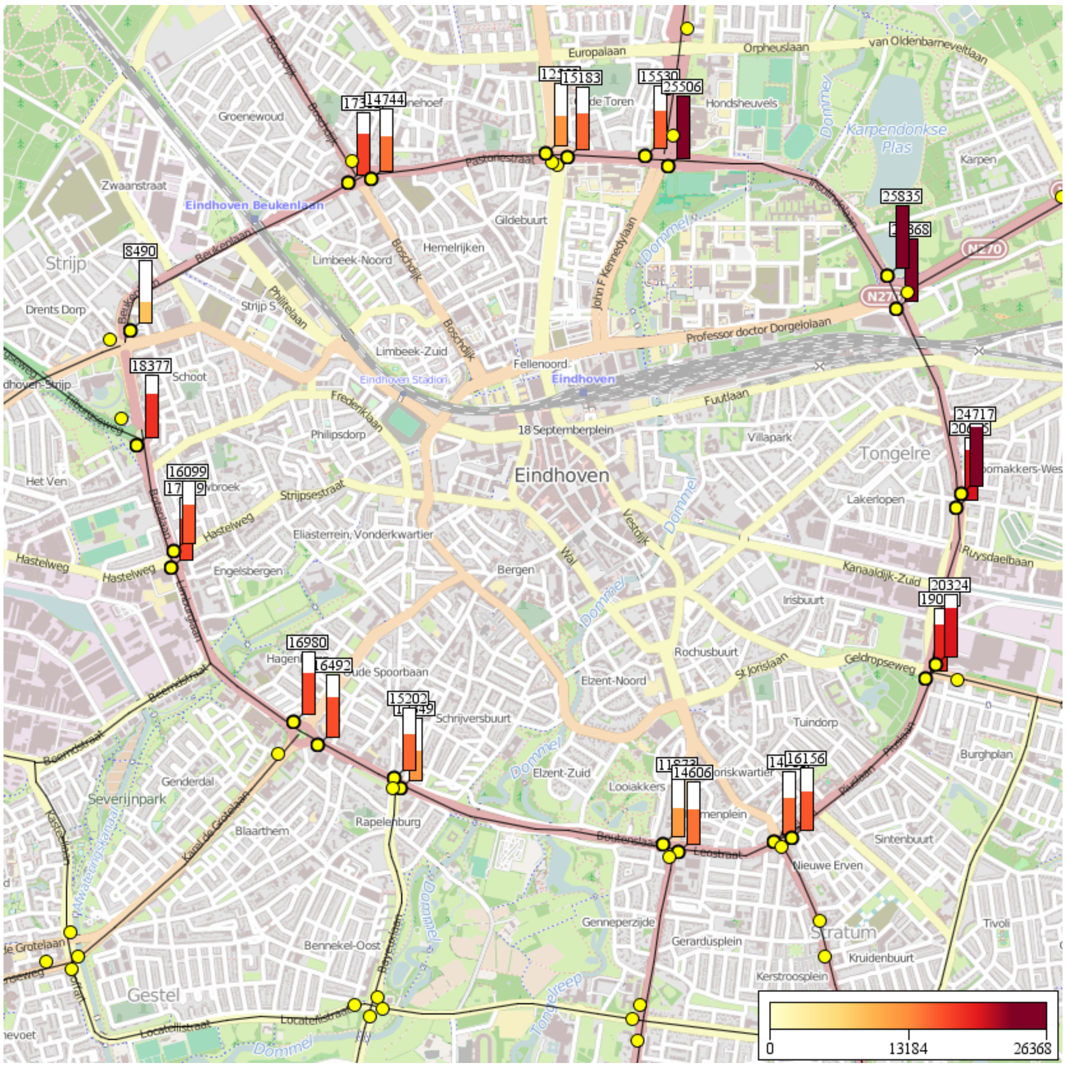


Figure B.16: Traffic data visualization over measuring locations

Group 4 of check boxes allows the selection of vehicle class. Vehicles are divided into classes by their length: $<= 5.6m$ (4b), $> 5.6m < 12.2m$ (4c), $> 12.2m$ (4d) and $Uncategorized$ (4e). By default you can select several boxes at the same time. Pressing button 5 allows the selection of only one class. Pressing button 6 selects all vehicle classes (4b to 4e). Button 7 deselects all selected classes and button 8 inverts the selection - selected classes get deselected and vice versa.

Table 9 keeps a list of all measuring locations in the model. During measuring locations visualization, the user can select for which locations he wants to see data. Using combo box 10 the user can select locations having the same label.

Controllers 11 to 13 allow selection of extra objects on the map. Box 11 shows the yellow points on the image (figure B.15), box 12 enables the small rectangles with values above the markers. Box 13 shows arrows over each road, pointing at the direction of the traffic. Slider 14 adjust the transparency of the background map.

With radio buttons 15a and 15b the measuring unit of the visualized data can be selected: *Total number of vehicles* or *Vehicles per hour*.

Radio buttons 16 allow you to select the maximum value used for the color scheme. 16a makes automatic selection and 16b allows you to put it manually. The value in text box 16b1 is used.

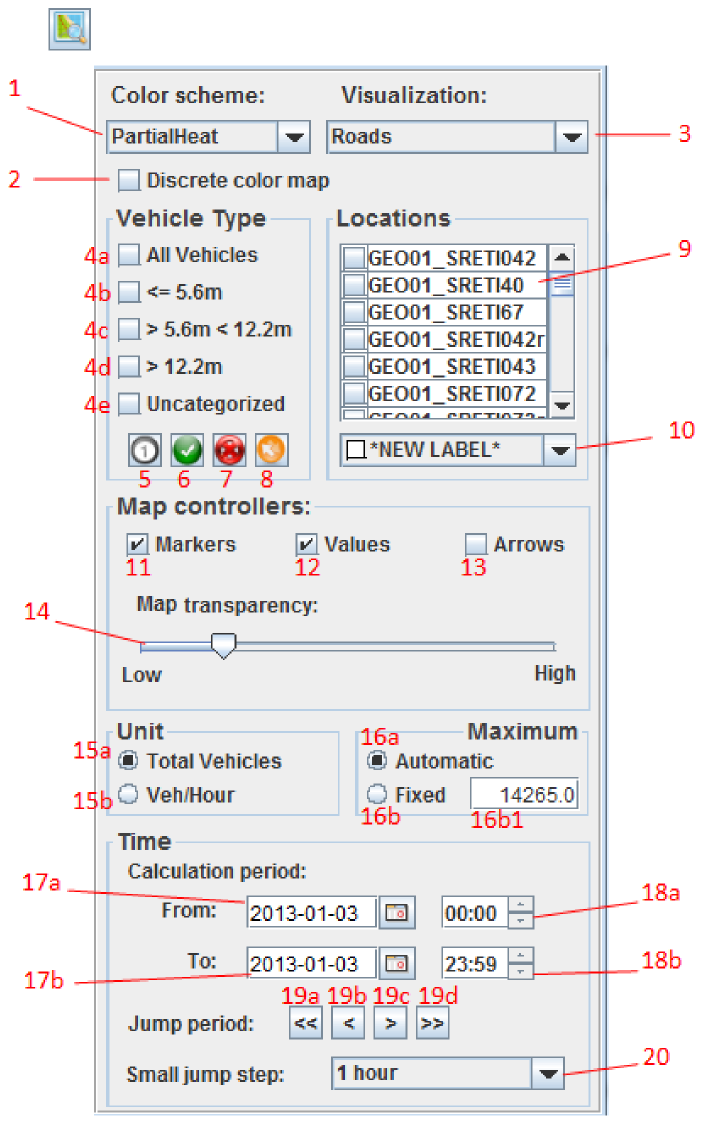


Figure B.17: Map visualizations user interface

Time controllers 17-20 allow the selection of a time period for which data is visualized. Boxes 17a and 18a show the start date and hour and 17b and 18b respectively the end date and hour. The small calendar button next to the date boxes gives the option to make a fast date selection using a calendar menu. Button group 19 make quick jumps forward and backward in time. 19a gives a jump back in time with a step equal to the selected time interval length [FromTime, ToTime]. 19d makes the same step but forward. 19b and 19c make steps backward and forward in time but with the step defined in combo box 20.

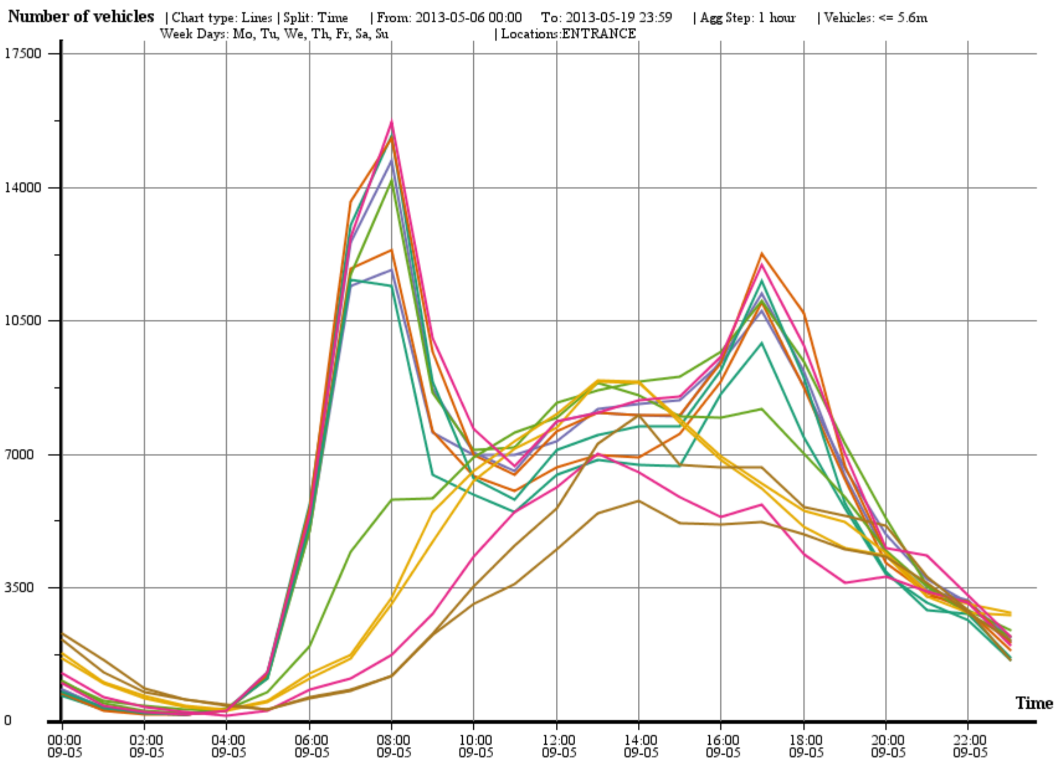


Figure B.18: Traffic data visualization with the use of line chart.

Charts visualization allows visualizing data with line or bar charts. The user interface (figure B.20) is similar to the one used for map visualizations. Most of the controllers are the same and have the same function. Here only the new controllers are explained.

Check box 1 allows the selection of the type of chart: lines (figure B.18) or bars (figure B.19).

With check box 3 user can select if he wants to visualize data based on label or individual locations.

Radio buttons 2a, 2b, 2c and 2d allow the selection of the attribute on which data should be split. 2a splits the data based on vehicle type and location, 2b on vehicle type only, 2c on location only and 2d on day of the week.

Button group 4 has the same function as in the map visualizations interface. In charts, the colors assigned to each class can determine the color of the drawn lines or bars. To change the color of the class, press the red square and select a color from the color swatch.

Button group 5 gives the option to select for which days of the week to see data (figure B.18). By default it is possible to select several days. The four buttons in the Week Days panel have the same function as the ones in Vehicle type panel - select one, select all, deselect all and invert selection.

The controllers in panel y-Scale (6) allow you to set the maximum value which is visualized on the chart.

Based on the type of chart and grouping selection, the Locations panel contains different controllers (versions A, B and C). Version A is available when we have lines chart and labels. Table 7 shows all predefined labels and from there the user can select for which labels he wants to visualize data. Version B is available when we have selected *Points* from combo box 3. This panel and controllers are the same as the one in map visualizations user interface. Version C shows up when we have bar charts and show data for labels. Combo boxes 8 and 10 allows the selection of labels for the visualized data. If check box 9 is selected, the user can compare the data from the two selected labels (figure B.19). If check box 11 is selected then an orange compare line is drawn, showing the difference between the height of the corresponding bars.

Positioning the mouse cursor over a line or a bar highlights it and a small box is shows, containing the visualized value at that point.

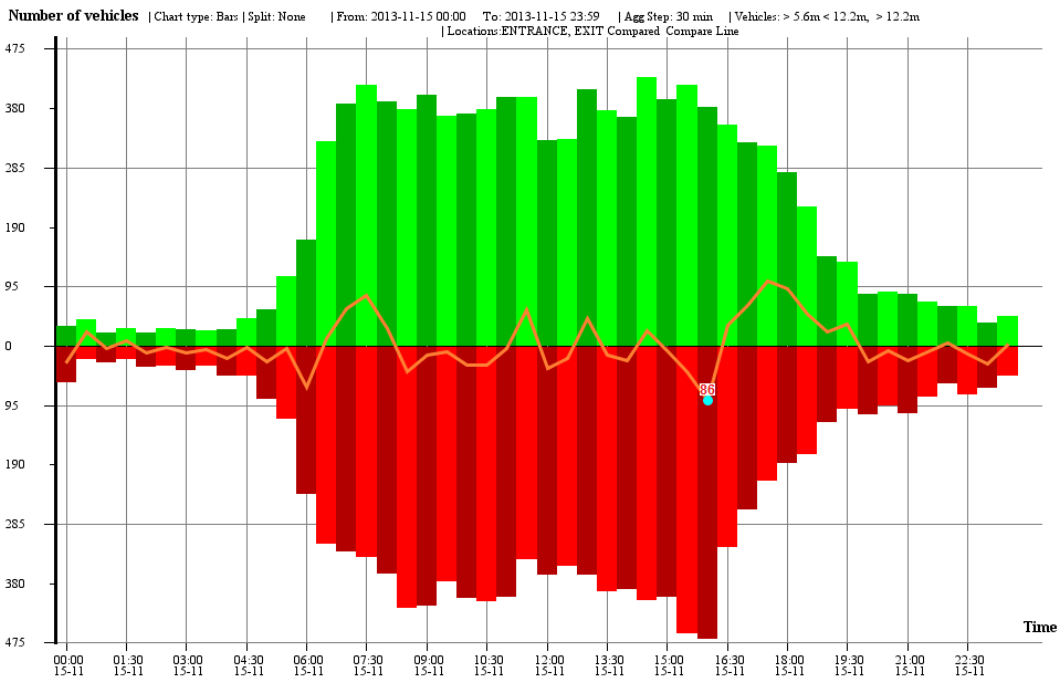


Figure B.19: Traffic data visualization with the help of bar chart.

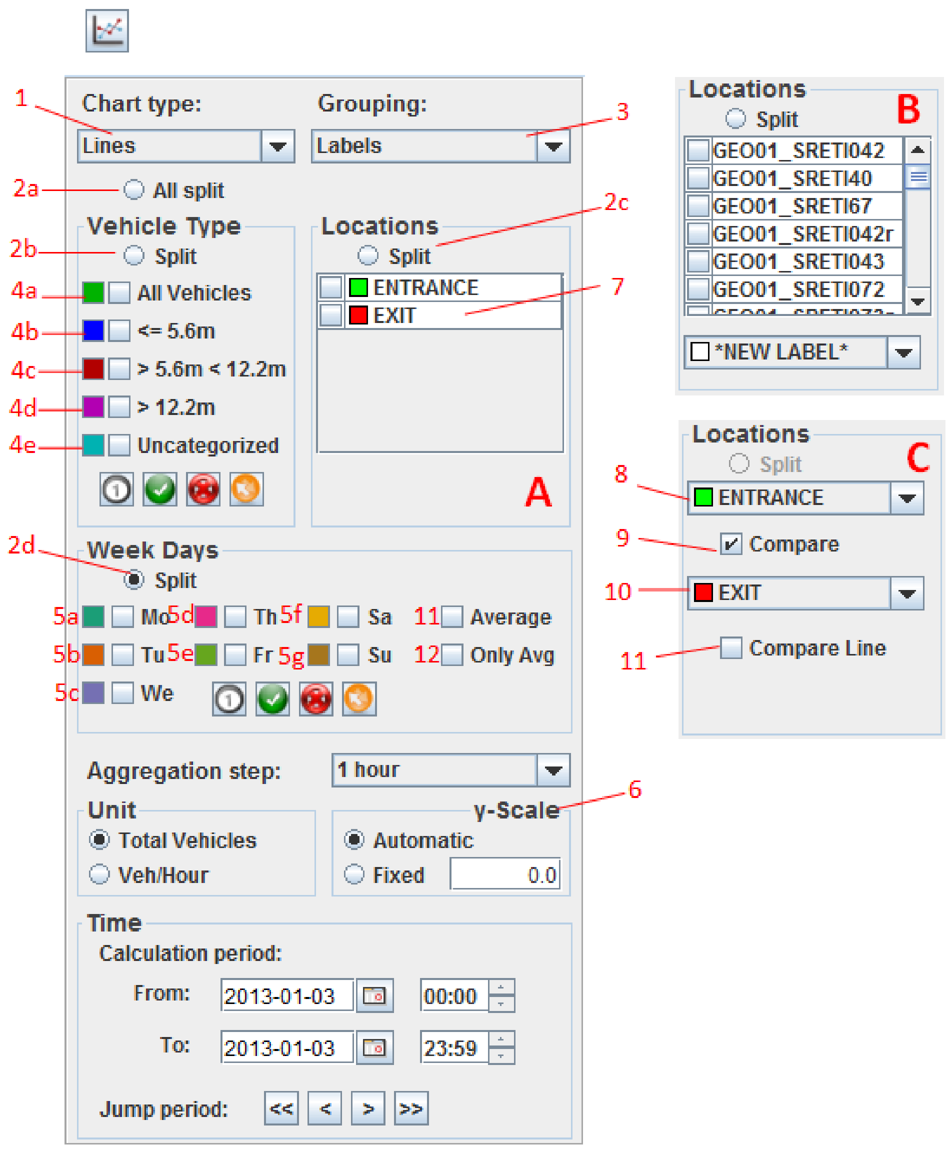
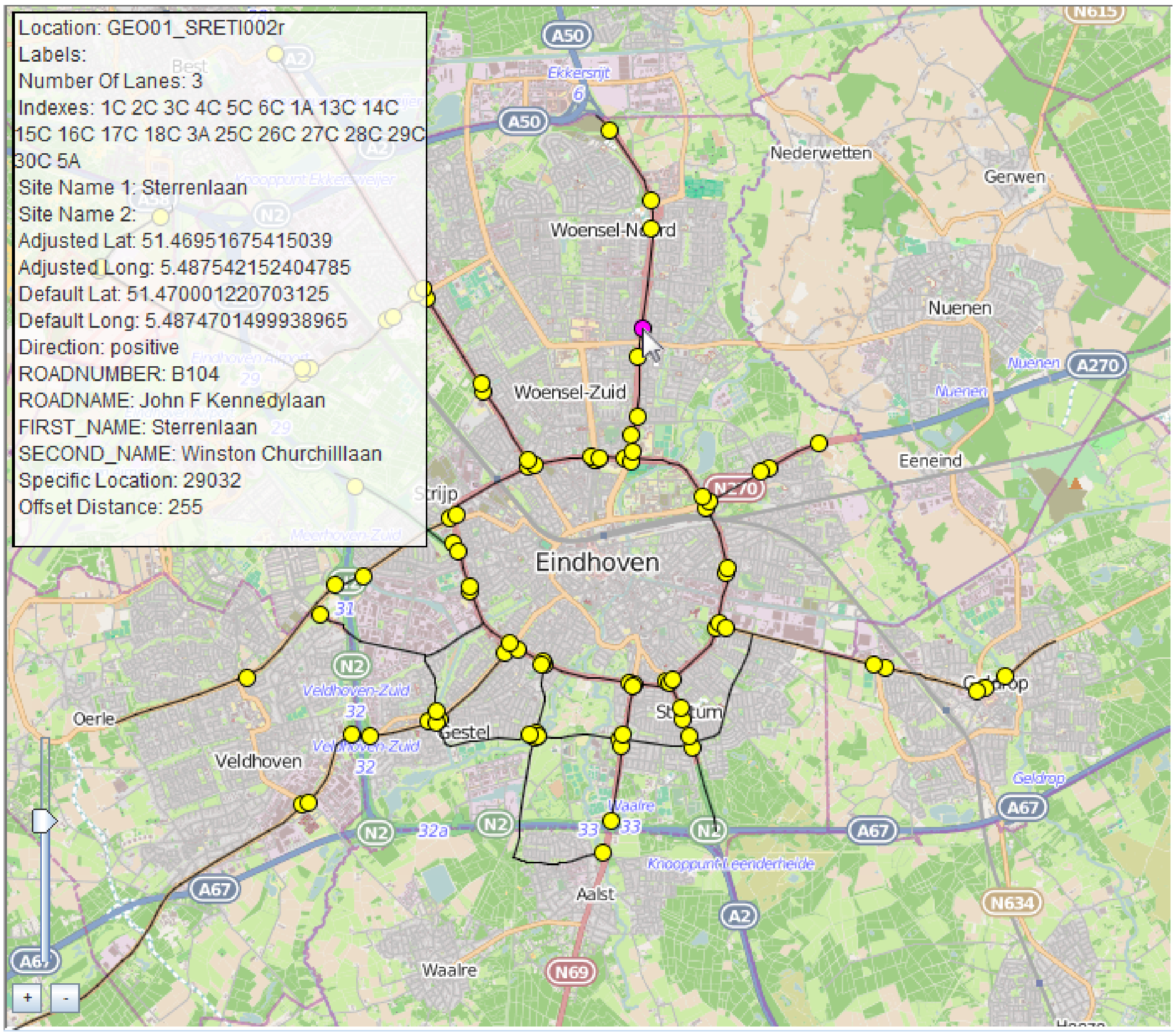


Figure B.20: Chart visualizations user interface

B.4 Map Interaction



Location: GEO01_SRETI002r
Labels:
Number Of Lanes: 3
Indexes: 1C 2C 3C 4C 5C 6C 1A 13C 14C 15C 16C 17C 18C 3A 25C 26C 27C 28C 29C 30C 5A
Site Name 1: Sterrenlaan
Site Name 2:
Adjusted Lat: 51.46951675415039
Adjusted Long: 5.487542152404785
Default Lat: 51.470001220703125
Default Long: 5.4874701499938965
Direction: positive
ROADNUMBER: B104
ROADNAME: John F Kennedylaan
FIRST_NAME: Sterrenlaan
SECOND_NAME: Winston Churchilllaan
Specific Location: 29032
Offset Distance: 255

Controlling the map is fairly easy. Scrolling the map around can be done by clicking and holding the right mouse button anywhere on the map and moving the mouse while still holding the button. Releasing the button stops the scrolling.

Adjusting the zoom level can be done in two ways. First, Zoom level controllers can be used. They are located at the bottom left corner of the map. Dragging the slider up or pressing the "+" button increase the zoom level (map comes closer) and dragging it down or pressing the "-" button decreases the zoom level (map gets further away). Optionally, the user can use mouse wheel to adjust the zooming of the map, or double click with the left mouse button to zoom in on that location.

Positioning the mouse cursor over any measuring location marker pops up an information box, presenting most of the meta data for this location. Removing the cursor from the marker causes the box to disappear.