

Map of Denmark

*First-Year Project, Bachelor in Software Development,
IT Univ. of Copenhagen*

Group 12

Jakob Melnyk jmel@itu.dk

Niklas Hansen nikl@itu.dk

Emil Juul Jacobsen ejuu@itu.dk

Jens Dahl Møllerhøj jdmo@itu.dk

Supervisors: Lars Birkedal

Advisors: Jonas Brabrand Jensen and Filip Sieczkowski

May 25th, 2011

Contents

1	Preface	4
2	Background	5
2.1	Problem area	5
2.2	Requirements for the map	6
2.3	Our requirements	6
2.3.1	Project requirements	6
2.3.2	Our own requirements	6
2.4	Data set	6
2.4.1	UTM-coordinates	6
2.4.2	Graph	6
2.5	MVC structure	6
3	User Interface analysis	7
3.1	User interface as a whole	7
3.2	Interesting features	8
3.2.1	Zoom	8
3.2.2	Navigation	8
3.2.3	Hotkeys	9
3.2.4	Route planning and markers	9
3.2.5	Bike/car	9

3.3	Features not implemented	9
3.3.1	Choice of roads to be displayed	10
3.3.2	Smooth scrolling	10
3.3.3	Dynamic route finding	10
4	Implementation	11
4.1	Dijkstra vs A-star	11
4.2	Evaluator	11
4.3	Quadtrees	11
4.4	Serialization	11
4.5	UTM-conversion	11
4.6	Mousezoom	11
4.7	Floats	11
5	UML-diagrams	12
5.1	MVC	12
5.2	Simple Diagram	12
5.3	Control flow	12
6	Tests	13
6.1	WhiteBox: closestEdge	13
6.2	JUnit	13
6.3	System test	13
7	Manual	14
7.1	Navigation	14
7.1.1	GUI	14
7.1.2	Keyboard	14
7.2	Zoom	14
7.2.1	GUI	14

7.2.2 Keyboard	14
7.3 Route find	14
7.4 Bike/car	14
7.5 Resize	14
7.6 Road display	14
8 Product conclusion	15
9 Group norms	16
10 Diary	17
11 Worksheets	18
12 Process description and reflection	19

Chapter 1

Preface

Chapter 2

Background

2.1 Problem area

Over the last decade people have switched from traditional roadmaps to using the web-maps. This is a change without any negative side-effects. The online services remove all the problems with determining the quickest route between two points and you spend no time browsing the pages of the map to find what you need. With the popular smartphones the online map is even more useful, since you no longer need to prepare your trip before you leave.

The online maps have now been used for many years and haven't been slow at adopting new features to improve their usability. They have both implemented satellite-maps that allow us to browse the entire planet from above, and lately the feature called Google Street View has upped the stakes when allowing us to look at any direction from a given point of a road. The two maps that we use the most are Google Maps and the Danish map called Krak. These maps both have the mentioned features but slight differences in the way the user navigates and searches for routes.

Because of the widespread knowledge of the online maps, the users have been accustomed to certain features and ways of using the map. It is very important that we, with a new map program, use this knowledge to our advantage and don't try to reinvent the wheel. By using some of the commonly used controls in our map, a user will be able to quickly adapt to our program and use it efficiently.

2.2 Requirements for the map

2.3 Our requirements

2.3.1 Project requirements

2.3.2 Our own requirements

2.4 Data set

We have been provided with a dataset of roads and intersections in Denmark from Krak. Additionally we got some code for loading the data in from the text files. We have only made minor changes to the code for loading the data.

2.4.1 UTM-coordinates

It is important to note that the KrakNodes are in UTM-32 coordinates. When using the UTM standard the origo is placed at the south-west corner. These coordinates need some conversion when using in Java since the origo is placed differently.

2.4.2 Graph

When the data has been loaded it is stored as a Graph containing KrakNodes and KrakEdges. The KrakEdges are the road segments and contains the name of the road, an estimated drive time, a direction of traffic and references to the two KrakNodes that are at either end of the road. The KrakNode itself contains only the coordinates for the point. The Graph itself contains a number of useful methods for searching the data like getting all edges that is connected to a KrakNode. We will be using these methods extensively throughout the project both for drawing the map and for finding the route between two points.

2.5 MVC structure

Chapter 3

User Interface analysis

In this chapter we describe our decisions and present our analysis and arguments regarding some of the features that we find interesting.

3.1 User interface as a whole

When we designed the first version of the graphical user interface in the first part of the project, we decided to make a window inside of the graphical user interface where the actual map should be displayed. We chose to have this window placed on the right side of our graphical user interface and interaction with the user mainly placed on the left.

We believe that this is a simple way of representing a user interface for a map. A lot of software use a menu bar with dropdown menus for selecting different functions. When we designed our outline for the graphical user interface, we did not design it with a huge amount of functions in mind.

The features that we have implemented in this version can easily fit in our simple user interface, but if features like searching for roads, route planning, etc. are included, then space and overview may become an issue on the left side.

Depending on the feature, we feel it would be beneficial to let the main window change when different feature types are selected.

Below is a screenshot of our standard user interface. How to use it will be explained in the **Manual** on page 14.

3.2 Interesting features

This section presents some of the interesting features we have implemented.

3.2.1 Zoom

We have a few options for zooming in and out on the map. As described in section 2.3.1 **Project requirements** on page 6, it was required that we made it possible to zoom by dragging a box around the part of the map the user wants to view.

In addition to the option of using the mouse to zoom, we have implemented a zoom-in and out function on the GUI and a hotkey for zooming out to the original view. We made the original view function a hotkey only because we did not want to have too many buttons on the left side. We considered making it a menu bar function, but we did not manage to get it into this version.

We felt we really needed a zoom out function, so users do not need to close the program and start it again, when the user wants to view the map further zoomed out. A combination of the zoom in and out functions helps the user a lot when navigating the map.

We have limited how far a user can zoom in and out. If the user tries to zoom out further than the original zoom level, the view will default to the original zoom level. If the user attempts to zoom in further than a width or height of 200 in UTM32-coordinates, the zoom function will do nothing. This limitation could be improved by zooming in on the smallest possible zoom at the position the user selected, instead of doing nothing.

3.2.2 Navigation

We have made it possible for the user to navigate the map by using the arrow buttons on the graphical user interface. When one of the buttons are pressed, the “view” will move in the direction specified by the button. While it was not specified as a requirement for the project, we felt it was a necessity to implement at least basic navigation functionality.

Like we did with the two zoom functionalities, we have limited how far a user can move around the map. The user is free to move around the map, but if user moves outside the bounds of the map in a way where the view would show an image that is not part of the map, the move function will not do anything.

3.2.3 Hotkeys

We have implemented hotkeys for all the buttons on the graphical user interface plus an additional for zooming back to the original view. When we discussed the benefits of hotkeys, we felt it was important for experienced users of the software should have a less cumbersome time navigating the map.

At first we just had hotkeys for the clearing of markers (mentioned in section 3.2.4) and zooming out to the original view, but we later added the hotkeys for the rest of the functionalities. If more features are added in a future version, it would be important for us that a hotkey were provided, if at all possible.

3.2.4 Route planning and markers

Part of the requirements for the project was to provide the user with a way to get the fastest or shortest route from one point to another. We accomplish this by putting a “marker” at the spot where the user clicks with the mouse. The marker shows which number it is. This will change if a marker is removed. Originally we had “pins” instead of markers, but we changed it, as we felt the pins we had were a bit large.

We have made it possible to place more than the two markers that the project requirements asks for. If the user places more than two markers, the software will find the shortest route between 1- i 2 and 2- i 3 and so on. This was cheap for us to implement, and felt it added a good feature to our software.

We have implemented two methods of removing pins from the map. We have assigned a hotkey to the graphical user interface button “Clear Markers”, which clears all the markers on the map.

3.2.5 Bike/car

3.3 Features not implemented

This section presents some of the features we have chosen not to implement. These features are not in the final program, because we did not feel there were compelling arguments for implementing them.

Features that we wanted to implement, but did not make it into the final version will be discussed in chapter Product conclusion on page 15.

3.3. FEATURES NOT IMPLEMENTED 3. USER INTERFACE ANALYSIS

3.3.1 Choice of roads to be displayed

3.3.2 Smooth scrolling

3.3.3 Dynamic route finding

Chapter 4

Implementation

4.1 Dijkstra vs A-star

4.2 Evaluator

4.3 Quadtree

4.4 Serialization

4.5 UTM-conversion

4.6 Mousezoom

4.7 Floats

Chapter 5

UML-diagrams

5.1 MVC

5.2 Simple Diagram

5.3 Control flow

Chapter 6

Tests

6.1 WhiteBox: closestEdge

6.2 JUnit

6.3 System test

Chapter 7

Manual

7.1 Navigation

7.1.1 GUI

7.1.2 Keyboard

7.2 Zoom

7.2.1 GUI

7.2.2 Keyboard

7.3 Route find

7.4 Bike/car

7.5 Resize

7.6 Road display

Chapter 8

Product conclusion

Chapter 9

Group norms

Chapter 10

Diary

3-4

Chapter 11

Worksheets

4-5

Chapter 12

Process description and reflection