Charles University in Prague
Faculty of Mathematics and Physics

BACHELOR THESIS

HakChol Pak

Simulation of traffic infrastructure

Department of Software Engineering

Supervisor: RNDr. Jan Kofroň, Ph.D.
Study Program: Computer Science, Programming

2010

I would like to thank my supervisor, RNDr. Jan Kofron, Ph.D, for the patient guidance and advice.

Contents

Název práce: Simulátor dopravní sítě
Autor: HakChol Pak
Katedra: Katedra softwarového inženýrství
Vedoucí bakalářské práce: RNDr. Jan Kofroň, Ph.D.
E-mail vedoucího: jan.kofron@d3s.mff.cuni.cz
Abstrakt: Cílem projektu je simulovat pohyb vozidel v silniční síti. Projekt se skládá ze dvou částí, simulátoru a editoru dopravní sítě. Editor dopravní sítě umožňuje uživateli vytvářet a upravovat dopravní sítě. Se zvolenou sítí lze simulovat pohyb vozidel a získat výsledek simulace. Simulace má dvě důležité části – model pohybu vozidel a ovládání semaforů.

Klíčová slova: simulace, dopravní simulace, mikroskopická simulace, semafor

Title: Simulation of traffic infrastructure
Author: HakChol Pak
Department: Department of Software Engineering
Supervisor: RNDr. Jan Kofroň, Ph.D.
Supervisor’s e-mail address: jan.kofron@d3s.mff.cuni.cz
Abstract: The aim of the project is to simulate the movement of vehicles in the traffic. The project consists of the two parts, the simulator and the editor of traffic network. Editor of traffic network allows a user to create and edit the traffic network. The second part of the program will be able to simulate the movement of vehicles in the traffic network and get simulation results. The simulation model consists of two important parts – the movement of the vehicles and traffic lights control.

Keywords: simulation, traffic simulation, micro-simulation, traffic light

Chapter 1

Introduction

This report is intended to document the bachelor project, whose name is “Simulation of traffic infrastructure”. It covers the analysis, implementation and user’s guide of the project.

1.1 Project Motivation

Traffic is an important aspect of the economy and our life. Every country invests lots of money to improve its state. Nevertheless, we are not satisfied with the current traffic system and we want some further development of it. For example, sometimes we meet traffic jam in the morning or in the evening, when people are going to or from the office. One of the possible solutions could be construction of new roads, but it is expensive. The program is intended to solve this problem without any reconstruction of the real traffic situation in some chosen city but to simulate traffic in general infrastructure. The simulation of traffic gives us information how to improve traffic system. Using the result of the simulation the traffic system might be optimized. However, building a complete simulation model is an uneasy task. Working on this project is a good chance to get some experience in both project development and also in building the simulation model.

1.2 Project Objectives

The project aims to build the simulation model, show its behavior and present its result in a graphical user interface. The program will provide an interface to edit the traffic network. The program will also provide an interface to specify parameters such as simulation speed and traffic intensity levels before simulation starts or dynamically change during the simulation. Finally, the program will provide statistical results for data gained from simulation.

In summary, there will be two parts to the project.

1. An interface to build any traffic network.

2. A simulation displaying the model run on the data supplied and producing statistical results.

1.3 Thesis Structure

Next chapter evaluates already existing simulators such as micro-simulation of road network, traffic simulation and traffic light simulations. Third chapter is about analysis of the project. Fourth chapter explains the implementation of the project in detail. Fifth chapter describes a user guide, which gives whole instruction for operating the program. The final chapter of the thesis is the conclusion of the project.

Chapter 2

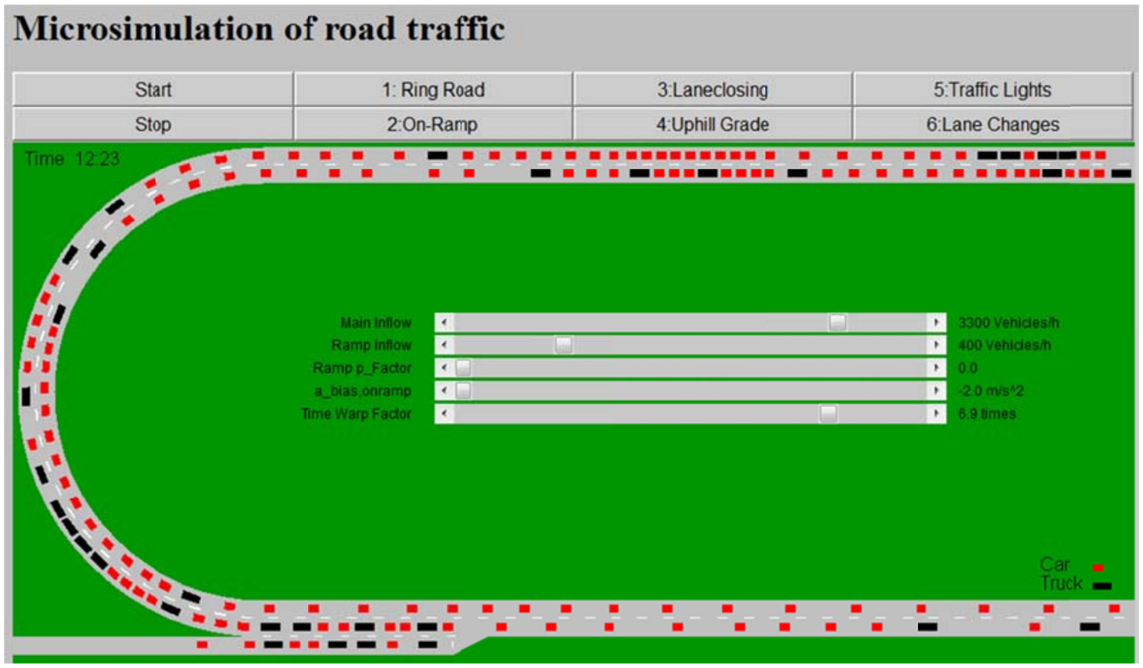
Existing programs

In this chapter, we evaluate already existing simulators to get information about the traffic simulation programs.

2.1 Micro-simulation of road traffic

Location: http://www.traffic-simulation.de/ [1]

Author: Dr Martin Treiber, Institute for Transports and Economics, Dresden University of Technology



Main Inflow 3300 Vehicles/h
Ramp Inflow 400 Vehicles/h
Ramp p_Factor 0.0
a_bias,onramp -2.0 m/s^2
Time Warp Factor 6.9 times

Figure 2.1: Screenshot of Micro-simulation of road traffic

The Micro-simulation of road traffic simulates one-directional road traffic on a single road with two lanes. It simulates the movement of vehicles of two types, cars and trucks. The simulation realized several traffic models such as the intelligent driver model and lane change model. This simulator is good at showing some basic behavior of traffic simulators. However, the simulator does not give any statistical result and does not allow editing the traffic network. It also does not realize the traffic light control model.

2.2 Traffic simulation

Location: http://www.phy.ntnu.edu.tw/oldjava/Others/trafficSimulation/applet.html [2]
Author: Kelly Liu, Department of Physics at National Taiwan Normal University

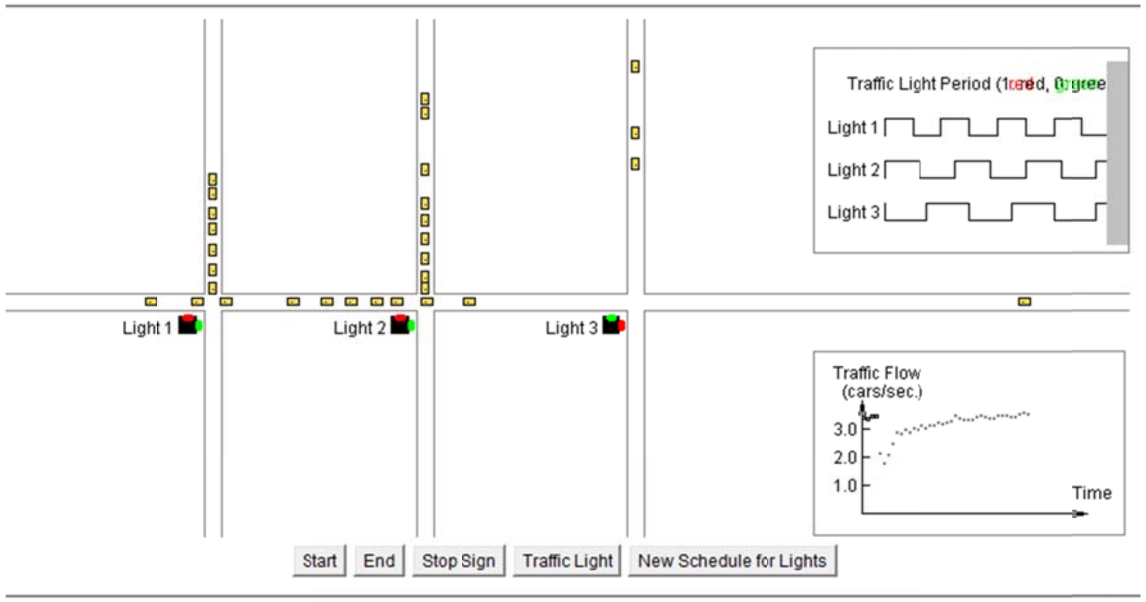


Figure 2.2: Screenshot of Traffic simulation

This program simulates the traffic network with three junctions. Each junction is controlled by either traffic lights or stop signs. The user can change the phasing of the lights from a short duration to a longer duration. This simulator is based on a simple traffic flow model. It produces a real time graph of the number of cars per second. This simulator is good at showing some basic behavior of traffic simulators. However, the roads are limited to one lane. This simulator does not allow editing the traffic network and does not give the usable statistical result except the number of cars per second. It simulates only 4-way junctions.

2.3 Traffic light simulation

Location: http://www.cs.washington.edu/homes/deibel/rt/ [3]

Author: Katherine Deibel, Computer Science & Engineering, University of Washington

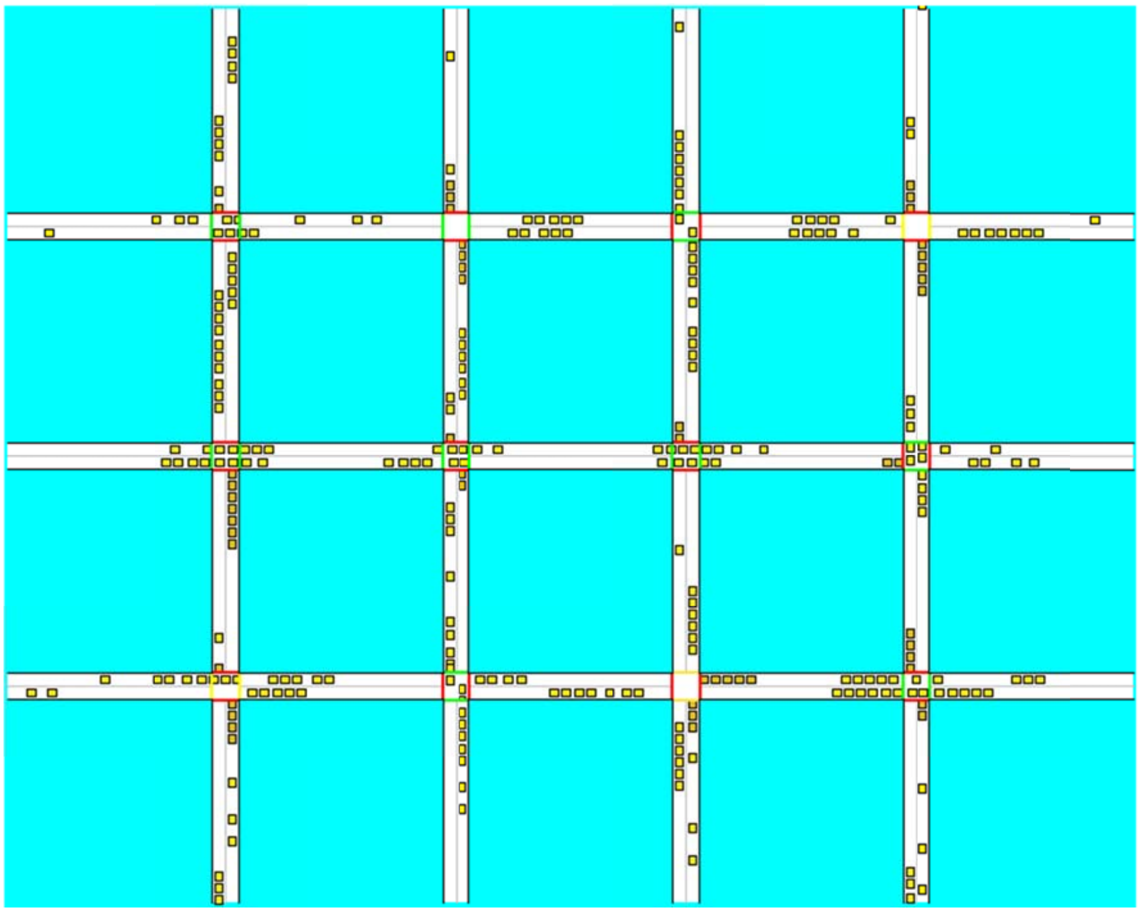


Figure 2.3: Screenshot of Traffic light simulation

This simulator simulates the traffic network in three different scenarios. The first scenario simulates the traffic network using the basic traffic light model. The traffic lights have a random duration. The duration of an amber light is constant. Traffic lights are not dependent on the number of vehicles on their incoming streets. The cars never turn onto cross streets, so they never turn left or right into other roads. The second and third scenarios involve the handling of ambulances. In the second scenario, the light is green if the ambulances are approaching a light. The cars pull over to the side if an ambulance is within a certain radius of the car and if there is a chance the car could be in the way of the ambulances. In the final scenario, the light is switched to the green before the ambulance arrivals so it does not need to slow down. This simulator is good at showing some basic behavior of traffic light control model. However, it does not provide any statistical result and does not allow editing the traffic network. It simulates only 4-way junctions.

Chapter 3

Analysis

In this chapter, we analyze project requirements and their possible solutions by describing advantages and disadvantages.

3.1 The simulator

3.1.1 Simulation model

First step in this analysis is a choice of a simulation model. There are two common approaches for simulation model, the macro-simulation model and micro-simulation model. The macro-simulation model evaluates traffic flow as a whole without consideration of the characteristics and features of individual vehicles in the traffic network [4]. The micro-simulation model simulates the behavior of individual vehicles in the traffic network. It considers the features and characteristics of the individual vehicles and uses vehicle following model. The aim of this project is to simulate the behavior of individual vehicles with showing of their movement in graphical user interface. Therefore, our simulation model must be taken as micro-simulation model.

3.1.2 Traffic flow model

Because the micro-simulation model was chosen in first step, it is necessary to build a traffic flow model. The traffic flow model simulate single vehicle-driver units, thus the dynamic variables of the modes represent properties like the position and velocity of a single vehicle [5]. The most important part of this model is to set the next position of a single vehicle through time. There are two possible models, the linear driver model and intelligent driver model [6]. The linear driver model sets next position of vehicles by linear function without considering characteristics of acceleration and deceleration of vehicles and the desired time headway to the vehicle in front. It has the advantage of the simple implementation. The disadvantage is that this model does not realize the behavior of vehicles in reality. In contrast to the linear driver model, the intelligent driver model realizes the behavior of vehicles in reality. It describes the dynamics of the positions and velocities of vehicles with considering the behavior of vehicles in reality. There exists the intelligent driver model definition by Helbing, Hennecke and Treiber. Nevertheless, this project will not use already existing model definition. I will build own intelligent driver model, because it is the important part of the project.

3.1.3 Traffic light control model

Building a traffic light control model is one of the important parts in the simulation of traffic infrastructure. However, the optimal control of traffic lights in the traffic network is highly complex problem. Therefore, most traffic lights in practice are controlled by fixed-cycle control models. It could be an effective solution for fixed-way junctions such as 3-way junction and 4-way junction. However, our project intended to support even more-way junctions. For more-way junctions there is no optimal control model yet. The program implements its own traffic light control model even for more-way junctions.

3.1.4 Route choice

After entry-point and exit-point of the vehicles are chosen, the simulation finds their shortest travel route. There are two typical algorithms to get the shortest path, the Dijkstra [7] and Bellman-Ford algorithm [8]. The Dijkstra algorithm finds the shortest path from one node to all nodes in a graph with non-negative edge weights. The complexity of this algorithm is quadratic (in the size of the graph). The Bellman-Ford algorithm finds the shortest path between each two nodes with the cubic complexity. It works even in a graph with negative edge weights. When comparing two algorithms, the Dijkstra algorithm is faster than the Bellman-Ford algorithm. In addition, the traffic network does not have negative edge weights. Therefore, I have chosen the Dijkstra algorithm.

3.1.5 GPS support

In the beginning of the project, GPS (Global Positioning System) support in the simulation was intended. It means that the simulation works with actual vehicles using GPS. However, I realized that GPS support is not necessary. The Global Positioning System (GPS) is a space-based global navigation satellite system that provides reliable location and time information in all weather and at all times and anywhere on or near the Earth when and where there is an unobstructed line of sight to four or more GPS satellites [9]. In other words, it gives information about the actual location of earth. In this case, there is no a need of the editor, because the editor does not design a traffic network in reality. It just could be used a real map data like the OpenStreetMap [10]. However, as the editor of the traffic network is the important part of the project, the project will not support GPS.

3.2 The editor

3.2.1 Traffic network design mode

The design mode of the traffic network should be easy and simple. So designing its parts should work in an intuitive way. For example, designing roads should be like drawing lines in a drawing package and each new node of a road should be connected with the previous node.

3.2.2 Storing a traffic network data

The editor of the traffic network should provide storing its data to load it in a simulation. It could be used an xml document or database to store the traffic network data. Using the database is convenient for storing and managing big amount of content. The disadvantage is that on a computer, on which the simulation is running, must be installed the database server. Using the xml document is useful if storing and managing small amount of content. It has the disadvantage of complex querying. However, the editor does not need lots of querying, so using the xml document suffices to store the traffic network data.

3.2.3 Multi-lane roads support

The traffic network consists of multi-lane roads. To display the movement of vehicles, the simulator should support at least two-lane roads, one lane in the direction of the road and another lane in the opposite direction. However, if it supports more multi-lane roads, the program will be more complex, because it must build a lane-change model yet. The program will support only two-lane roads to avoid the complexity problems.

3.3 Chosen technologies

The program is platform dependent on the Windows operation system. For being the independent platform, the project should be realized in Java. However, I am not such a good Java programmer. I have chosen .NET Framework 4.0 technologies to realize the project. The project has a modern graphical user interface according to the requirements. In order to satisfy this condition, we can choose .NET Windows or .NET Windows Presentation Foundation technology [11]. The .NET Windows Presentation Foundation is stronger than .NET Windows for representing graphical objects. Therefore, I have chosen .NET Windows Presentation Foundation technology to realize the project.

Chapter 4

Implementation

This chapter explains the implementation of the project in detail. Firstly, we show the object model of the project and then we describe the implementation.

4.1 Object model

The object model consists of several class packages divided by functionality. It helps to have less relationship between each class and understand the system quickly.

Class packages designed in this project are following.

- Map – represents the traffic network. It contains the classes **Map**, **Way**, **Node**, **OutLine**, **CenterLine** and **Lane**.
- DrawManager – allows to draw all graphical objects in the project. It contains the classes **DrawManager**, **ObjectShape** and inherited classes from the **ObjectShape** class such as the classes **CenterlineShape**, **OutlineShape**, **NodeShape**, **EdgeShape** and **VehicleShape**. The **DrawManager** class manages the object drawing **ObjectShape** on the drawing panel. The **ObjectShape** class represents the shape of the object.
- XMLMananger – handles saving and loading of the traffic network data to and from the xml file. It contains the classes **XMLManager**, **NodeElement** and **WayElement**. This package uses the **Map** package when importing or exporting the traffic network data.
- TimerManager – manages all aspects of the simulator through the time. It contains the classes **TimerManager**, **GenerateManager**, **MoveManager,** **TrafficLightManager** and **Timer**. The class **GenerateManager** generates vehicles during the simulation. The **MoveManager** class realizes the movement of vehicles through time. The **TrafficLightManager** class serves to control all traffic lights. This package gets information about vehicles and traffic lights from the classes **VehicleList** and **TrafficLightList**.

- Vehicle – represents the vehicle. It contains the classes **Vehicle**, **Car** and **Truck**. The **Vehicle** class has its simulation model and route choice algorithm.
- SimulationModel – realizes the simulation model. It contains the interface **SimulationModel**, the class **IDM**, **IDMCar** and **IDMTruck**. The **IDM** class represents the intelligent driver model.
- PathAlgorithm – calculates the vehicle's route. It contains the class **Dijkstra**, **ShortestAlgorithm** and **PathAlgorithm**.
- TrafficLight – controls the behavior of the traffic light. It contains the classes **TrafficLight**, **LongestQueueFirst** and **LongestWaitFirst**. The **LongestQueueFirst** class represents the traffic light, which lets pass the vehicles from the longest queue at first. The **LongestWaitFirst** class represents the traffic light, which lets pass the vehicles from the longest waiting queue at first.
- StatisticsManager – provides the statistical result of the simulator. It contains the class **StatisticsManager**.
- GeometryManager – provides all geometric calculations used in the project. It contains the class **GeometryManager**.

Figure 4.1 shows the relationships between each of the classes in the project.

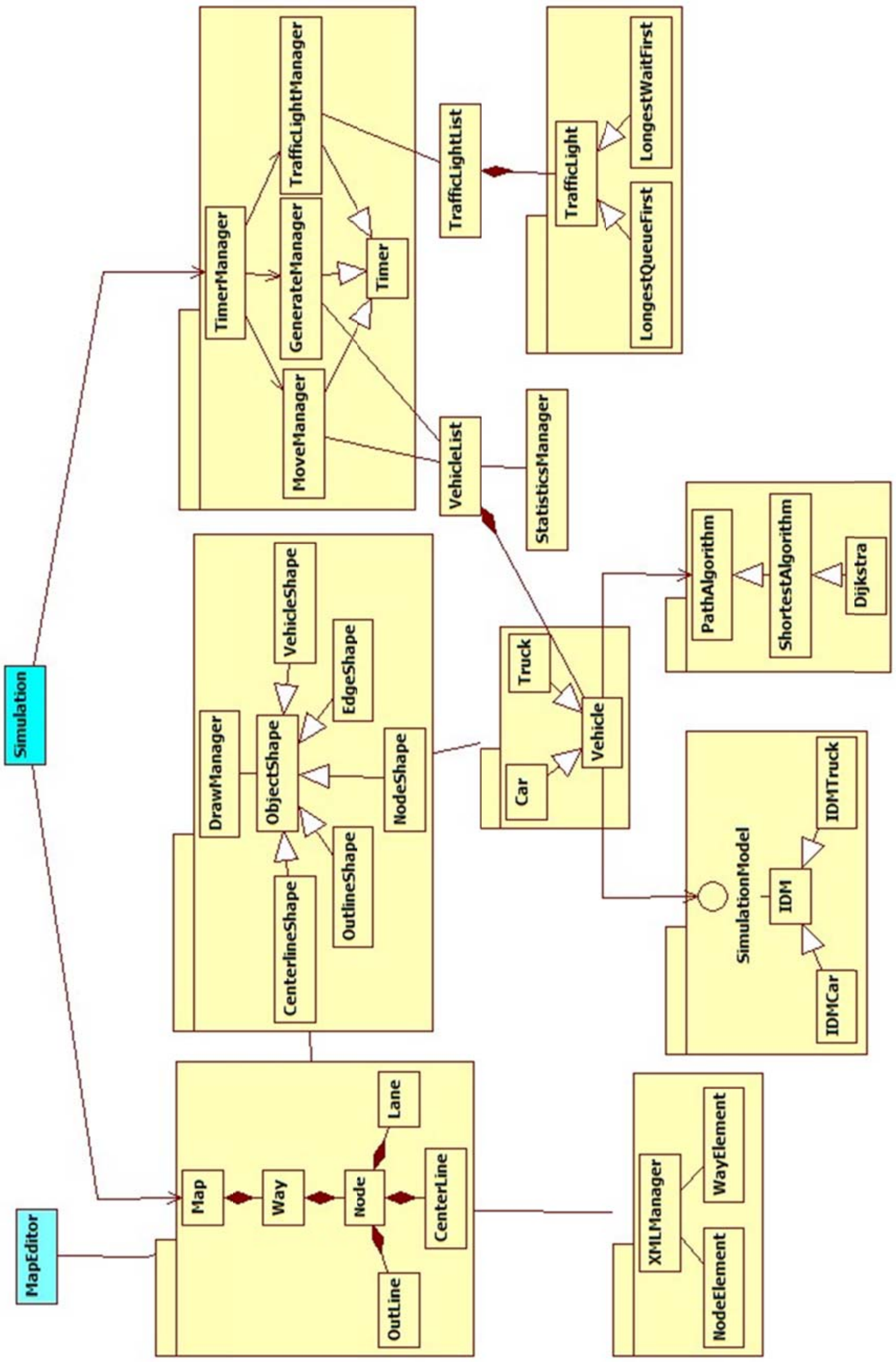


Figure 4.1: Class diagram of the project

4.2 Implementation details

4.2.1 Generating vehicles

The **GenerateManager** class generates vehicles during the simulation. The **GenerateVehicle** method in the class generates one vehicle at regular intervals. This interval is determined by two parameters of the simulator: “simulation speed” and “vehicle flow”. The parameter “simulation speed” indicates how fast the simulation runs. The parameter “vehicle flow” indicates how many vehicles is generated per hour. Before generating the vehicle, its type(car or truck) have to be chosen. The type of the vehicle is determined by the parameter “car percentage”. The program generates the random value between 0 and 100. If this value is smaller than the value of the parameter “car percentage”, it generates the car. Otherwise, it generates the truck. When a vehicle is generated, the generator sets an entry point and exit point of the vehicle. The entry point and exit point are nodes, which has one degree in the traffic network in order to show the leaving and arrival of the vehicle clearly. When the entry point and exit point are chosen, the generator calculates its shortest route. Finally, the generator adds the vehicle to the list of vehicles, which is represented by the **VehicleList** class.

4.2.2 Route choice

The project realizes one algorithm for a route choice. The **Dijkstra** class calculates the shortest route of the vehicle from the entry to the exit. It realizes the Dijkstra’s algorithm [7]. The **GetPath** method in the **Dijkstra** class returns the vehicle shortest route.

4.2.3 Movement of vehicles

The **MoveManager** class provides the movement of the vehicles. The **MoveVehicles** method in the class moves the vehicles. It moves each vehicle from the **VehicleList** class by the intelligent driver model. The **IDM** class realizes the intelligent driver model. The **CalcAcc** method in the class returns the acceleration of the vehicle. When it is given the acceleration of the vehicle, the **Move** method in the **Vehicle** class sets the next position of the vehicle and moves the representation of the vehicle to the next position.

4.2.4 Traffic light control

The **TrafficLightManager** class manages the control of traffic lights. The **InitTrafficLights** method in the class initializes each traffic light with the traffic light control model LongestQueueFirst or LongestWaitFirst. The LongestQueueFirst model releases first the longest queue. This model is realized by the **LongestQueueFirst** class. The LongestWaitFirst model releases first the longest waiting queue. This model is realized by the **LongestWaitFirst** class. The **UpdateTrafficLights** method in the class updates the status of each traffic light from the **TrafficLightList** class.

4.2.5 Calculating statistical result

The results of the simulation are the statistical results. The static **StatisticsManager** class returns the statistical results of the simulation. The **InitStatistics** method in the class initializes the statistical results and the **UpdateStatistics** method updates the statistical results. Both methods get information about vehicles from the **VehicleList** class.

The class calculates the statistical result as follows

- How many vehicles have entered the traffic network.
- How many vehicles have exited the traffic network.
- How many kilometers have travelled.
- How long have the vehicles travelled.
- The average speed of all the vehicles.
- The delay time of all vehicles.

4.2.6 Drawing objects

In the project, each object, which is drawn on the drawing panel, has its shape object. For example, the **Vehicle** class has its shape class - **VehicleShape**. The **VehicleShape** class represents the shape of the vehicle on the drawing panel. Such classes are inherited from the **ObjectShape** class. Then there is the **DrawingManager** class to manage the **ObjectShape** class. There are three important methods in the class: **Draw**, **Remove** and **Clear**. The **Draw** method draws the object shape to the drawing panel. The method

Remove removes the object shape from the drawing panel. The **Clear** method clears the drawing panel.

4.2.7 Import & Export a traffic network data

The project uses the xml file to import and export a traffic network data. The **XMLManager** class provides loading and saving of the traffic network data. There are two important methods: **LoadMap** and **SaveMap**. The **LoadMap** method loads the traffic network data from the xml file. The **SaveMap** method exports the traffic network data to the xml file.

Below there is the xml file structure of the traffic network that is written in DTD (Document Type Definition).

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT map (nodes, ways)>
        <!ELEMENT nodes (node*)>
                <!ELEMENT node EMPTY>
                <!ATTLIST node id ID #REQUIRED>
                <!ATTLIST node x CDATA #REQUIRED>
                <!ATTLIST node y CDATA #REQUIRED>
        <!ELEMENT ways (way*)>
                <!ELEMENT way (nd*)>
                <!ATTLIST way id ID #REQUIRED>
                        <!ELEMENT nd EMPTY>
                        <!ATTLIST nd ref IDREF #REQUIRED>
```

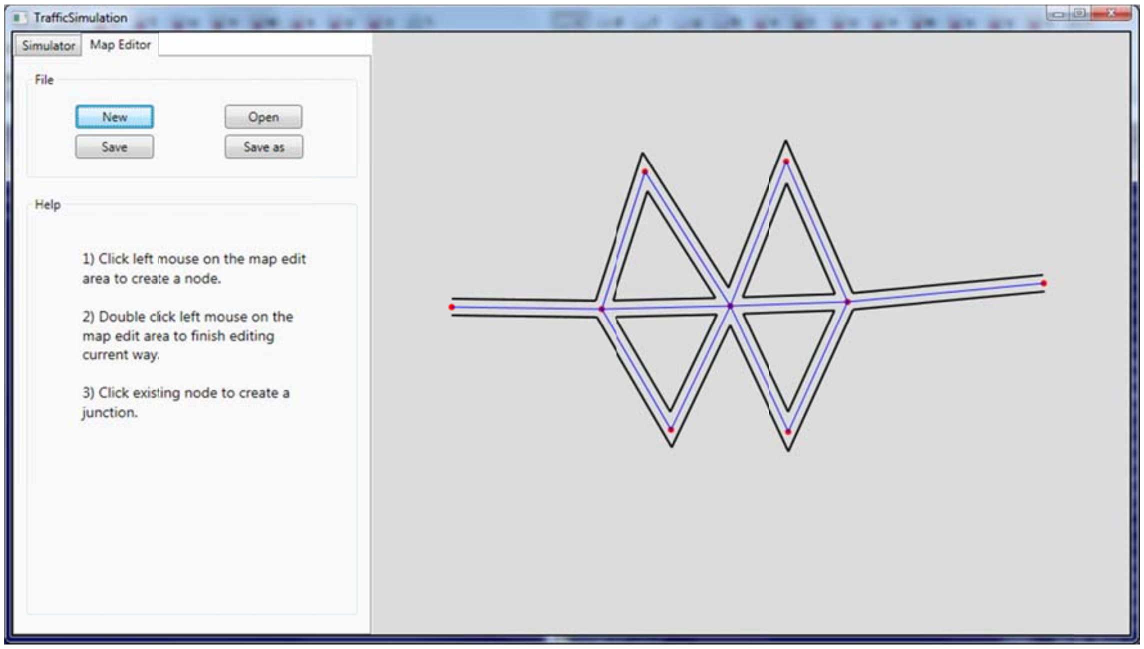
Chapter 5

User's Guide

This chapter gives users a complete guide of the program. It explains how to create a traffic network and run a simulation.

5.1 The editor

The editor is the part of the program in which a traffic network can be created, saved, loaded and modified. It consists of two sections, a main menu and drawing section.



1) Click left mouse on the map edit area to create a node.

2) Double click left mouse on the map edit area to finish editing current way.

3) Click existing node to create a junction.

Figure 5.1: The editor interface

5.1.1 Main menu

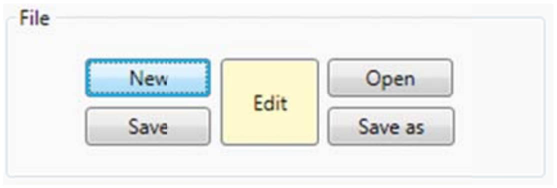


Figure 5.2: The main menu of the editor

The main menu is on the left and top of the editor. It consists of several buttons such as "New", "Open", "Save", "Save as" and "Edit".

Functionality of each button:

- New - creates the new traffic network. By clicking the button the content of the drawing panel (on the right side of the window) is erased and ready to draw the new traffic network.
- Open - opens existing traffic network. The file extension is ".map".
- Save - saves the traffic network into a file. If traffic network has been made up before then the changes are saved to corresponding file. Otherwise, behaves like the button "Save as".
- Save as - saves the traffic network into the file with specified name.
- Edit - modifies the traffic network loaded for a simulation.

5.1.2 Designing a traffic network

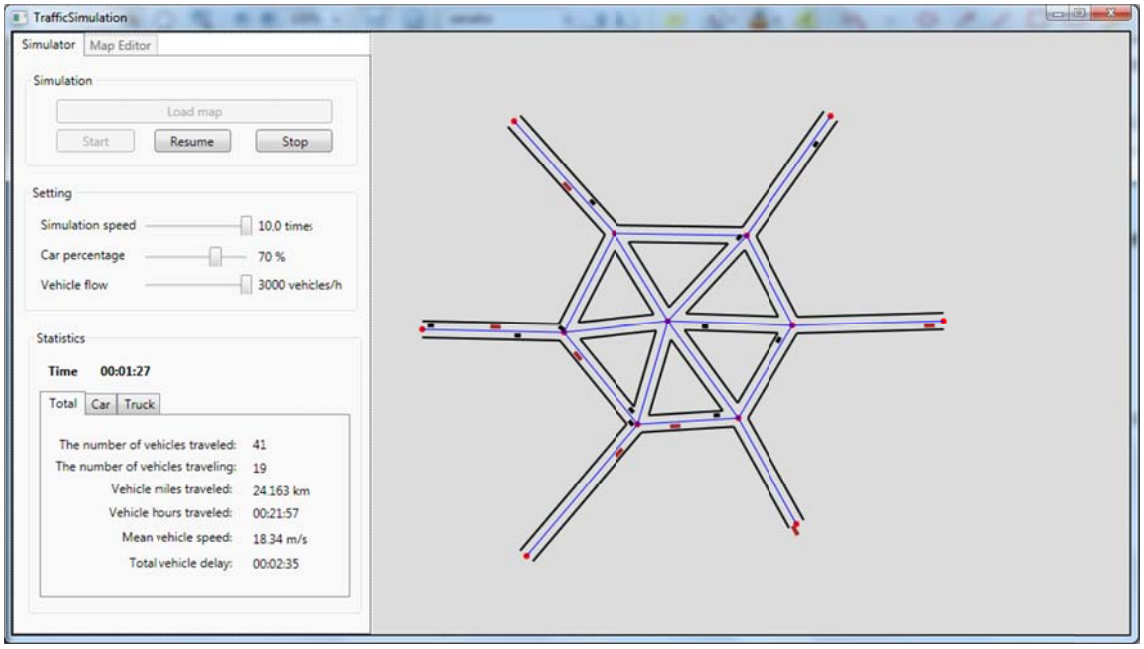
A traffic network consists of roads. A road consists of nodes. Construction of a traffic network is initiated by clicking on the design panel situated on the right part of the window.

How to design a traffic network:

- Single click by a mouse left button
    - If it is the first click after finished construction of the previous road, it creates a new road and it adds a start node into the created road.
    - If it is not the first click in the current road design, it adds one node into the current road.
    - If it is clicked on some already existing node, the previously generated node and the node that was clicked on are connected with a new road.
- Double click by a mouse left button
    - It finishes drawing the current road.

5.2 The simulator

The simulator shows a run of the simulation - the movement of vehicles on the loaded traffic network and gives the statistical result of the simulation. It consists of four sections, main menu, setting parameters of the simulation, statistical results and vehicle movement section.



Simulation speed 10.0 times
Car percentage 70 %
Vehicle flow 3000 vehicles/h

Time 00:01:27

The number of vehicles traveled:	41
The number of vehicles traveling:	19
Vehicle miles traveled:	24.163 km
Vehicle hours traveled:	00:21:57
Mean vehicle speed:	18.34 m/s
Total vehicle delay:	00:02:35

Figure 5.3: The simulator

5.2.1 Main menu

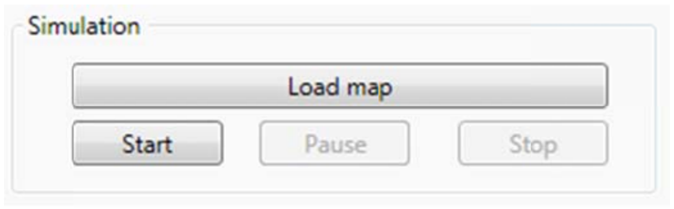


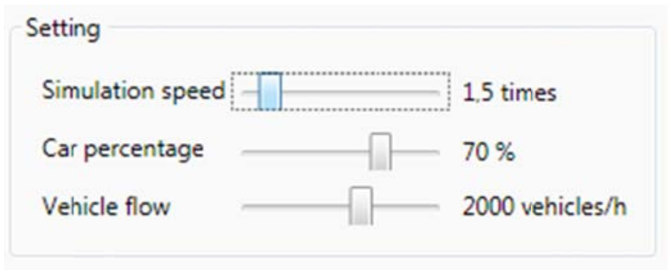
Figure 5.4: The main menu of the simulator

The main menu is at the top of on the left side of the window. It consists of several buttons: “Load map”, “Start”, “Pause”, “Resume” and “Stop”.

Functionality of each button:

- Load map - loads the traffic network from a file. On this network will be simulated the movement of vehicles.
- Start - starts the new simulation. This button is disabled, if there is not loaded any traffic network.
- Pause - pauses the running simulation. By clicking this button its text and function changes to “Resume”.
- Resume - resumes the paused simulation. By clicking this button its text and function changes to “Pause”.
- Stop - stops the running simulation.

5.2.2 Setting parameters of the simulator


Setting

Simulation speed 1,5 times

Car percentage 70 %

Vehicle flow 2000 vehicles/h

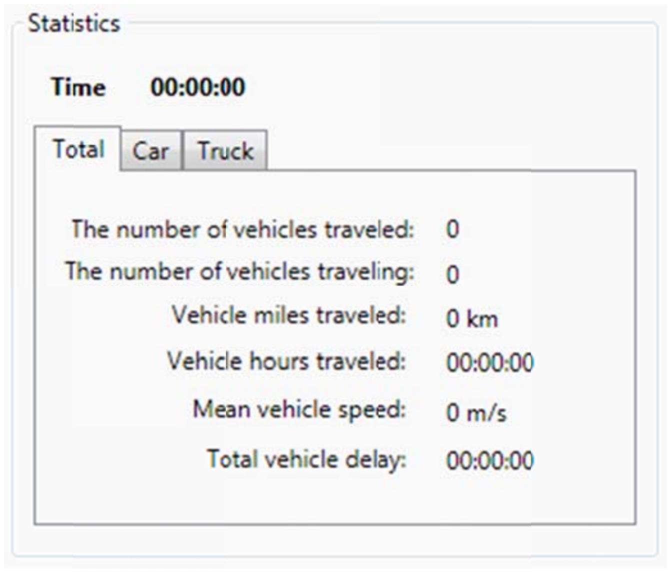
Figure 5.5: Parameters Setting

Just below the main menu of the simulator is a section for parameters settings. Parameters could be changed at any time.

Meaning and value interval of each parameter is as follows.

- Simulation speed - This parameter indicates the value of how fast to run the simulation. The higher is the value of this parameter the higher is speed of simulation. Its range is in from 0.1 to 10.0. The default value is 1.5.
- Car percentage - This parameter indicates the percentage of generated cars from their total number. It is in the range 0 to 1000. The default value is 70.
- Vehicle flow - This parameter indicates the value of how many vehicles are generated per hour. Its range is in from 0.1 to 10.0. The default value is 2000.

5.2.3 Statistical result


Statistics

Time **00:00:00**

```
The number of vehicles traveled:  0
The number of vehicles traveling: 0
         Vehicle miles traveled:  0 km
         Vehicle hours traveled:  00:00:00
             Mean vehicle speed:  0 m/s
            Total vehicle delay:  00:00:00
```

Figure 5.6: The statistical result of the simulation

The statistical results section is at the bottom on the left side of the window. It also contains statistics of the simulation according to type of the vehicles and all types of vehicles. There are two types of vehicles, car and truck. Therefore, there are three different statistics.

Meaning of each parameter is as follows.

- The number of vehicles traveled - This parameter indicates the number of vehicles that have already finished driving (went in and out).
- The number of vehicles traveling - This parameter indicates the number of vehicles in the traffic network.
- Vehicle miles traveled - This parameter indicates the traveled distance of vehicles that have passed and that are moving just now (all vehicles in the traffic network).
- Vehicle hours traveled - This parameter indicates the traveled time of vehicles that have passed and that are moving just now (all vehicles in the traffic network).
- Mean vehicle speed - The value of this parameter equals "Vehicle miles traveled" / "Vehicle hours traveled".

- Total vehicle delay - This parameter indicates the delayed time of vehicles that have passed and that are moving just now (all vehicles in the traffic network).

Parameters mentioned above are only for all types of vehicles together. There are two identical sets of parameters for cars and trucks too.

5.2.4 Simulation running section

The simulation run section is on the right side of the simulator window. This section shows the movement of vehicles on the traffic network in time.

Chapter 6

Conclusion

The aim of the project was to simulate the movement of vehicles in the traffic network. In order to solve the project specification, there were the simulator and editor of the traffic network realized. In detail, we produced the editor that enables to design a traffic network easily and simply in graphical user interface. We also made up the simulator that simulates the movement of vehicles in the traffic network. In the simulator, we built its own intelligent driver model and traffic light control model. The only deviation from the project specification is that the simulator does not support GPS. We already mentioned this deviation in chapter 3.

Although almost all project specifications were implemented, there still exist further development possibilities. In the project, the traffic network consists of two-lane roads. To get more correct simulation results it should consist of multi-lane roads. The traffic light control model might be more optimal. For the vehicle routes the program used the routing algorithm to get the shortest path. The routing algorithm to get the quickest path might be added.

I have learned how to build the traffic simulation model and realize it when working on this project.

Bibliography

[1] Martin Treiber, Institute for Transports and Economics, Dresden University of Technology
Micro-simulation of road traffic
http://www.traffic-simulation.de

[2] Kelly Liu, Department of Physics at National Taiwan Normal University
Traffic Simulation
http://www.phy.ntnu.edu.tw/oldjava/Others/trafficSimulation/applet.html

[3] Katherine Deibel, Computer Science & Engineering, University of Washington
Traffic Light Simulations
http://www.cs.washington.edu/homes/deibel/rt

[4] Mark Yand, Principal, DKS Associates
Macro versus Micro Simulation Modeling Tools
http://www.dksassociates.com/admin/paperfile/9A_Yand_Paper.pdf

[5] Wikpedia
Microscopic traffic flow model
http://en.wikipedia.org/wiki/Microscopic_traffic_flow_model

[6] Wikipedia
Intelligent Driver Model
http://en.wikipedia.org/wiki/Intelligent_Driver_Model

[7] Wikipedia
Dijkstra's algorithm
http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

[8] Wikipedia
Bellman-Ford algorithm
http://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm

[9] Wikipedia
Global Positioning System
http://en.wikipedia.org/wiki/Gps

[10] OpenStreetMap
OpenStreetMap
http://www.openstreetmap.org/

[11] MSDN
Windows Presentation Foundation
http://msdn.microsoft.com/en-us/library/ms754130.aspx