SUMO 2017 – Towards Simulation for Autonomous Mobility

Proceedings

Berichte aus dem DLR-Institut
für Verkehrssystemtechnik

Band 31

DLR Deutsches Zentrum
für Luft- und Raumfahrt

Berichte aus dem DLR-Institut für Verkehrssystemtechnik

Volume 31

Proceedings of the SUMO2017

Towards Simulation for Autonomous Mobility

May 8 – 10, 2017
Berlin-Adlershof

Publisher:
Deutsches Zentrum für Luft- und Raumfahrt e. V.
Institut für Verkehrssystemtechnik
Lilienthalplatz 7, 38108 Braunschweig
Telephone +49 (0) 531-295-3401
Telefax +49 (0) 531-295-3402

ISSN: 1866-721X

DLR-TS 1.31

Berlin, May 2017

Institute Director:
Prof. Dr.-Ing. Karsten Lemmer

Preface

Dear reader,

You are holding in your hands a volume of the series „Reports of the DLR-Institute of Transportation Systems“. We are publishing in this series fascinating, scientific topics from the Institute of Transportation Systems of the German Aerospace Center (Deutsches Zentrum für Luft- und Raumfahrt e.V. – DLR) and from his environment. We are providing libraries with a part of the circulation. Outstanding scientific contributions and dissertations are here published as well as projects reports and proceedings of conferences in our house with different contributors from science, economy and politics.

With this series we are pursuing the objective to enable a broad access to scientific works and results. We are using the series as well as to promote practically young researchers by the publication of the dissertation of our staff and external doctoral candidates, too. Publications are important milestones on the academic career path. With the series „Reports of the DLR-Institute of Transportation Systems / Berichte aus dem DLR-Institut für Verkehrssystemtechnik“ we are widening the spectrum of possible publications with a building block. Beyond that we understand the communication of our scientific fields of research as a contribution to the national and international research landscape in the fields of automotive, railway systems and traffic management.

This volume contains the proceedings of the SUMO Conference 2017 which was held from 8th to 10th May 2017 with a focus on autonomous mobility. In the current transition process traffic simulation is the only tool which can give us insights in the mechanisms of traffic in largely automatized traffic scenarios. SUMO as an open source tool provides a wide range of traffic planning and simulation functionalities to support the scientific community. The conference proceedings offer an overview of the applicability of the SUMO tool suite as well as its universal extensibility due to the availability of the source code. The major topic of this fifth edition of the SUMO conference is the calibration of simulation to real world or handbook data as well as communicating networks of intelligent vehicles. A number of contributions cover heterogeneous traffic networks, junction control and new traffic model extensions to the simulation. Subsequent specialized issues such as emission modelling and personal rapid transit simulation are targeted as well. At the conference the international user community exchanged their experiences in using SUMO. With this volume we provide an insight to these experiences as inspiration for further projects with the SUMO suite.

Prof. Dr.-Ing. Karsten Lemmer

Contents

1 Simulation and Characterisation of Traffic on Drive Me Route around Gothenburg using SUMO

Eric Bjärkvik, Florian Fürer, Mitra Pourabdollah, and Björn Lindenberg; Volvo Car Group, Gothenburg, Sweden
{eric.bjarkvik, florian.fuer, mitra.pourabdollah, bjorn.lindenberg }@volvocars.com

1.1 Abstract

In 2017, Volvo Car Group will demonstrate a large-scale trial of autonomous vehicles on public roads in a project called Drive Me. One hundred customers will use autonomous vehicles on the Drive Me route around the Swedish city of Gothenburg. This dual carriageway ring road has heavy traffic during rush hour and is separated from pedestrians and intersecting traffic for most of the route. To evaluate how autonomous vehicles interact in traffic and what impact they will have on traffic and energy efficiency, the environment on this 40 kilometer long route is modeled. In this paper, SUMO is used to simulate the Drive Me route by matching features such as road geometry, vehicle and bus lanes, intersections, speed limits, and car inflow and outflow rates. The traffic environment is first verified with the default Krauss car following model and at a later stage, the Intelligent Driver Model is used with parameters that were optimized to replicate driving behavior from recorded trips on the Drive Me route. The results show that it is possible to replicate traffic of a large scale road network in SUMO. However, when using the Intelligent Driver Model with parameters optimized to replicate the driver behavior on the route, some problems occur which are discussed.

Keywords: SUMO, Traffic Simulation, Autonomous Drive, Driver Model, Intelligent Driver Model, Drive Me, Microscopic traffic flow modeling, traffic flow.

1.2 Introduction

Autonomously driven vehicles are becoming more and more popular among car manufacturers and are soon to be introduced in real traffic. An autonomous vehicle has the ability to react faster than a human and does not get tired or distracted which could result in fewer traffic accidents (6). The driving style of an autonomous vehicle can be optimized for safety, fuel economy, traffic flow, or travel time based, depending on the desire of users and legislators (10). This can have significant impact on traffic systems, i.e., traffic flow and emissions (9). In order to analyze the effects of autonomous vehicles on traffic flow and emissions, a well defined description of the current traffic is needed. Once the description of the current traffic is available, autonomous vehicles can be inserted into the system to analyze their effects.

Traffic simulation softwares provide a cost effective solution to evaluate a traffic system. A traffic system description should include what fraction of cars, buses, and trucks utilize the road network. It should also include traffic density measurements at different times of the day and basic weather data (2).

Traffic simulations are commonly divided into three main groups, i.e., macroscopic, mesoscopic, and microscopic. In order to study the detailed effects of autonomous vehicles on traffic flow and emissions, microscopic traffic simulators such as SUMO (3) can be used to describe the traffic with a high level of detail.

One important part of traffic simulations is the car following model, which governs the longitudinal behavior of vehicles. A car following model usually consists of differential equations describing the dynamics of a vehicle such as position and velocity. Since accuracy of the results from traffic simulations greatly depend on the car following model, they have to reflect the behavior of drivers in real traffic.

Another crucial part of driver models in traffic simulations is the lane changing algorithm which governs how vehicles interact when changing lanes. Lane changing is in general complex; it requires that cars adapt their own speed to increase their chances of finding a gap and also that other cars create gaps for them. The model should not only resolve these problems so traffic can flow, but also replicate human behavior so that traffic congestion occurs in a similar way as in reality.

In this paper, the steps taken to build the traffic environment of an autonomous driving approved route in the traffic simulation tool SUMO are described. The so called Drive Me route is the road on which one hundred Volvo customers will test autonomous vehicles in the first pilot of the Drive Me project in 2017 in Gothenburg, Sweden. The traffic is simulated during rush hour and the results are validated using traffic data from Trafikverket (TrV) [1]. Both the default car following model in SUMO, the Krauss model, and the Intelligent Driver Model (IDM) are used in the simulations.

This paper is organized as follows. In Section 1.3 an overall view of the problem including the description of the Drive Me project and route, together with the steps taken to implement the Drive Me route in SUMO are given. The car following models used in SUMO, i.e., the Krauss and IDM, and the lane changing model is also presented. In Section 1.4, the result of the simulation using the default Krauss car following and lane changing models are first presented. Then IDM is used which was calibrated to represent driver behavior similar to drivers on the Drive Me route. The result of both cases are verified against measured data. Finally, conclusions are drawn.

1.3 Method

In this section an introduction to the Drive Me project is given, the Drive Me route is characterized and the initial steps to build the simulation environment such as roads and vehicle routes are given. Then, the two car following models Krauss and IDM are presented. Lastly the lane changing model used in the simulation is described.

1.3.1 Drive Me Project and Route

Drive Me is a research project aiming to investigate the effects of autonomous vehicles on personal mobility. It will also study the technical and societal needs in order to make autonomous vehicles a reality. In 2017, Volvo Car Group together with several public and academic partners, will start the first pilot of the Drive Me project, involving 100 customers driving autonomous vehicles on public roads in Gothenburg, Sweden (8). The route in Gothenburg, shown in Fig. 1.1, is a 40 km long typical commuting route, where customers are expected to benefit most from an autonomous vehicle. The road type of the Drive Me route is mainly dual carriageway with 2-3 lanes in each direction with speed limits between 50 and 80 kph and no access for bicycles or pedestrians.

In order to better understand the traffic on this route, data from Volvo Car's Field Operation Test Data project is analyzed. The database includes around 160'000 recorded trips on the Drive Me route, with signals collected from the CAN-bus of three different Volvo vehicle models: V60, V70,

[1] Trafikverket is the Swedish Transport Administration, http://www.trafikverket.se/

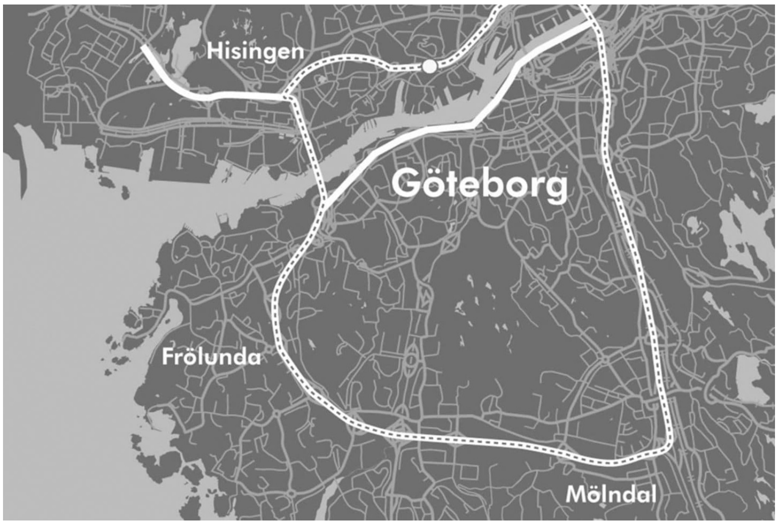


Figure 1.1: The Drive Me route around Gothenburg.

and XC70. The data includes GPS-position, time of day, car velocity, velocity of and distance to the car in front detected by the radar, and on-board diagnostics signals, such as current gear, engine speed and fuel consumption. The starting time of the trips, depicted in Fig. 1.2, shows that the traffic density on the Drive Me route peaks around 7:00 to 8:00 and 16:00 to 17:00.

1.3.2 Drive Me Route in SUMO

To simulate the Drive Me route in SUMO [2], the road network is imported from OpenStreetMap, and later manually edited with NETEDIT in SUMO. The junctions and intersections are manually corrected and speed limits, traffic lights, and bus lanes including bus stops are added to the network. All entrance and exit roads to the Drive Me route are included and to get a realistic inflow of cars to the route, parts of the connecting roads are also modeled, as shown in Fig.1.3.
For a better visualization, a satellite background is added to the simulation network. The background consists of three images that all are a collection of merged satellite images from Eniro[3], a Swedish map provider. Since there is a limitation on the number of background images that can be used in SUMO, this method provides a solution for a high resolution background despite the size of the traffic model. Part of the Drive Me route is shown in more detail in Fig.1.4.
Inflows and outflows are derived from a VISUM traffic model executed and owned by the city of Gothenburg. The traffic demand, i.e., the inflows and outflows between 16:00 to 17:00 are shown in Figure 1.5. Based on these inflows and outflows, vehicle route definitions are converted into SUMO format and matched with the built road network.
There are many factors that need to be considered in order to assure a representative traffic simulation environment, however, in this paper traffic flow and average vehicle speed has been chosen as performance indicators. The traffic flows and the average speed of the vehicles on 15 stations on the Drive Me route in both directions, as seen in Fig. 1.6, are measured by the Swedish Transport Administration during afternoon peak hour between 16:00 to 17:00.

1.3.3 Car Following Model

As mentioned earlier, the accuracy of the simulation greatly depends on the car following model. In this section, two car following models used in SUMO, i.e., Krauss and IDM, are presented.

[2] SUMO 0.29.0 (Simulation of Urban MObility x64) [3] Eniro, http://www.eniro.se/

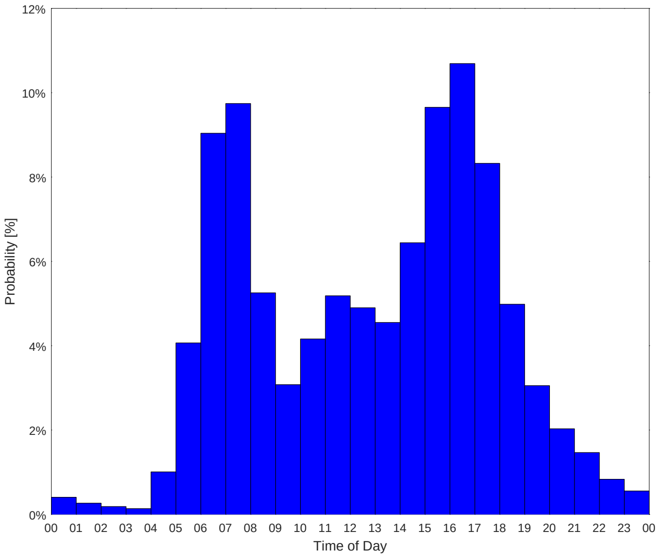


Figure 1.2: The distribution of starting times of the trips on the Drive Me route.

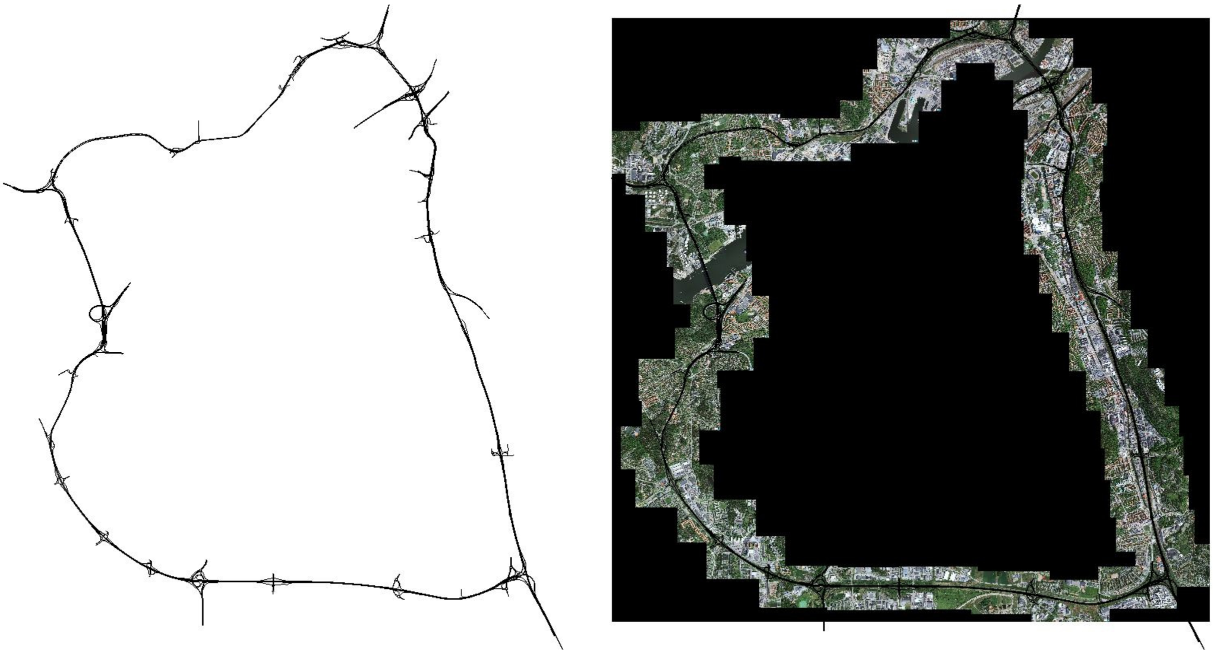


Figure 1.3: A figure of the Drive Me route in SUMO, without and with satellite background pictures.

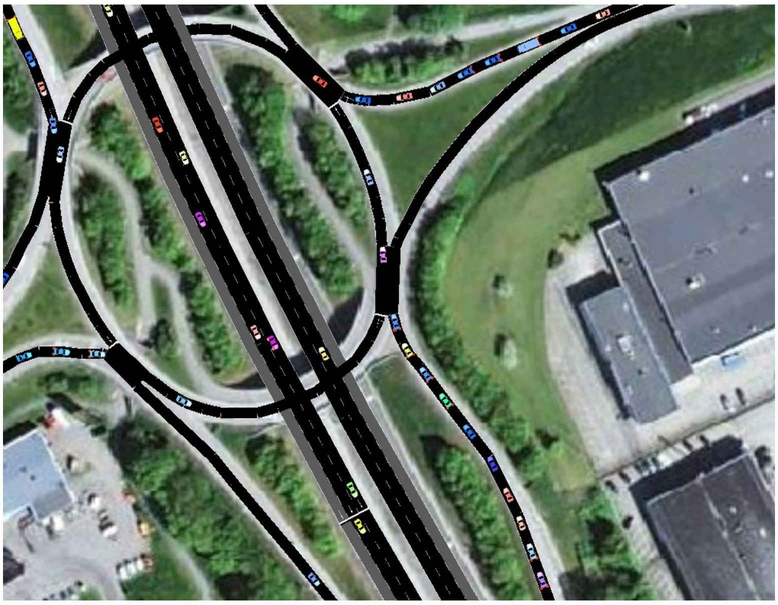


Figure 1.4: Part of the satellite background of the Drive Me route used in the SUMO simulation.

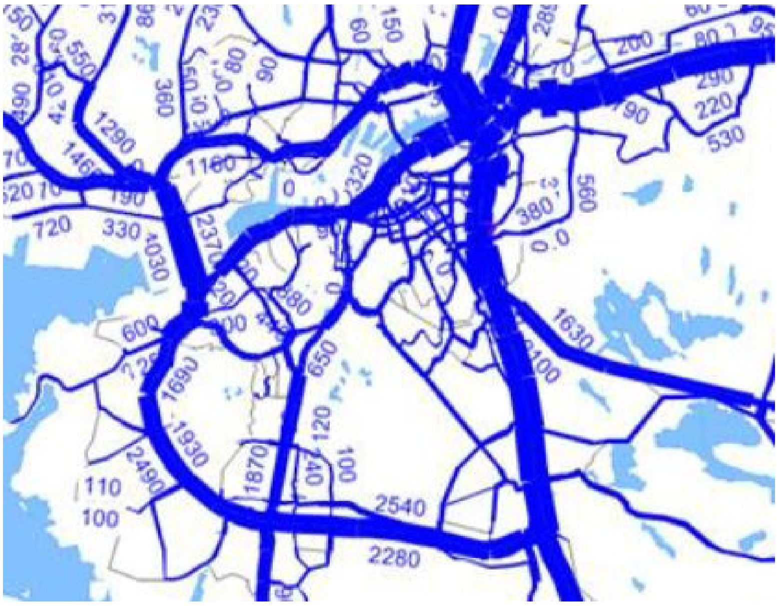


Figure 1.5: The inflows and outflows from a VISUM traffic model for rush hour between 16:00 to 17:00 on and around the Drive Me route.

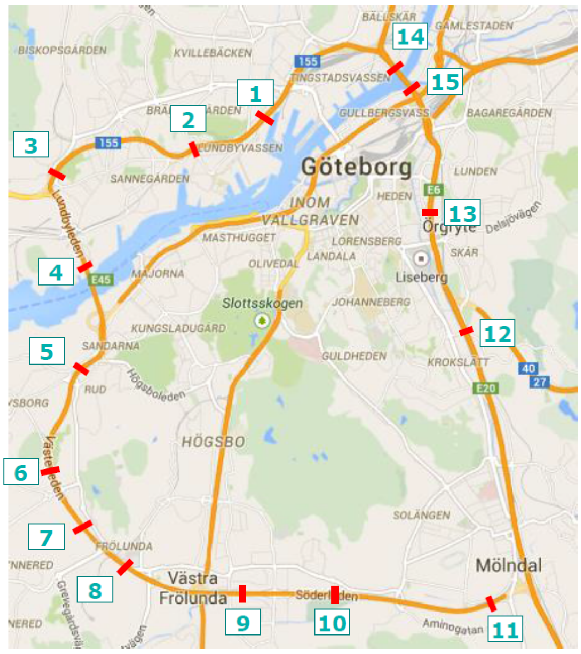


Figure 1.6: The 15 measurement stations on the Drive Me route, where the flow and average speed of the vehicles are measured, are shown with red marks.

Krauss Car Following Model

The default car following model in SUMO is a modified version of the Krauss car following model, developed by Stefan Krauss in 1997 (4). The model calculates the desired velocity, v_{des}, according to a safe velocity. The main parameters of the Krauss model are the acceleration ability $accel$, the deceleration ability $decel$, the driver's imperfection $sigma$, maximum desired velocity, $maxSpeed$, minimum desired net distance to the front vehicle $minGap$, and the driver's desired time headway tau.

Intelligent Driver Model

Another standard car following model in SUMO is the IDM which was initially proposed by Treiber et al. in (7). The model calculates the set acceleration of the vehicle as a function of the follower vehicle's current velocity, distance to the front vehicle, and velocity difference from the front vehicle, at each time instant. The main parameters of the IDM are $accel$, $decel$, $maxSpeed$, $minGap$, tau, and acceleration exponent $delta$.

1.3.4 Lane Changing Model

Another factor that influences the traffic flow significantly is the lane changing model. The lane changing model used in this study is the LC2013 which has several parameters such as, $lcStrategic$ i.e., driver's eagerness to perform strategic lane changing, $lcCooperative$ i.e., the driver's willingness to perform cooperative lane changing, $lcSpeedGain$ i.e., the driver's eagerness to perform lane changing to gain speed, and $lcKeepRight$ i.e., the driver's eagerness to follow the obligation to keep right (1).

1.4 Simulation Environment

In order to make it possible to estimate the effect of autonomous vehicles on energy efficiency of a single car as well as the traffic system as a whole, it is crucial to develope a baseline traffic simulation environment which represents real traffic situations. In this chapter, a SUMO simulation of the traffic on the Drive Me route, as explained in the previous section, is conducted. The simulation is done first using the default Krauss car following model in SUMO. Then, to match the behavior on the Drive Me route, the parameters of the IDM are calibrated using real driving data. To verify the simulation result, the flow and average speed of the vehicles in SUMO are measured by detectors on locations shown in Fig. 1.6, and compared to the real measurement provided by TrV.

1.4.1 Calibrating Krauss car following model Parameters

To verify the simulated traffic flow on the route against measurements done by TrV, first the default car following model in SUMO, Krauss, is used. The parameters of the car following model, given in Table 1.1, are tuned to give a flow as close as possible to the measured traffic flows.

Parameter	tau	$accel$	$decel$	$minGap$	$impatience$	$sigma$	$maxSpeed$
Value	1.45	2.78-3.25	4.5	2	0.5	0	100

Table 1.1: The parameters used in Krauss car following model.

The lane changing model used in the simulation is the default model in SUMO, i.e., LC2013. The parameter of the lane changing model, lcStrategic, is adjusted from its default value of 1 to 1.2 so that vehicles would make strategic lane changes earlier to improve the traffic flow, since this better replicates the actual driving behavior on the route. The speed factor parameter, which is multiplied with the speed limit on the road to allow for vehicles to drive faster than the speed limit, is set to 1.18.

Moreover, at some intersections, e.g. the one shown in Fig. 1.4, the vehicles braked harsh even when no intersecting traffic is passing. To prevent the excessive queue build up in these intersections, the visibility of the vehicles is increased by altering the $visibility\ parameter$ of the junction.

1.4.2 Calibrating IDM Parameters

To assure a good result, the parameters of the driver models should be calibrated so that the model reflects the driving style of real customers on the Drive Me route. In this section, the IDM is used as car following model since it is shown in (5) that it can represent real driving more accurately compared to the Krauss or Wiedemann car following models.

To find the best parameters of the IDM, 1'000 trips are chosen from a database of real recorded trips on the Drive Me route. The trips are randomly chosen but so that no driver assistant functions are activated, are longer than five minutes, are more than 99% of the time in car following mode, and include both high and low speed driving. The default value of 4 is used for the acceleration exponent $delta$ and the minimum gap measured on the 1000 trips, shown in 1.7, is fit by a gamma distribution and implemented in SUMO.

A genetic algorithm is used to find the optimal value of the other four parameters of the IDM, namely $accel$, $deccel$, $maxSpeed$, and tau, by minimizing an error that is proportional to the difference in energy demand and gap to the car in front between measurement and simulation. The distributions of the IDM optimal parameters for the trips are shown in Fig. 1.8. It can be seen that the desired acceleration has a mean value of $1.5\,\mathrm{ms}^{-2}$ and is lower compared to what is used to verify the traffic flow using the Krauss car following model. Desired deceleration has a wide distribution and

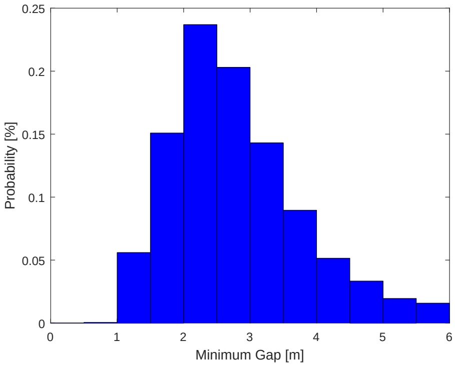


Figure 1.7: Distribution of minimum gap at stand still based on 1'000 trips.

is stretching from almost zero to 7 ms^{-2} where the maximum deceleration is capped. The desired speed has a mean value of 102.5 kph and the desired time gap has a log normal distribution with a mean value of 1.0 s.

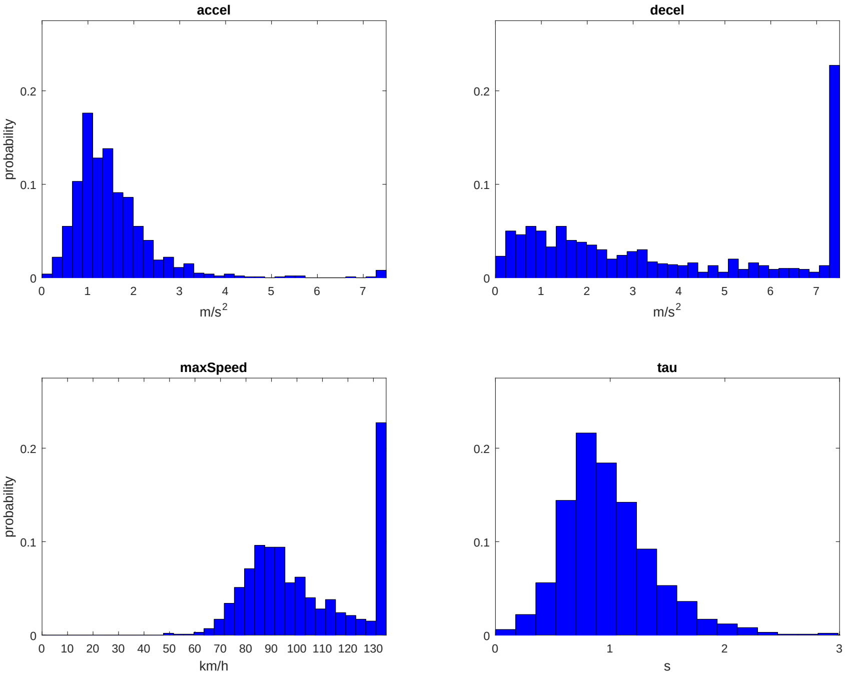


Figure 1.8: Distributions of the optimal parameters for IDM based on 1'000 recorded real trips.

The parameters are then sorted into 100 clusters, representing a unique driving style, i.e., $vehicle\ type\ id$ in SUMO, and values of the centers of the clusters are used as parameters for the IDM. The number of cars generated from each cluster are proportional to the size of the cluster and sets the $probability$. This would assure that the car following behavior has the same distribution as the real world and is representative for real drivers.

1.4.3 Traffic Flow verification

The simulation is run with a frequency of 10 hz with the parameter settings for Krauss and IDM as presented in the previous sections. Since the inflows are defined stochastically and traffic can momentarily be heavily congested, traffic flow is measured over one hour. In this report the traffic flow is averaged over 30 minutes, which is assumed to be long enough to have stable average measurements. Despite minor variation it is shown that real world traffic flow is replicated in the traffic simulation software SUMO.

The simulated flows in clockwise and counterclockwise directions, are shown in Fig. 1.9 and Fig. 1.10. Using the Krauss car following model, it can be seen that the simulated traffic flows are similar to the measured flows at the 15 detector stations along the Drive Me route. The simulated average speed from the simulation and measurement in clockwise and counterclockwise directions are also shown in Fig. 1.11 and Fig. 1.12. The figures show that the simulation achieves similar average speed as in the real traffic, which indicates that the model captures some basic driving behavior such as time gap to car in front, vehicle speed profiles, and stability of the whole traffic system. However, this does not guarantee that the driving style or energy efficiency of the vehicles matches reality.

When the IDM is used, however, no stable solution is found with the calibrated driver model parameters for IDM. Traffic flow and vehicle speed are significantly lower than the measured values on the route looking at an average of 30 minutes of driving as shown in 1.9 and 1.11 for the clockwise direction, and 1.10 and 1.12 for the counterclockwise direction. The problem persisted despite tuning the lane changing parameters. The main problem observed in this case is that the lane changes are not executed as smoothly as when using the Krauss car following model. One reason could be that in IDM, the acceleration and deceleration parameters are the desired values. The actual acceleration calculated at each time instant can exceed the desired parameters. However, in various other functions of the simulation the parameters are used as maximum and minimum acceleration. This resulted for example in higher desired net distances to the front car to assure a safe stop, which then led to hard braking in dense traffic situations, which in turn build queues in the system. Additionally it can be observed that when using the IDM, vehicles were much more hesitant to enter a junction without the right of way. This in turn led to queues up to insertion points to the network and therefore prevented continuous insertion of vehicles into the system.

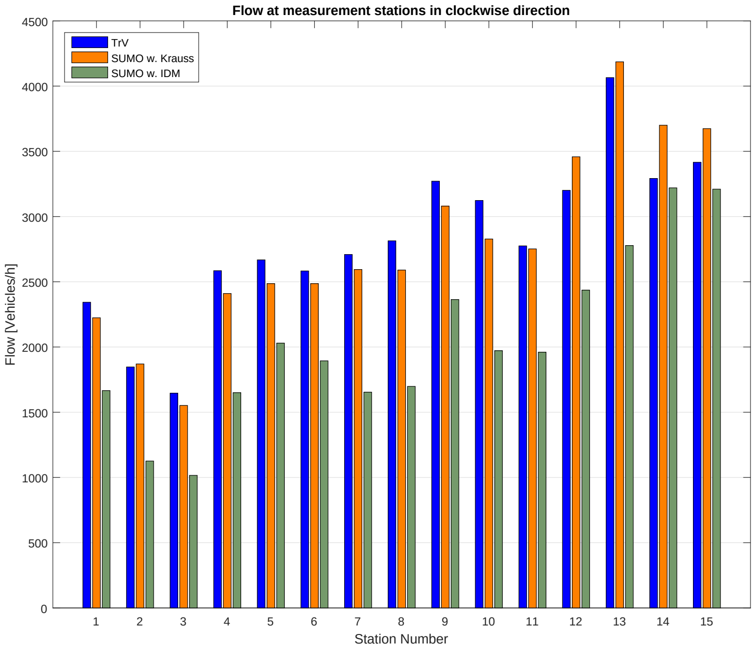


Figure 1.9: Traffic flow in clockwise direction measured in SUMO using the Krauss model and the IDM, versus measured values from real traffic on Drive Me route.

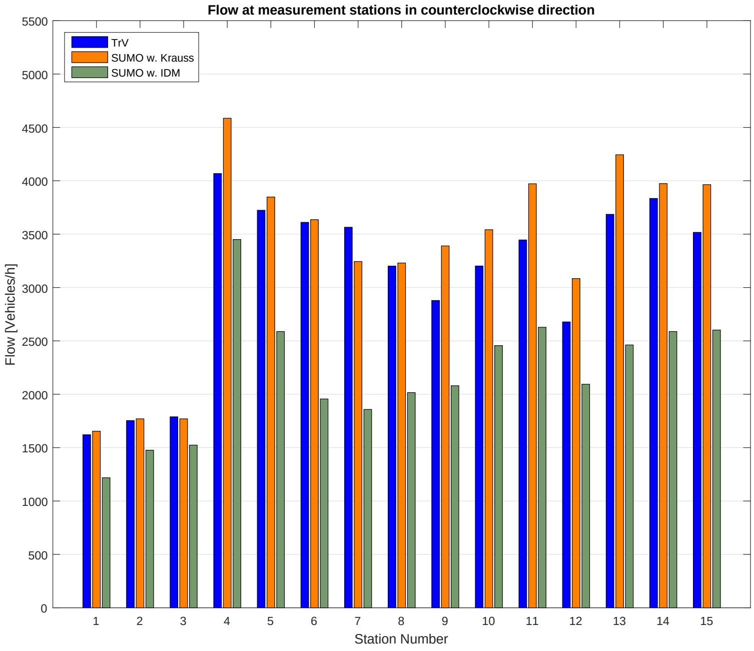


Figure 1.10: Traffic flow in counterclockwise direction measured in SUMO using the Krauss model and the IDM, versus measured values from real traffic on Drive Me route.

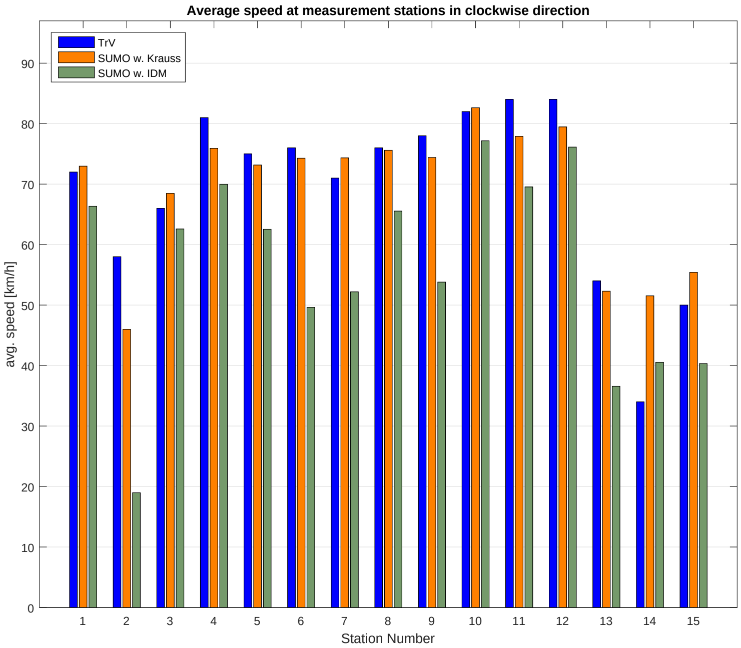


Figure 1.11: Average speed in clockwise direction measured in SUMO using the Krauss model and the IDM, versus measured values from real traffic on Drive Me route.

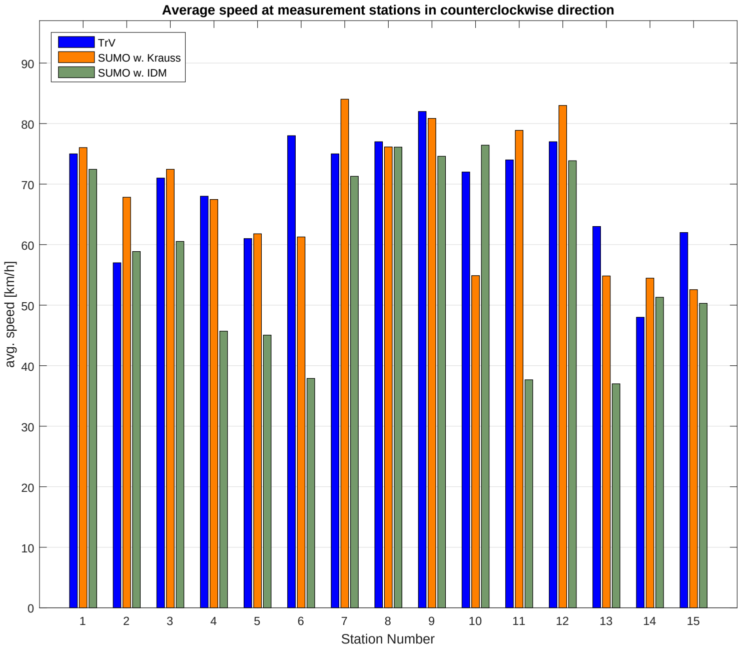


Figure 1.12: Average speed in counterclockwise direction measured in SUMO using the Krauss model and the IDM, versus measured values from real traffic on Drive Me route.

1.5 Conclusion

In this paper, we have simulated the traffic environment on the Drive Me route around Gothenburg, Sweden. In the first step, routes are imported to SUMO from OpenStreetMap, inflow and outflow of vehicles are defined, speed limits are set and bus lanes are added. The simulation is run first using the default Krauss driver model, with parameters manually tuned to generate realistic traffic flows. Then, in order to better represent the behavior of the drivers on the Drive Me route, the parameters of the IDM are calibrated using measured data from vehicles driving on this route.

To verify the simulation result, the traffic flow and vehicles' average speed were measured on 15 different positions along the route in both directions. It is was shown that SUMO generates results similar to real measurements when using the default Krauss car following model.

Simulating traffic with the calibrated IDM resulted in many conflicts in the simulation mainly during vehicles lane changing. The parameters of the lane changing algorithm were adjusted to achieve a more efficient lane changing behavior. However, some problems such as deadlocks, especially on roads with three to four lanes remained unsolved, which caused long queue build ups in the system and therefore affected the traffic flow and average speed. Additionally, it was observed that in dense traffic situations vehicles brake unreasonably hard when other vehicles merge in front. This had a cascade effects further back in the traffic system and could lead to heavy congestion.

In future work the lane change model needs to be adjusted in order to make more realistic lane changing possible when using the IDM. Furthermore, a model for autonomous vehicles needs to be added to the simulation environment so that the effect of autonomous vehicles on emissions and traffic on the Drive Me route can be evaluated. This could be done by replacing the standard car following models with the car following algorithm used by autonomous vehicles. By Using complete vehicle models, emissions and energy consumption can be evaluated and compared to real life fuel consumption obtained by the Drive Me pilot in Gothenburg, Sweden.

1.6 Acknowledgement

The Drive Me programme is partly funded by FFI - Strategic Vehicle Research and Innovation, a partnership between the Swedish government and automotive industry.

1.7 Bibliography

[1] *Simulation of Urban MObility - Wiki*. http://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-_Wiki, 2017.

[2] Goodwin, Lynette C: *Weather impacts on arterial traffic flow*. Technical Report, Federal Highway Administration, Washington, prepared for Road Weather Management Program, 2002.

[3] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Bieker Laura: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3-4):128–138, 2012.

[4] Krauss, Stefan: *Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics*. D L R - Forschungsberichte, (8), 1998.

[5] Mitra Pourabdollah, Eric Bjarkvik, Florian Fuerer, Bjorn Lindenberg, Klaas Burgdorf: *Optimization and Assessment of Car Following Models Using Measured Data*. Manuscript submitted for publication, 2017.

[6] Tettamanti, Tamás, István Varga and Zsolt Szalay: *Impacts of Autonomous Cars from a Traffic Engineering Perspective*. Periodica Polytechnica Transportation Engineering, 44(4):244–250, 2016.

[7] Treiber, Martin, Ansgar Hennecke and Dirk Helbing: *Congested Traffic States in Empirical Observations and Microscopic Simulations*. Physical Review, 62(2):1805–1824, 2000.

[8] Trent Victor, Marcus Rothoff, Erik Coelingh, Anders Ödblom and Klaas Burgdorf: *When Autonomous Vehicles Are Introduced on a Larger Scale in the Road Transport System: The DriveMe Project*. In *Automated Driving. Springer International Publishing*, volume 63, pages 541–546. 2017.

[9] Wadud, Zia, Don MacKenzie and Paul Leiby: *Help or hindrance? The travel, energy and carbon impacts of highly automated vehicles*. Transportation Research Part A: Policy and Practice, 86:1–18, 2016.

[10] Wu, Changxu, Guozhen Zhao and Bo Ou: *A fuel economy optimization system with applications in vehicles with human drivers and autonomous vehicles*. Transportation Research Part D: Transport and Environment, 16(7):515–524, 2011.

2 Modelling and calibrating situation adaptive lane changing and merging behavior on Chinese elevated roads

Marc Semrau[1], Jakob Erdmann[2], Jens Rieken[4] and Bernhard Friedrich[3];
[1]Volkswagen AG, Berliner Ring 2, 38440 Wolfsburg, Germany
[2]Institute of Transportation Systems, German Aerospace Center
[3]Institute of Transportation and Urban Engineering, 38108 Braunschweig, Germany
[4]Institute of Control Engineering, 38106 Braunschweig, Germany

Marc.Semrau@volkswagen.de

2.1 Abstract

This paper describes the extension and the calibration of a driver model in SUMO that enables a situation adaptive driver behavior for lane changing and merging processes. The extensions are tailored to improve the representation of Chinese traffic dynamics but are expected to yield improvements for other regions of the world as well. To design and calibrate the model extensions FOT (field operational test) Data is used to analyze which influence critical traffic situations have on driving behavior. The results are discussed and findings for driver models, as well as for piloted driving systems are derived. To this end a detailed evaluation of merging and lane change behavior, with respect to different traffic situations, is conducted. Subsequently the model extensions of *emotion-memory* and *lateral encroachment* are explained and it's implementation and calibration discussed. Finally the resulting driver model is validated by using real world data from Chinese elevated road interchanges and merging areas.

Keywords: Heterogeneous traffic, Driver Model, Testing ADAS

2.2 Introduction

2.2.1 Motivation

A new field of application for traffic simulation and driver modeling is continuously growing in importance, namely testing and developing driving functions. Currently a lot of real driving tests are carried out, when testing or improving assisted or piloted driving functions. Their precision is so far not reached with simulations. Nevertheless, with rising automation level of driving functions they become more complex and the testing process becomes not only more and more expensive but is also extremely time consuming. Consequently, time restrictions often render comprehensive real world tests cumbersome, which is why simulations are an indispensable part of the overall testing process. Semrau et. al. (15) introduced a framework for this testing process, described some extensions of driver models and the first calibration methods for lateral dynamics in (15). Based on findings described in (15) further research was necessary in several aspects. With testing driving functions in a traffic simulation we choose a behavior based approach, meaning that the driver characteristics

have a significant influence on the simulation. The behavior of human drivers is not constant but changing due to various conditions, like interactions with other drivers and surroundings. This is especially true for complex merging situations at highway intersections, since a lot of interactions happen here.
Section 2.3 describes the generation of the data basis for this research, as well as the processing of the new Velodyne lidar data. In section 2.4 the lane changing an merging behavior is analyzed. It starts with lane changes for merging and the concept of emotional memory and goes on with a detailed analysis of pushing behavior and virtual lane formation. After the implementation is described, section 2.5 presents the validation concept. Section 2.6 concludes this paper by discussing open points and limitations of the developed models.

2.2.2 Related work

The importance of driver emotions for the driving style is well known in the research community. Nevertheless there are not many findings about the concrete influence on driver to driver interactions. Cai et al. used a multiple simulator system with three human drivers to include emotion into their study in (1). This enabled them to display high fidelity driver to driver interactions. They decided to replace simulated cars with human drivers due to the importance of emotions and the difficulties to measure this short term effect.
Kraus went one step further and analyzed the OCC (Ortony, Clore and Collins) emotion model (10) and selected the emotions with influences on traffic in (8). In his work he modeled automated, situation adaptive lane changes and validated them in a study. To this end the participants were asked to evaluate the automated lane changes. Two big differences to our work are the use of an automated system and the questioning of the participants concerning the emotions. We will prove the existence of emotions without questioning the participants and with naturalistic driving data in section 2.4.2.
Leu et. al. also analyzed and described different emotion models and decided to use the five factors model model (5) in (9). They used a classic car following and lane changing model and modified it with a short term emotional factor. In contrast to the research presented here, Leu et. al. only proved the influence of emotions by comparing macroscopic parameters like travel times. Therefore they only calibrated the influence on a macroscopic level, but not on the car to car interaction level.
The fka ("‘Forschungsgesellschaft Kraftfahrwesen mbH Aachen"’) discussed emotions with regard to lane change motivations in their whitepaper (4). Without doubt this is one important point of emotions. The influences discussed are similar to the memory for lane changing motivations used in SUMO. However this work is dealing with the influence of emotions on the viability of lane changes, which is not discussed in (4).

2.3 Driving studies (Data recording)

The data source used here consists of three consecutive studies, all executed in Shanghai. The first two studies have been described by the author in (15). Their main purpose was to point out critical situations, typical for Chinese traffic, which could benefit from driver assistance systems. Moreover driver behavior was recorded during those situations and analyzed afterwards. Consequently this studies deliver the data used in section 2.4 to model lane changing behavior. Therefore a representative driver distribution was used here, so the behavior of the ego vehicle was used for analysis and calibration of the simulation. The two studies have a major limitation in sensor view, there is no object data alongside the ego vehicle.
To close this gap a consecutive study was planed. The focus were traffic jam situations on Chinese elevated roads, especially merging situation at motorway intersections. The Audi A7 test car was

equipped with an additional Velodyne lidar on the rooftop. This makes object detection all around the vehicle possible.

2.3.1 ADAS Study

The ADAS study is a large-scaled study with the aim to investigate existing driver assistant systems (DAS) on the Chinese Market, respectively in Chinese traffic. As already described in (14; 15) the focus is the evaluation of availability and usability of DAS (including the recording of sensor data) as well as the subjective opinions of the participants. Therefore, it is possible to derive situations in Chinese traffic which cannot be handled by state-of-the-art DAS but assistance was demanded by the test persons. To get representative results the study was carried out with 40 Chinese drivers that where selected according to the typical Chinese driver distribution, concerning age, driving experience and gender. The surveys in this study took part in several phases of the test. The drivers were interviewed before, during and after the driving tests. Additionally, sensor data and data of the vehicle were recorded to make the analysis of the occurring situations possible. Especially congested situations are more common in China than in Germany and the complexity in congested situations is much higher, too. Details on this point will be shown in section 2.4.

2.3.2 Traffic jam study

The two previous studies had a major limitation in sensor view, there is no object data besides the vehicle. To close this gap a consecutive study was planed. The focus were traffic jam situations on Chinese elevated roads, especially merging situation at motorway intersections. The Audi A7 test car was equipped with an additional Velodyne lidar on the rooftop. This makes object detection all around the vehicle possible. The enhancement of sensor view is shown in figure 2.1. The solid areas visualize the sensor view during the first two studies, area with diamonds represents the extension with the Velodyne lidar. The following paragraphs describe the details of the study, including route, sensor data processing and analysis. The focus were traffic jam situations on Chinese elevated roads, especially merging situation at motorway intersections. The driven route is shown in figure 2.2. It includes 1420 kilometers of Shanghai Inner City Elevated roads. For the scope of this paper we will mainly use the motorway intersection, including the marked access roads. For validation purpose traffic flow data for the whole area is available.

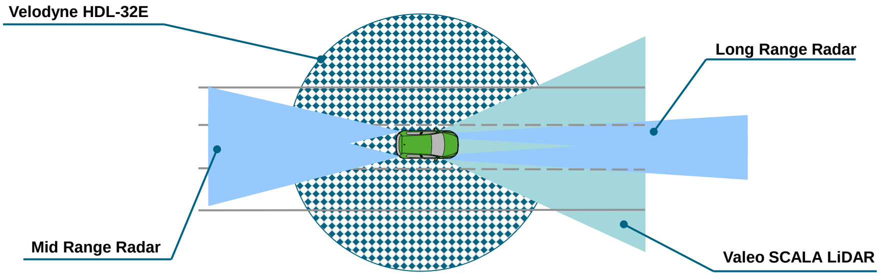


Figure 2.1: Sensorview

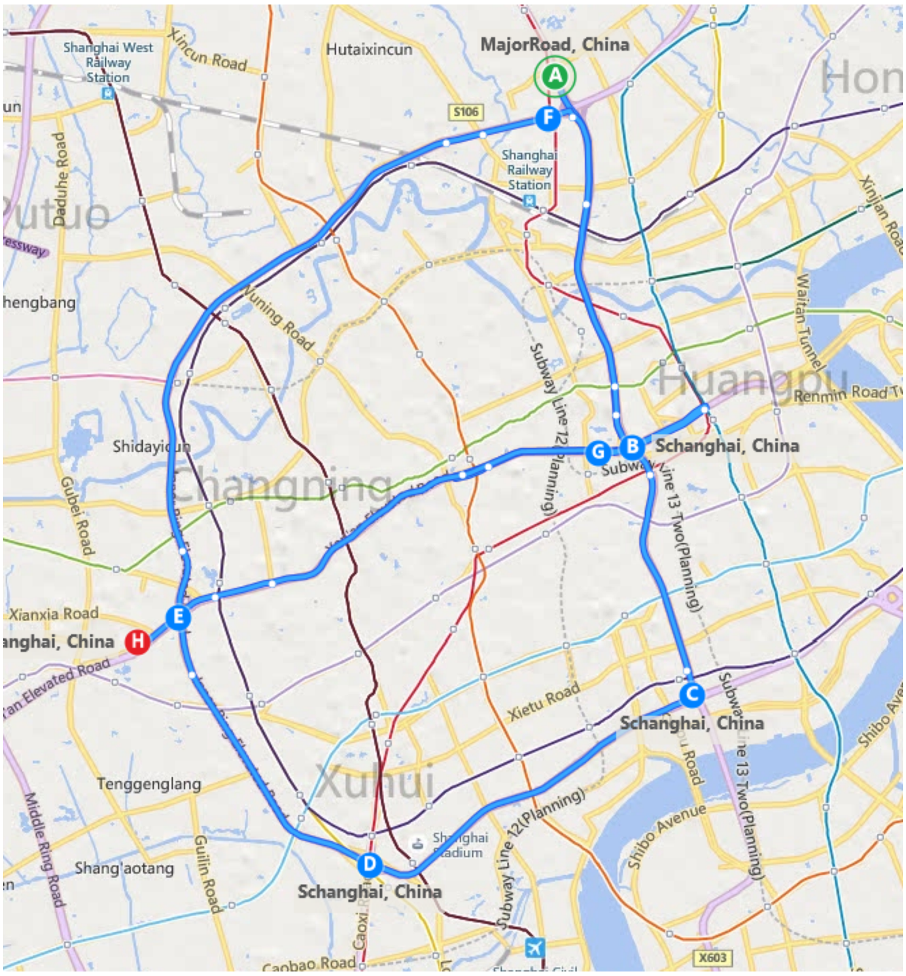


Figure 2.2: driven route on bing maps

2.3.3 Velodyne raw data processing

The environment perception and modeling is accomplished using a Velodyne HDL-32 Lidar sensor. The sensor is mounted on the vehicle's roof. It is thus able to provide a 360 degree view around the ego vehicle, capturing the road surface as well as elevated elements.

For proper processing, point measurements of each full rotation are collected into a single scan. As measurements of the road surface itself are irrelevant to the object tracking algorithm, they are classified and excluded from further processing. This is done by a grid-based algorithm (cf. (12), see results in Figure 2.3a). This approach also provides an estimation of the ground surface, which allows for a ground-relative height estimation of elevated elements. The remaining measurements are considered as originated from elevated targets. They are converted into a 2.5D Stixel representation by projecting them into a polar grid and grouping them based on their polar position. Stixels are then clustered into segments by making use of their spacial distance properties, which result from the channel-based acquisition principle of the sensor.

In order to separate moving from non-moving elements, a consistency-based grid model is utilized to classify clusters based on their spatio-temporal consistency. The classification results are shown in Figure 2.3b. A box fitting algorithm is then applied for moving and potentially movable clusters. This algorithm classifies the shape of the outer contour and provides a bounding box estimation for each cluster. A more detailed description of the processing algorithms is given in (13).

An Extended Kalman Filter-based approach is finally applied to perform the tracking task. Each object hypothesis is represented by a separate filter instance. Based on a constant-acceleration model, dynamic states are estimated from subsequent associated box hypotheses, along with object's dimensions and most probable class. During the tracking process, timing effects resulting from the scanning principle of the sensor are compensated, as described in (11). As a result, the tracking module provides a time-consistent representation of all movable object hypotheses around the ego vehicle, as shown in Figure 2.3c.

2.3.4 Object fusion

Mostly the plausibility of the objects recognized by the Velodyne lidar is higher than the one of the integrated vehicle sensor objects, delivered by a separate object detection. Nevertheless we decided to use an object fusion on object level to get the best possible results. This is particularly useful for the detection of stationary objects. The object fusion is kept as simple as possible, since it is not the main focus of this research. The developed algorithm checks whether objects overlap and if necessary which sensor delivers to most reliable data. Subsequently a merged object is generated, based on the best available sensor data.

2.4 Lane changing and merging behavior

2.4.1 Lane changes for merging

The lane changing model in SUMO has been described in (2). It distinguishes a hierarchy of change motivations with *strategic* lane changing as the most important reasons. A lane change is called strategic, when it is necessary for following the route. Less urgent reasons include overtaking slower vehicles or helping other drivers. Lane changes by drivers merging onto a motorway from an on-ramp are strategic since the on-ramp typically ends at some point and does not permit further driving without changing.

In a fashion, typical for other microscopic traffic simulations, SUMO re-uses the car-following model for determining the viability of lane-changing to ensure safe driving. This is done by evaluating the car-following model for the hypothetical situation that would result from a lane change for the follower vehicle on the target lane as well as for the ego vehicle with respect to the leader on the target lane. The influence between lateral dynamics and longitudinal dynamics runs in both ways however. Not only does the car-following model put restrictions on lane-changing, the lane-changing model may pose additional constraints at vehicle speeds as well. In SUMO this is accomplished by determining speed changes that aim to create an acceptable gap for lane changing. In the context of strategic changes, cooperative speed adaptions by surrounding traffic may also be triggered.

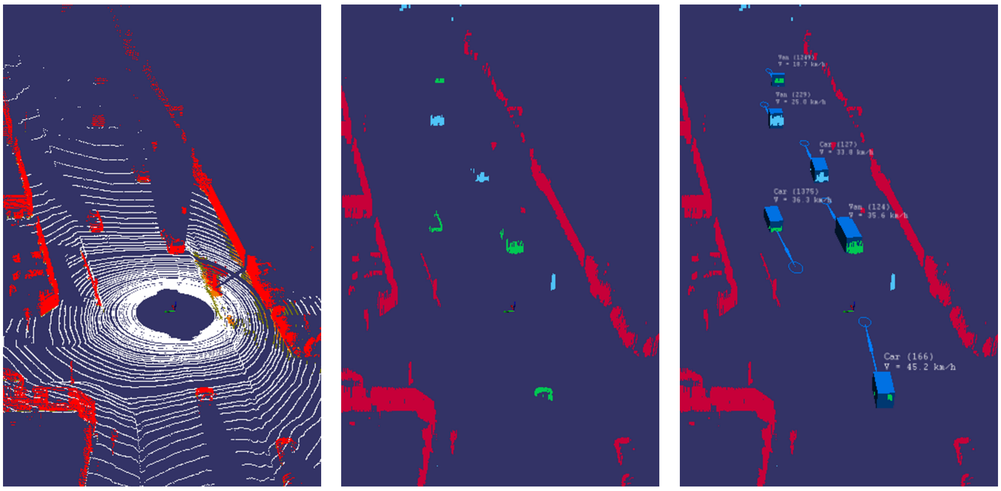
The car-following model allows the calibration of acceleration and deceleration limits and the desired time headway as well as minimum physical gap.

In SUMO these parameters are attributes of each vehicle. To meet the characteristic driver distribution in Chinese traffic which comprises aggressive, moderate and conservative drivers, a representative distribution of parameters was used for calibration.

The overall characteristics of lane changes have been described several times, for example in (2; 6; 15; 17) and will only be summarized here. In general the previous data analysis in (15) shows that the lane changing behavior strongly depends on the traffic conditions and on the level of driving experience. The difference between novice and expert driver increases with rising complexity of the traffic situations. As a result it is necessary to distinguish between driver types and traffic conditions. The general assumption under all circumstances is safe driving, as described in (7).

2.4.2 Emotional Memory

New research, presented in (15) proved that the assumption of safe driving is not true for all situations. Drivers undercut the safe gaps in several situations. The main focus of this paper is the change in



(a) Ground points (b) Stixel dynamics (c) Object hypotheses

Figure 2.3: Processing steps of the Velodyne processing algorithms. Fig. 2.3a shows the result of the ground classification algorithm. Ground points are colored in white, elevated targets in red. Fig. 2.3b illustrates Stixel generation and dynamic classification. Stationary clusters are colored in red, stable moving clusters are colored in green. Potentially movable clusters are blue. The final object hypotheses are shown in Fig. 2.3c.

behaviors of drivers due to emotions. To this end we analyzed the studies described before regarding critical traffic situations and driver's impatience. The main point was to prove the dependency of drivers aggressiveness and emotions caused by surrounding traffic. This assumption is supported by Kraus, who analyzed different emotions in traffic in (8).

The recorded situations suggest a continuous change between the focus of the two motivations: Reaching the destination safe against reaching it fast. Which is why we focus on these emotions here. We were able to empirically prove the dependency of these emotions from traffic situations experienced by the driver. To this end we compared characteristic values for the viability of lane changes shortly after the driver experienced critical situations with his average behavior. We define the *safety coefficient* as:

$$s = \frac{d}{d_{safe}} \tag{2.1}$$

With d being the actual gap and d_{safe} representing the minimal safe braking gap. The drivers reduce the safe gap, which was found to be important for evaluating the risk (cf. (15)). The data analysis showed, that the average safe coefficient is reduced by 24%, in case the driver experienced a critical situation during the last five seconds. We where only able to prove this effect for this small amount of time. This suggests that this emotion is rising instantaneously, but not lasting long. With the existing data we where not able to prove a permanent effect on the drivers behavior. This outcome is supported by the findings in (1) and (9), Consequently a well-founded statement on this point requires more research.

None of the so far used models considers an emotional memory to take account of this effect. Thus the impatience factor I is used to manipulate the safety criterion for lane changes with regard to the emotional memory. The implementation of this factor enables us to influence the lane changing behavior separately from the car following behavior. To this end we detect critical distances, including critical cut in situations, and increase the impatience factor by 24%. After five seconds the impatiences is decreased to the original value of the driver. Moreover we assume a dependency of aggressiveness when executing pushing maneuvers and the drivers emotions. Consequently we analyzed pushing maneuvers and their impact on aggressiveness. Results are presented in the next section.

2.4.3 Pushing and virtual lane formation for merging

The second key aspect of this paper is the pushing behavior in Chinese traffic. We observe a behavior that we call *pushing* which leads to the formation of an additional temporary lane. The behavior is exhibited by some drivers when they are unable to find a safe gap for merging from a motorway on-ramp into the main flow. Instead of waiting for a gap to form, they encroach laterally on the vehicles in the main flow, thus *pushing* them aside. Another term to describe this effect is *forced cooperative lane changing* since the pushed vehicle cooperates with the merging vehicle rather than insisting on its right-of-way and risking an accident. This leads to a lateral compression of the road lanes and allows a virtual lane to form on which the merging vehicle may drive. The traffic jam study gives us the unique possibility to observe pushing behavior from surrounding traffic. This makes modeling of the complicated forced cooperative lane changes, that have been introduced in (15), possible. As we stated in (15) the principle of forced cooperation is working due to the lateral safety distance the vehicles try to keep and the so called *pushy* factor. The findings for emotional changes in 2.4.2 suggest that the level of aggressiveness is not constant but changing due to the experiences a driver made during his travel. The more aggressive the driver type is, the lower his pushy threshold will be. This results in an earlier pushing behavior with the resulting small lateral distances. The next section investigates if this is true for the pushing behavior as well. There are three possible conditions for pushing situations, giving the driver the choice between the given patters:

1. There is no blocking car and the vehicle can move away easily
   a) The driver could brake
   b) The driver could ignore the pushing car
   c) The driver could start moving lateral
2. There is a blocking car and the vehicle cannot move easily
   a) The driver could brake
   b) The driver could ignore the pushing car
   c) The driver could start pushing himself to move lateral
3. There is a static obstacle and it is not possible to move away lateral
   a) The driver could brake
   b) The driver could ignore the pushing car

The first condition and the resulting patterns have already been described in (15). Consequently the focus is on the second and the third condition. In case of the condition 2 the driver can react in three possible ways. All of them are shown in figure 2.4. In the first row one can identify a braking reaction 2a. Here the ego vehicle is pushed from the right (shown in figure 2.4a) and is braking to avoid a collision. The resulting cut in (shown in figure 2.4b) is particular narrow, but the describing parameters and emotions leading to this behavior are considered in section 2.4.2 and won't be analyzed in this section. Nonetheless the pushing behavior before the cut in can be observed. In contrast pattern 2b does not result in a cut in, since the pushing vehicle is ignored. In row two in figure 2.4c a fast vehicle is approaching on the right side. In figure 2.4d the same situation is displayed a little later. Obviously the approaching car was forced to brake, since the lane ended and the pushing was not successful. One important factor for this pattern and the success of a pushing maneuver is the longitudinal position of the pushing vehicle. This correlation will be analyzed in detail later in this section. The last row in figure 2.4 gives an example for the pattern 2c. The ego vehicle is pushed from the right side and tries to avoid a collision by pushing into the left lane (shown in figure 2.4e). After the pushing situation is over the ego vehicle is driving back into its original lane. This can be seen based on the driving path prediction (visualized by the red lines in figure 2.4e), which indicates a right turn. This kind of interaction will also be analyzed in the following, since it gives insights into the minimum lateral distance required by the drivers.

Under condition 3 the possibilities are limited, since it is not possible to push a solid object without a collision. One can easily identify the possible patterns 3a in row one of figure 2.5 and pattern 3b in row two of picture 2.5. Again the braking reaction results in a regular cut in, as shown in 2.5b. The standard lane change parameters have already been considered in the merging behavior in section 2.4.2. Nevertheless the characteristics distances of this pattern can help modeling the virtual lane formation. The second pattern 3b is at least as important for the analysis of pushing behavior. As already described for the second condition 2 we can learn the characteristic lateral safety distance, as well as the influence of the longitudinal position of the pushing vehicle from those situations. In the shown situation (cf. figure 2.5c) the taxi on the right tries to push into the lane of the ego vehicle. The next figure 2.5d displays the reaction, the pushing vehicle is ignored. Those kind of situations will be analyzed to find parameters characterizing those desicions.

We did not investigated all possibilities, Table 2.1 displays the distribution of patterns. Obviously 1 was not monitored at all. This leads us to the conclusion, that pushing and the resulting virtual lane formation is only occurring in heavy traffic. This seems logical, since there is no need for virtual lanes if the physical ones are still empty.

Table 2.1: distribution of reaction patterns

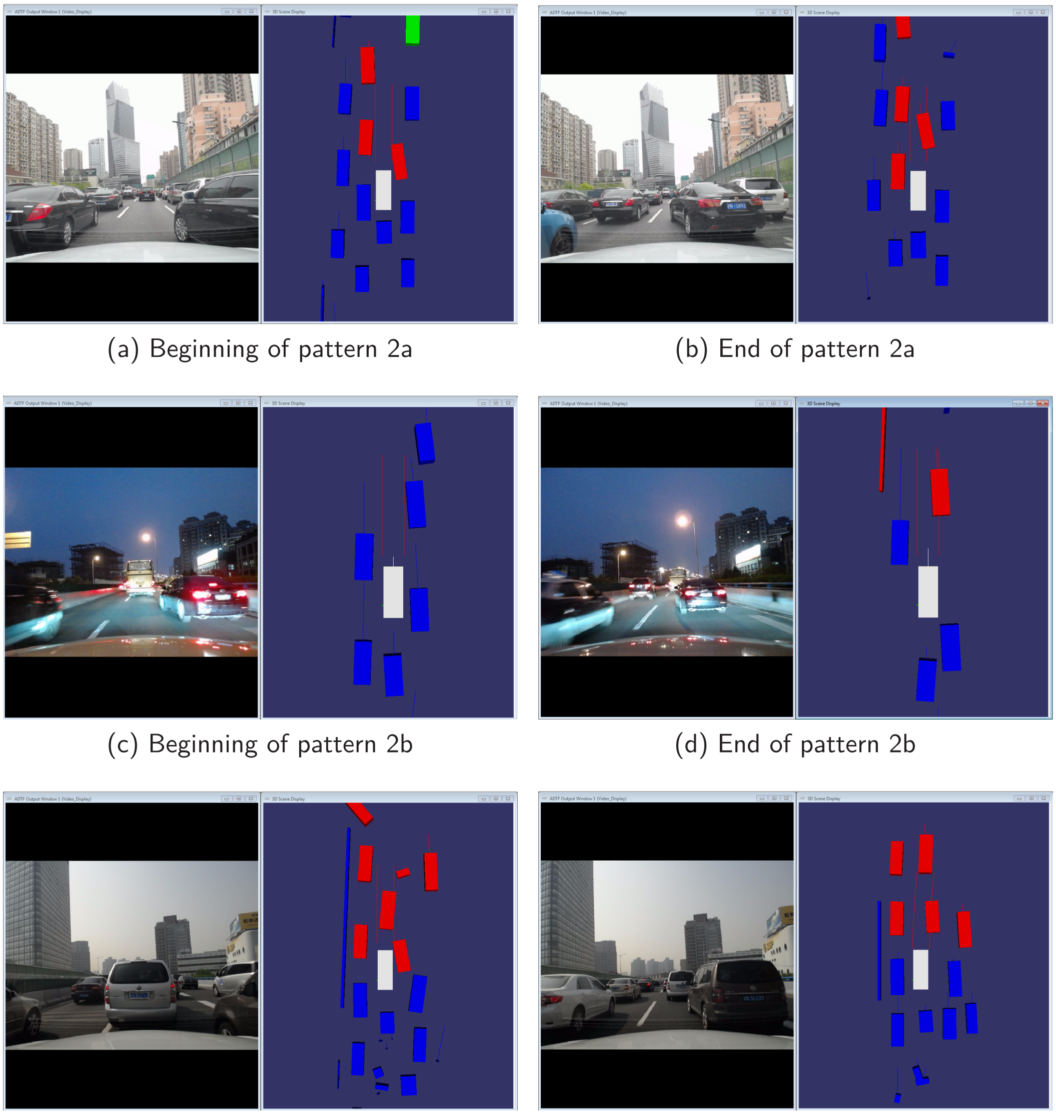
1a	1b	1c	2a	2b	2c	3a	3b
0%	0%	0%	57%	11%	16%	13%	3%

Moreover, we can see that 70% of all analyzed pushing maneuvers result in a regular cut in (patterns 1a, 2a and 3a, meaning no virtual lanes are formed. However we were able to gain knowledge out of those maneuvers, too.

Table 2.2 shows the longitudinal heading of the pushing vehicle for a resulting cut in in comparison to the situations not resulting in a cut in. What we can conclude out of this is the importance of the longitudinal position of the pushing vehicle. In average the heading while pushing for cut in situations (braking) is more than three times the heading for pushing situations resulting in a virtual lane formation (moving lateral). For the pattern ignoring the heading is even less. So the importance of this parameter is obvious for the resulting behavior. Of course the behavior differs between drivers and interviews with Chinese drivers implicate, that the reactions 2b and 3b, i.e. to ignore the pushing car, are more likely to happen, in case the driver experienced a similar reaction himself before. The same is true for reaction 2c. Nevertheless we where not able to find a significant correlation in the evaluated situations. This does not prove the assumption wrong, we are just not able to prove it with the used database.

Table 2.2: characteristic values for the analyzed patterns

		pattern		
		braking (a)	ignoring (b)	moving lateral (c)
$\Delta x_{\mathsf{min}}[m]$	mean	3,082	0,101	0,813
	standard deviation	1,892	1,814	2,004
$\Delta y_{\mathsf{min}}[m]$	mean	0,274	0,076	0,235
	standard deviation	0,189	0,192	0,262



(a) Beginning of pattern 2a

(b) End of pattern 2a

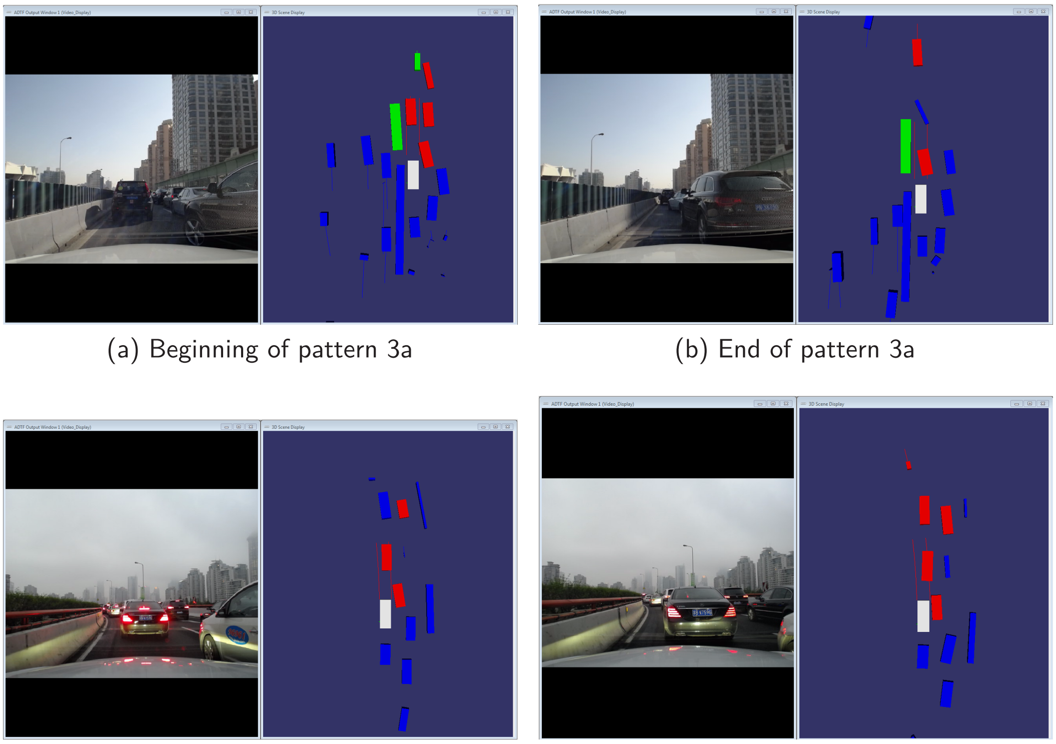
(c) Beginning of pattern 2b

(d) End of pattern 2b

(e) Beginning of pattern 2c

(f) End of pattern 2c

Figure 2.4: Possible reaction patterns in case of heavy traffic (condition 2). The ego vehicle is shown in white. The red boxes are obstacles detected by the Velodyne and integrated vehicle sensors while the red lines represent the prediction of the vehicle path. The blue boxes are Velodyne objects and the green ones are objects only detected by integrated vehicle sensors. The small lines starting in the center of the boxes are speed indicators.


ADTF Output Window 1 (Video_Display)
3D Scene Display
ADTF Output Window 1 (Video_Display)
3D Scene Display
ADTF Output Window 1 (Video_Display)
3D Scene Display
ADTF Output Window 1 (Video_Display)
3D Scene Display

(a) Beginning of pattern 3a

(b) End of pattern 3a

(c) Beginning of pattern 3b

(d) End of pattern 3b

Figure 2.5: Possible reaction patterns in case of a solid obstacle (condition 3). The ego vehicle is shown in white. The red boxes are obstacles detected by the Velodyne and integrated vehicle sensors while the red lines represent the prediction of the vehicle path. The blue boxes are Velodyne objects and the green ones are objects only detected by integrated vehicle sensors. The small lines starting in the center of the boxes are speed indicators.

Reactions 2b and 3b make up 14% and could result in a critical situation, since the distances are small. For those patterns is the average minimal lateral distance between vehicles is $\Delta y_{\text{min}} = 7.6cm$ (cf. table 2.2). Even though the standard deviation is high this is a strong hint for the criticality of those situations. The distances give detailed insights into the interaction between drivers and support modeling drivers in that way.
Pattern 2c makes up 16% and here we can get insights into the interaction principle for successful virtual lane formations. If the pushing maneuver is successful it causes vehicles to drive parallel instead of causing strong braking reactions to avoid a collision. However, the result depends on the behavior of the vehicle the driver tries to push. The possible conditions and patterns remain the same. Not to repeat the already explained cases we only consider the situations, when the pushed car can move away. Table 2.2 displays typical minimal lateral distances, as well as the standard deviation. What tracks attention is the higher value of the average of the lateral distance here. One could conclude that this value is the true lateral safety distance and the very small values for cases 2b and 3b result from a higher level of aggressiveness or impatience.
As soon as possible vehicles start merging into their original lanes and the original value of the lateral orientation is used again. This can already be seen in figure 2.4f, when the ego vehicle is driving back into its original lane. This procedure is characterized by the car following and lane changing parameters. Depending on traffic conditions, driver type and emotions the situation is solved as soon as possible, meaning as soon as enough space is available. A similar strategy is used in VISSIM (3), where the traffic density is monitored and the lateral resolution is set back to normal lanes, as soon as driving on virtual lanes is not necessary anymore.

2.4.4 Implementation

To qualitatively recreate the observed behavior in the SUMO simulator, a model for lateral distance keeping was added. The desired lateral distance h between an ego vehicle and another vehicle that overlap along their length is defined as:

$$h = m \cdot r \cdot \min(1, \frac{\max(v, |u - v|)}{v_l}) \tag{2.2}$$

This value grows with the involved speeds of the vehicles: Either the speed of the ego vehicle v or its difference to the speed of the other vehicle u up to a limit of v_l where it is assumed to grow no further. The gap depends on a calibration factor m and a situational coefficient r described below. The coefficient r is defined as $1 - p$ when in the context of a strategically urgent lane change and 1 otherwise. In this case m denotes the desired gap at velocities of v_l. The value of m can be set using the driver parameter *minGapLat*. The value of v_l is currently set to $50km/h$. The value of p can be set using the driver parameter *lcPushy*. Its significance is described below. During each simulation step, vehicles perform lateral movements to ensure that a gap of h is maintained to all surrounding vehicles. In case there is insufficient lateral space to fulfill requirements on both sides of the vehicle, the desired lateral position is computed by dividing the available physical space in proportion to h_{left} and h_{right}, the minimum value of h on the left and right sides respectively. To distinguish the degree to which vehicles engage in *pushing* behavior the driver parameter p (*lcPushy*) is used. For $p = 0$, drivers only undertake lane change maneuvers where the destination may be reached without interfering with surrounding vehicles. Otherwise, the lane changing is undertaken to within the lateral gap h of that driver. If this gap undercuts the value of another vehicle, it will move laterally in the same direction in subsequent simulation steps to maintain its gap, thus being *pushed*. Due to the definition of the situational coefficient r, this leads to the creation of virtual lanes whenever the following conditions are met:

- a vehicle cannot continue on it's current lane

- the necessary lane change is blocked by neighboring traffic
- the neighboring traffic has sufficient space on the opposite side
- the p value of the vehicle is high enough to displace the neighbors and open enough space to allow changing to the target lane

2.5 Validation concept

The validation concept is based on single vehicle and traffic flow data.
We observed a highway intersection shown in figure 2.6. To do so we modeled the inflow and outflow as shown and measured the resulting traffic flow on the highway at the marked induction loops. To meet the characteristic driver distribution in Chinese traffic which comprises aggressive, moderate and conservative drivers, a representative distribution of parameters was used for calibration. Practically this was done with a script to generate a certain vehicle distribution with different attributes. This was done for the three different kinds of drivers. This guarantees a realistic number of vehicles in the observed area. This is important, since one major factor for all types of lane changes is the traffic density (16). We compared the parameters of the occurring pushing and merging situations with the values described in 2.4. Due to that we proved the validity of the car to car interactions in pushing and merging situations.

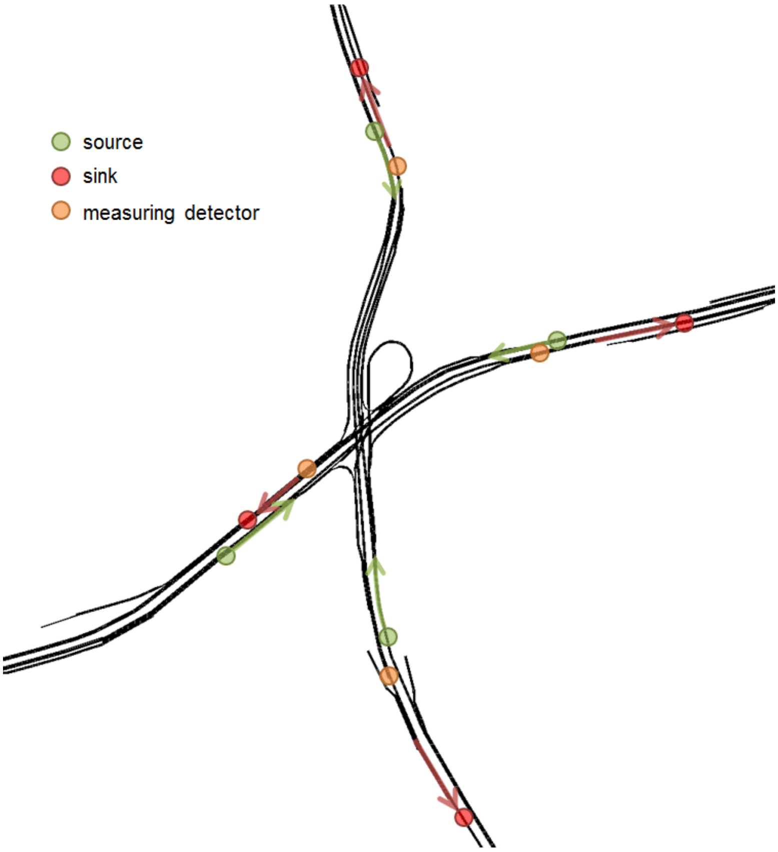


Figure 2.6: Observed area with detector positions. This figure shows the area on the Inner Ring in Shanghai, simulated and observed for validation purpose. It also displays the sources and sinks of cars, as well as the measurement positions.

Moreover the validation concept enables us to compare measurement data of real traffic flow and measurement data of the simulated traffic flow. This macroscopic validation is still subject to ongoing research at the moment, which is why it won't be described here.

2.6 Conclusion and future work

The previous sections have shown that it is possible to prove the influence of emotions in naturalistic driving data. Moreover we introduced an implementation for the emotional memory and the concept of pushing behavior for virtual lane formation.

The analysis of lane changing and merging behavior has been discussed in section 2.4. In this context the implementation of the emotional memory and the virtual lane formation has been described. The emotional memory enables us to integrate one more important factor of human behavior in our simulation tool. The analysis of the process of virtual lane formation is one important step to the simulation of heterogeneous traffic, not only in Shanghai. The validation concept, presented in section 2.5, combines the validation on microscopic and macroscopic level.

Future work will focus on the improvement of the emotional memory, especially the improvement of models for the long term effect. Moreover the effects of the emotional memory, the pushing behavior and the virtual lane formation will be validated as a whole. For this purpose the validation study, especially traffic flow data, will be used to complete the validation on the microscopic and macroscopic level.

2.7 Bibliography

[1] Cai, H, Y Lin and Ronald R Mourant: *Study on driver emotion in driver-vehicle-environment systems using multiple networked driving simulators*. In *Proceedings of the Driving Simulation Conference North America (DSC 07)*, 2007.

[2] Erdmann, Jakob: *SUMO's Lane-Changing Model*. In *Modeling Mobility with Open Data*, pages 105–123. Springer, 2015.

[3] Fellendorf, Martin and Peter Vortisch: *Microscopic traffic flow simulator VISSIM*. In *Fundamentals of Traffic Simulation*, pages 63–93. Springer, 2010.

[4] Forschungsgesellschaft Kraftfahrwesen mbH:: *PELOPS White Paper*, 2010.

[5] Goldberg, Lewis R: *An alternative" description of personality": the big-five factor structure.* Journal of personality and social psychology, 59(6):1216, 1990.

[6] Kesting, Arne: *Microscopic modeling of human and automated driving: Towards traffic-adaptive cruise control*. 2007.

[7] Krajzewicz, Daniel, Reinhart Kühne and Peter Wagner: *A Car Drivers Cognition Model*. In *ITS Safety and Security Conference*, volume CD, 2004.

[8] Kraus, Sven Bernd: *Fahrverhaltensanalyse zur Parametrierung situationsadaptiver Fahrzeugführungssysteme*. PhD thesis, Universität München, 2012.

[9] Leu, George, Neville J Curtis and Hussein A Abbass: *Modeling and Simulation of Artificial Drivers with Personality and Emotions for Effective Planning in Road Traffic Networks*.

[10] Ortony, A, G Clore and A Collins: *The Cognitive Structure of Emotions: Cambridge Uni*. Press, New York, 1988.

[11] Rieken, J. and M. Maurer: *Sensor scan timing compensation in environment models for automated road vehicles*. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 635–642, Rio de Janeiro, Brazil, Nov 2016. IEEE.

[12] Rieken, Jens, Richard Matthaei and Markus Maurer: *Benefits of Using Explicit Ground-Plane Information for Grid-based Urban Environment Modeling*. In *International Conference on Information Fusion (FUSION)*, Washington D.C., USA, July 2015. ISIF.

[13] Rieken, Jens, Richard Matthaei and Markus Maurer: *Toward Perception-Driven Urban Environment Modeling for Automated Road Vehicles*. In *IEEE Intelligent Transportation Systems (ITSC)*, pages 731–738, Gran Canaria, Spain, Sep 2015. IEEE.

[14] Semrau, Marc: *Entwicklung eines Verfahrens zur Identifikation von ausgewählten Verkehrssituationen aus dem chinesischen Großstadtverkehr*. Master's thesis, Technische Universität Braunschweig, Braunschweig, 11.11.2014.

[15] Semrau, Marc, Jakob Erdmann, Bernhard Friedrich and Waldmann René: *Simulation framework for testing ADAS in Chinese traffic situations*. SUMO Proceedings, 2016:103–114.

[16] Shi, Bin, Li Xu, Jie Hu, Yun Tang, Hong Jiang, Wuqiang Meng and Hui Liu: *Evaluating driving styles by normalizing driving behavior based on personalized driver modeling*. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 45(12):1502–1508, 2015.

[17] Wang, Ruili, Wensheng Zhang and Qinghai Miao: *Effects of driver behavior on traffic flow at three-lane roundabouts*. International Journal of Intelligent Control and Systems, 10(2):123–130, 2005.

3 Investigating the application of an alternative fuel technology – A scenario-based simulation approach to analyze environmental impacts

Christina Flitsch, Sarah Pfoser, Matthias Neubauer, Oliver Schauer;
LOGISTIKUM Steyr, University of Applied Sciences Upper Austria, Austria
{Christina.Flitsch, Sarah.Pfoser, Matthias.Neubauer, Oliver Schauer}@fh-steyr.at

3.1 Abstract

The environmental impact of road traffic represents a key factor of today's traffic management. Thus, the potential of alternative fuel technologies needs to be considered in the design of future road traffic. Alternative fuel technologies aim to reduce emissions and to provide environmentally sustainable solutions. This paper investigates the application of Liquefied Natural Gas (LNG) as alternative fuel technology for Light-Duty and Heavy-Duty-Vehicles in Upper Austria. More than a quarter of Austrian exports stem from Upper Austria. Especially for Light-Duty and Heavy-Duty-Vehicles new solutions need to be found and translated into action. The investigation takes a scenario-based traffic simulation approach. Thereby, three scenarios (2025, 2030 and 2050) are defined for selected roads in Upper Austria and changes in emissions are compared. Technically, the SUMO simulation tool from ITS West is applied to implement the scenarios and gain statements about different emissions levels. SUMO provides two options to evaluate emissions. The first option is to use the built-in HBEFA functionality, the second is to use the extensive PHEMlight functionality. Results presented in this paper are derived with PHEMlight. Initial results indicate that alternative fuels will have a positive influence on the selected areas. Future work will consider the optimization of routings considering future LNG-gas stations in Upper Austria.

Keywords: Emission Simulation, Liquified Natural Gas (LNG), Scenario-based Emission Simulation.

3.2 Introduction

The transport sector causes more than a quarter of the total greenhouse gas emissions in the European Union. In 2014, road transport generated almost 73 % of total greenhouse gas emissions from transport and is therefore a severe obstacle to the proper compliance with environmental targets (2). As a matter of fact, measures have to be taken to reduce the negative impact of road transport. Adopting alternative fuels is a possible strategy to improve the environmental footprint of road vehicles. Several alternative fuel technologies have emerged recently including hydrogen, biofuels, electric and natural gas vehicles. Natural gas is considered to be the alternative that allows best to substitute conventional fuels, since it exhibits the potential to reduce environmental impacts and is easily available and accessible (23). Natural gas may either be used in a compressed state which is referred to as CNG (compressed natural gas), or in a liquid state which is referred to as LNG (liquefied natural gas). To convert natural gas to LNG, it needs to be cooled down to a temperature of -162°C. At this temperature gas becomes liquid and reduces its volume roughly 600 times. The volume reduction during the liquefaction process and the high energy density qualify LNG as well suited for heavy-duty vehicles (6). Actually, there are only a few substitutes readily available which fulfill long-haul transport requirements regarding engine power and fuel range. LNG represents an important alternative for trucks which has already been applied and tested by several fleet operators. For instance, the Spanish transport company Transordizia was the first one to send a truck fueled only by LNG right across Europe from Madrid to Hamburg in July 2016 (3). A major benefit of LNG is that the combustion of LNG causes nearly 99% less particle (PM) and sulfur oxide (SOx) emissions, around 80% less nitrogen oxides (NOx) and around 20% less carbon dioxide compared to diesel (18). The positive environmental effects can even be enhanced by mixing liquefied bio-methane into the fuel. The successful implementation of LNG depends, however, to a great extent on the approval and support of public authorities and decision makers which are supposed to promote the application of alternative fuels by regulations in favor of environmentally friendly technologies. In order to support informed decision making, evidence related to impacts of alternative fuel technologies is required. Traffic simulation represents a suitable means to investigate the effects of alternative fuels and inform decision makers. This paper investigates the application of Liquefied Natural Gas (LNG) as alternative fuel technology for Light-Duty and Heavy-Duty-Vehicles in Upper Austria. The investigation takes a scenario-based traffic simulation approach. In this context, three scenarios (2025, 2030 and 2050) are defined for selected roads in Upper Austria and emission impacts are compared applying microscopic traffic emission simulation. The remainder of this paper is structured as follows. First, the steps taken are within the study and the methods used are described on a general level. Second, concrete activities and results are detailed for each step. Finally, the results are discussed and future work is illustrated.

3.3 Study Approach

The generic steps taken in this study as well as the methods used are depicted in Figure 3.1. Subsequently, the steps are detailed with respect to a concrete application case in Upper Austria. For each step aspects such as involved stakeholder, input/output data, and methods and tools used will be detailed in the following subsections. In general, scoping represents the initial step and is intended to set the stage for a certain emission impact study. This may comprise activities like defining concrete goals and research questions of the study, identifying relevant stakeholder, identifying and selecting relevant background information. Within the defined scope, possible future situations may be predicted within certain scenarios. Scenario analysis is a method allowing to illustrate different future developments and to analyze related impacts. Given certain scenarios, concrete traffic simulations and emission simulation models can be developed and configured accordingly. These models provide the basis for concrete computer-based simulations which generate foreseen future emission developments. The generated emission data may then be used for data analysis and report generation for certain stakeholders and decision makers.

3.3.1 Scoping

Several studies point out that the market introduction of alternative fuels depends not only on public authorities, but generally on the actions of many stakeholders such as vehicle industry, fuel companies and consumers (e.g. (14; 10)). The study has been initiated by the following stakeholders:

- an LNG truck manufacturer

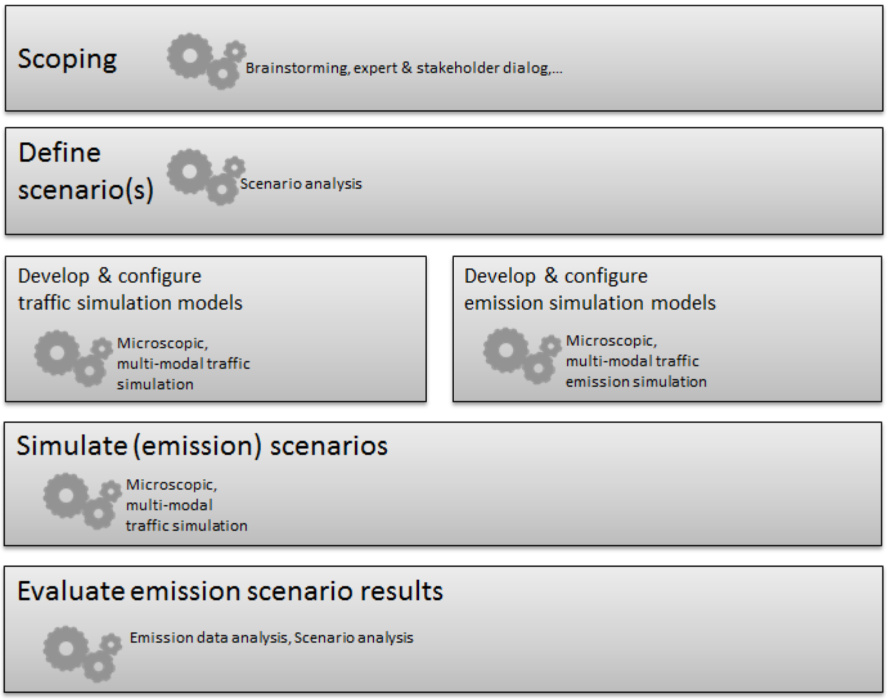

Scoping
Brainstorming, expert & stakeholder dialog,...
Define scenario(s)
Scenario analysis
Develop & configure traffic simulation models
Microscopic, multi-modal traffic simulation
Develop & configure emission simulation models
Microscopic, multi-modal traffic emission simulation
Simulate (emission) scenarios
Microscopic, multi-modal traffic simulation
Evaluate emission scenario results
Emission data analysis, Scenario analysis

Figure 3.1: Generic steps and applied methods

- a gas company in Austria aiming to construct and supply LNG filling stations in Austria
- an infrastructure operator intending to become an LNG hub in Austria
- an association strongly promoting the adoption of LNG as an alternative fuel
- public authorities playing a key role in promoting alternative fuels such as LNG
- researchers in the field of LNG and traffic simulation

In an initial stakeholder dialog, recommendations for LNG as alternative fuel for heavy traffic in Upper Austria have been identified as global goal of this study. Related to this goal, two main research questions were defined:

- RQ1: What are general conditions related to the diffusion of LNG for heavy traffic in Upper Austria?
- RQ2: How could the diffusion of LNG impact future traffic emissions in Upper Austria?

Addressing RQ1, the research team carried out desktop research and literature review to gain first insights into possible future developments of the LNG market. LNG Market adoption may be directly stimulated via incentive-based instruments by public authorities, since they play a key role in promoting alternative fuels such as LNG (5). These instruments include regulations in favor of LNG such as reduced fuel taxation, but also investment funding through subsidies or a guaranteed LNG price. Aside to the LNG market adoption, the applicability of LNG for heavy traffic and its economic feasibility has been investigated. The results indicate that LNG can be used internationally with respect to different types of traffic (e.g. city traffic, regional traffic, light offroad traffic). Furthermore, LNG is part of European guidelines (21) for alternative fuel technologies and there is even a European LNG strategy (11) favoring the diffusion of LNG. LNG represents a mature technology for heavy weight traffic which is economically feasible. Given the background information related to RQ1, scenarios have been developed in a following step. The scenarios are described in the next section and provide the basis for answering RQ2 via a scenario-based traffic emission simulation approach.

3.3.2 Scenario Definition

Several studies point out that the market introduction of alternative fuels depends not only on public authorities, but generally on the actions of many stakeholders such as vehicle industry, fuel companies and consumers (e.g. (10)Jannsen et al. 2006). For this reason we consulted several Austrian stakeholders from various domains to define realistic scenarios for the simulation. We organized a focus group meeting to bring them all together and discuss potential future developments of the LNG market based on the collected background investigations. Participants of this focus group included representatives from:

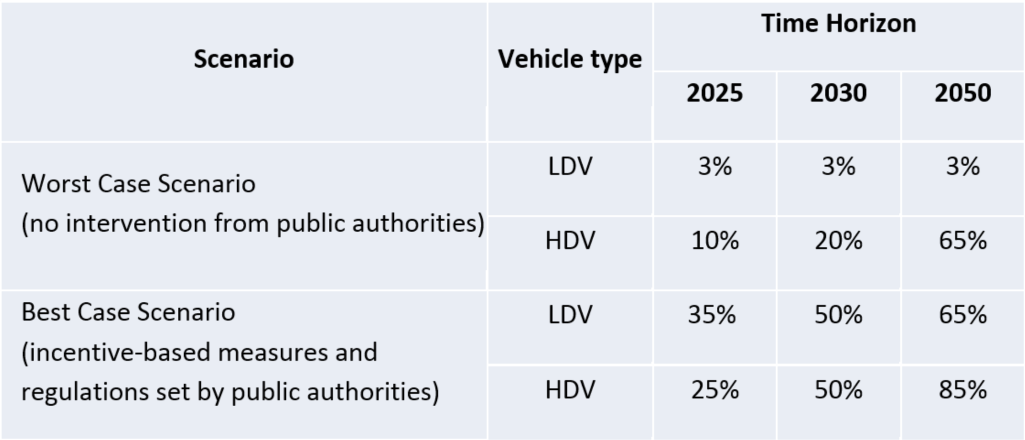
- IVECO as the leading European LNG truck manufacturer,
- RAG as the dominating gas company in Austria which aims to construct and supply LNG filling stations in Austria,
- Port of Enns as infrastructure operator with intentions to become an LNG hub in Austria,
- Pro Danube as an association which strongly promotes the adoption of LNG as an alternative fuel.

We also discussed the scenarios with NGV, the European Natural Gas Vehicle Association; however they were not present at the focus group meeting. The basic idea of the scenario planning method is that a best case and a worst case scenario are developed to cover the range of potential future developments (19). We adopted this approach and specified two main scenarios with the 'best case scenario' being a scenario with distinct incentive-based measures and regulations set by public authorities to facilitate LNG introduction. The considered 'worst case scenario' is characterized by a weak market development caused by missing intervention from public authorities. Three time horizons have been chosen to evaluate the potential development of LNG adoption, namely the years 2025, 2030 and 2050. These three time horizons cover the medium and long term periods which are most often also the target periods of various regulatory measures to reduce emissions. We additionally distinguished between light duty vehicles (LDV, gross vehicle weight between 3.5 and 7.5 tons) and heavy duty vehicles (HDV, gross vehicle weight of more than 7.5 tons). A specific motorway section between the cities of Ansfelden, Linz and Enns has been chosen as area under consideration since this is the section with the highest traffic load in Upper Austria according to the Austrian traffic club VCÖ (22). Following, table 3.2 summarizes the estimated best-case and worst-case shares of light-duty and heavy-duty LNG vehicles in the years 2025, 2030 and 2050. It should be noted that the scenarios are rather optimistic towards the future shares of LNG. This is due to discussion related to the expected prohibition of Diesel in Cities in the future. Given a prohibition of Diesel vehicles, LNG is a viable alternative. Therefore, the scenario definition resulted in a LNG optimistic projection of the future.

3.3.3 Traffic Model Development and Configuration

The traffic model builds upon the Austrian Graph Integration Platform (GIP). GIP.at is a nationwide transport graph which provides a digital map of Austria's transport network (1). The defined scenarios focus exclusively on routes in Upper Austria. For this reason, the graph for Upper Austria has been imported in SUMO to simulate the concrete scenario simulations. The GIP graph encodes the road network and represents the reference point for traffic information such as hydrographs related to road sections or traffic events.

The simulation of the different scenarios builds upon GIP and traffic data provided by 'ITS Austria West' (16; 15). This data base provides historic real-time traffic data retrieved via permanent counting stations from ASFINAG (7), Austria's Motorway Operator, and stations maintained by the



Scenario	Vehicle type	Time Horizon		
		2025	2030	2050
Worst Case Scenario (no intervention from public authorities)	LDV	3%	3%	3%
	HDV	10%	20%	65%
Best Case Scenario (incentive-based measures and regulations set by public authorities)	LDV	35%	50%	65%
	HDV	25%	50%	85%

Figure 3.2: Defined Simulation Scenarios

traffic department of Upper Austria. The data base provides a realistic portrait of the recent traffic volume in Upper Austria. Therefore, it has been applied within the simulation models to generate a realistic amount of LDV and HDV vehicles. Data retrieved by permanent counting stations only cover the total traffic volume for an average weekday and the total volume observed on typical weekend days. However, building the traffic model required to distribute the measured total traffic volume per day across the given day. The distribution builds upon a time variation curve (see Figure 3.3) which allows to distribute the traffic volume on an hourly basis (12).

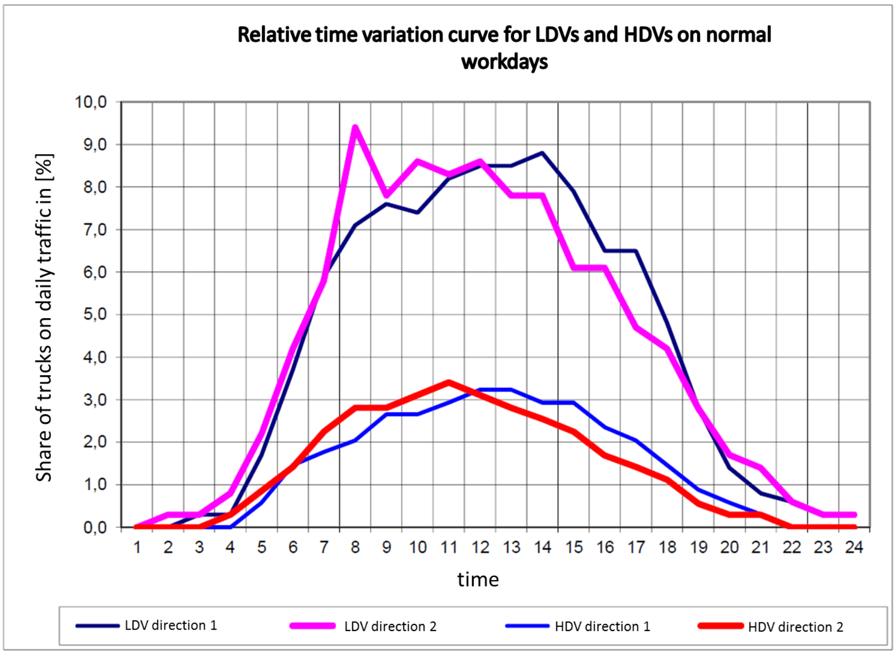


Figure 3.3: Traffic volume distribution

Aside the traffic volume distribution, the share of different vehicle types is relevant for building a traffic model. Historic real-time traffic data gained via permanent counting stations and household surveys allowed to determine the shares of delivery cars, trucks, and passenger cars to be used in the traffic simulation model. Figure 3.4 depicts the different vehicle type shares used in the traffic model. A further important step was the definition of the routes that build the reference point for the simulated vehicles. The traffic simulation model focused on the area of 'Knoten Linz' in Upper Austria. In this area, six routes were defined which cover the highway A1 and A7. The routes are depicted in Figure 3.5. The major object of emission investigation is the dashed road section termed 'measured links'.

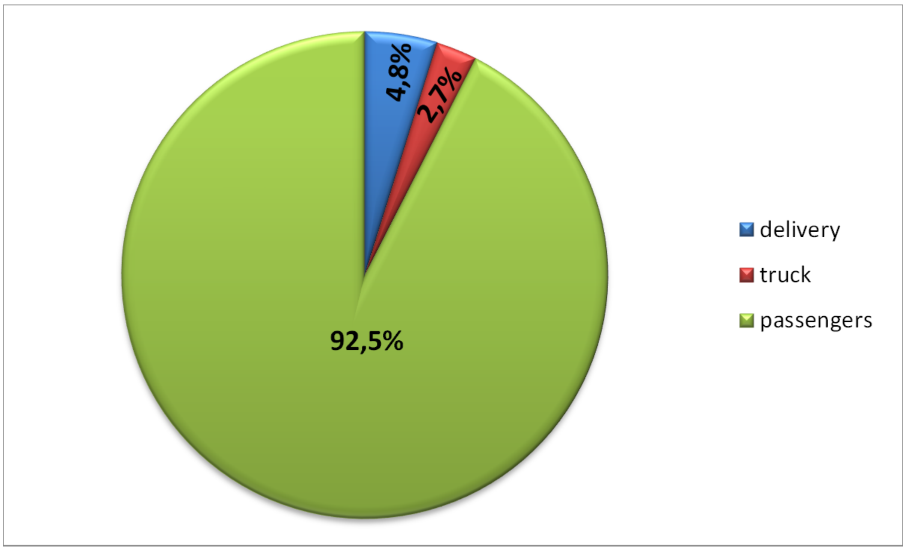


Figure 3.4: Distribution of the volume of traffic by different transport means

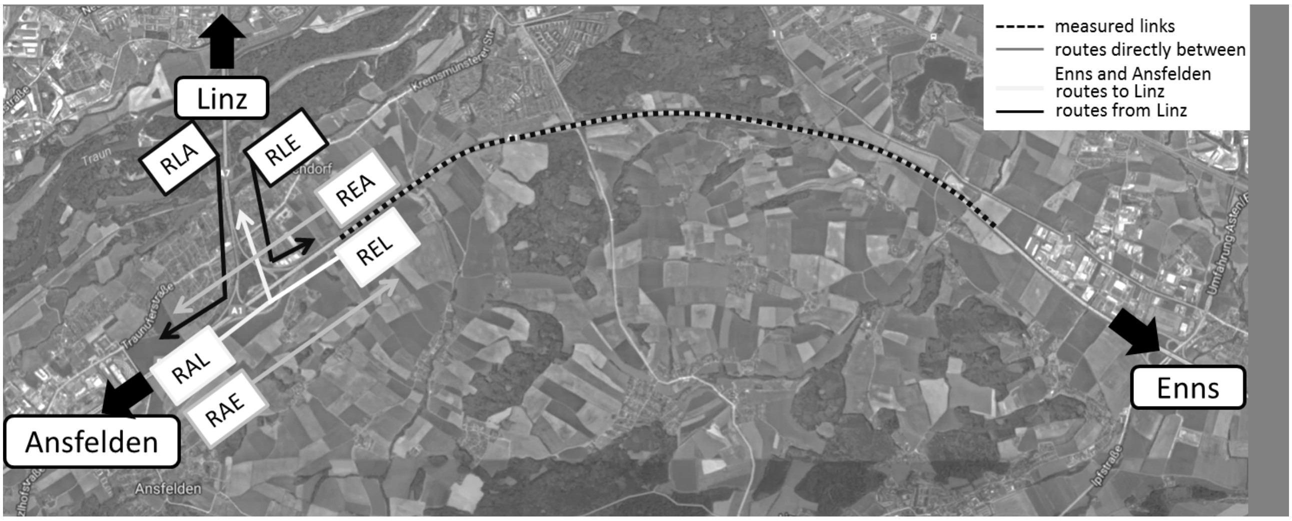
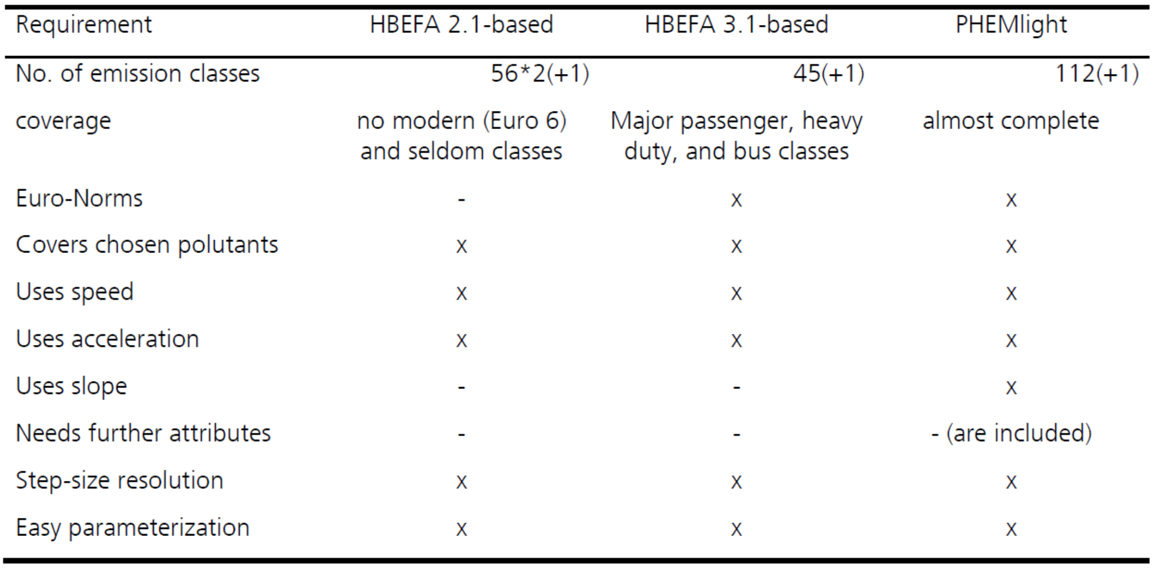


Figure 3.5: Routes implemented in the traffic simulation model and measured links

3.3.4 Emission Model Development and Configuration

One method to measure emission outputs on real-world data is to use traffic simulation. Different simulation programs are available commercially and open source. The most commons are products from PTV, like VISUM and VISSIM, and SUMO from DLR. Both can be used to simulate vehicle emissions. PTV is using HBEFA for the calculation of emissions (20). Calculating emissions with SUMO can be done with different emission models. Either using HBEFA (2.1 or 3.1 based) or using PHEMlight. PHEMlight is an implemented version of the emission model PHEM, developed by TU Graz. In Figure 3.6 a comparison between HBEFA classes and PHEMlight is shown.

PHEMlight does include almost every emission class. In PHEMlight emission curves, which represent real world driving cycles, are used. Since both, LNG and CNG, are composed of the same substance, i.e. natural gas, and only their aggregate phase is different, PHEMlight CNG emission classes were used in the emission simulation model for investigating LNG as alternative fuel technology. These emission classes were developed within the COLOMBO project (13). In the COLOMBO project CNG emission classes were developed for passenger cars, light duty vehicles [LDV] and heavy duty vehicles [HDV] (4). Generally, SUMO provides different output options related to emission simulation results (17). For simulating the impact of LNG on emission values in Upper Austria, emission outputs per links and per vehicles were used. Thereby, five vehicle types, with a different emission class each,



Requirement	HBEFA 2.1-based	HBEFA 3.1-based	PHEMlight
No. of emission classes	56*2(+1)	45(+1)	112(+1)
coverage	no modern (Euro 6) and seldom classes	Major passenger, heavy duty, and bus classes	almost complete
Euro-Norms	-	x	x
Covers chosen polutants	x	x	x
Uses speed	x	x	x
Uses acceleration	x	x	x
Uses slope	-	-	x
Needs further attributes	-	-	- (are included)
Step-size resolution	x	x	x
Easy parameterization	x	x	x

Figure 3.6: Comparison of HEBFA classes and PHEMlight, (17)

were added to the simulation:

- Passenger cars
- Delivery (emission class: LDV; HBEFA)
- CNG-delivery (emission class: LCV_III_CNG_EU6; PHEMlight)
- Trucks (emission class: HDV; HBEFA)
- CNG-trucks (emission class: HDV_CB_CNG_EU6; PHEMlight)

To do an approximation on CNG-trucks in the simulation the heavy duty vehicle class [HDV] of buses was used.

3.3.5 Scenario Simulation

The simulation of the different scenarios in SUMO builds upon the above described traffic and emission models. The diverse simulations for the given scenarios were performed within the timeframe 0:00–9:30. This timeframe has been selected because it includes various traffic situations, e.g. partially congested, heavily congested, or free flowing. Furthermore, the timeframe includes the peak of heavy duty traffic. In an initial simulation attempt, the SUMO built-in duarouter was used to create the vehicles and routes for selected areas. Thereby, the duarouter automatically creates a route-file. However, the emission output data revealed an issue related to the comparability of the different scenarios. Since the duarouter defines the departure time and the routes of certain vehicles within the simulations, simply changing the amount of vehicles for the given vehicle types in the different scenarios does not lead to proper results. In other words, the duarouter creates on a specific link different amounts of vehicles within a given timeframe among different scenarios. For this reason, this attempt to compare emissions outputs has not been regarded suitable. In a second simulation attempt, the departure time of vehicles and the route of certain vehicles were manually defined based upon historic real-time data gained via permanent counting stations. As a result, for the given scenarios six different route-file were defined as input for the SUMO simulation. The simulation of the different scenarios generates emission output data. Actually, there are different emission output modes, i.e. emissions per vehicle, emissions per lane, or emission per edge. The aim of this study was to compare emissions related to a certain road section. For this reason, sumo emission outputs were generated per lane for the different scenarios. In addition, one may configure the time granularity of the emission output within a simulation time frame. In other words, emissions related to a certain lane could be generated on the basis of seconds, minutes, hours, etc. in a given time frame, e.g. 0:00–9:30. In this study, the emission output per lane was generated for the total time frame. The generated XML output data comprises normed emissions of CO2, NOx and PMx in [g/km/h] per vehicle type. The described data analysis in the subsequent section is done via Microsoft Excel.

3.3.6 Scenario Evaluation

The results presented in this section are dedicated to answer RQ2: How could the diffusion of LNG impact future traffic emissions in Upper Austria? To tackle this question the generated emission data of the simulation were analyzed. Thereby, simple data analysis techniques were used to compare the emission output of the scenario simulations. First, the impact on the total amount of emissions in the measured area was analyzed. This means, all vehicles implemented in the simulation scenarios, passenger cars, LDVs, LNG-LDVs as well as HDVs and HDVs-LNG were measured (Figure 3.7). As Figure 3.7 shows, NOx and PMx emissions decrease significantly. Comparing the worst case scenario 2050 (without any incentives or regulations in favor of LNG) with the current state, NOx-emissions decrease around 16%. In the best case scenario 2050 (with incentives and regulations), the LNG-vehicles reduce NOx-emissions by 26%. However, no significant impact on CO2-emissions can be observed, due to the high share of passenger cars. For this reason, particularly the impact of LDVs and HDVs on emissions was analyzed in a further step.

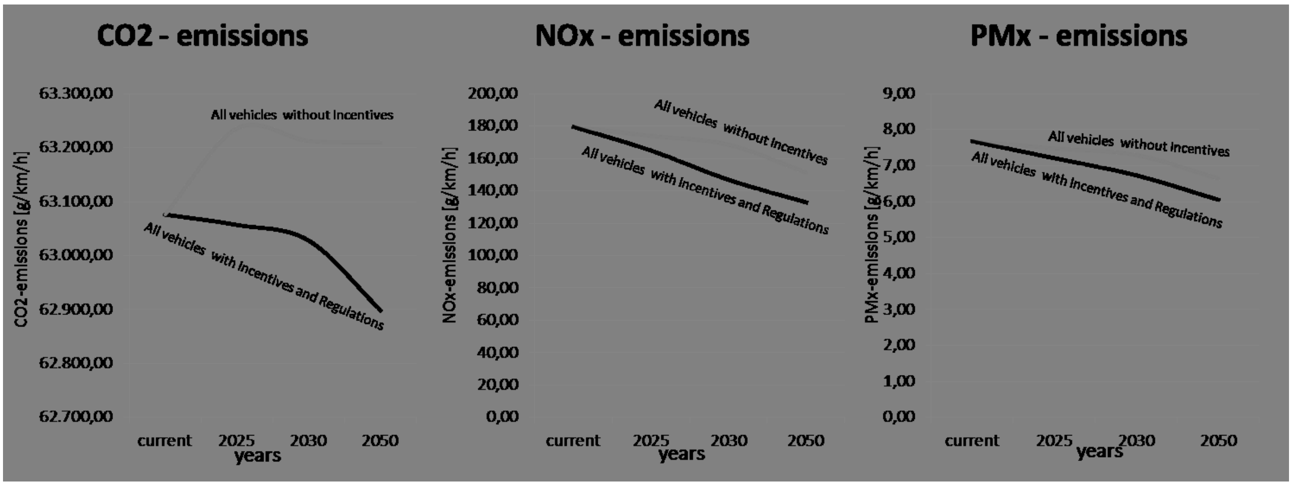


Figure 3.7: Emission-Output related to all vehicles in the selected area of A1 and A7 between 0:00 - 9:30

Figure 3.8 indicates that LNG has a strong impact on CO2, NOx and PMx emissions. The worst case scenarios assume a constant share of LDVs (3%) for the time horizons 2025, 2030, 2050. A share of 3% LNG-LDVs could reduce CO2-emissions around 21%, NOx-emissions around 19% and PMx-emissions up to 12.5%. Further analysis showed that incentives and regulations in favor for LNG provide the potential to decrease CO2-emissions of LDVs up to 45% in 2050. A comparison between the scenario of the current state and the scenario of 2050 also shows that NOx-emissions decrease around 82% and PMx-emissions up to 47%. Looking at the HDVs, the analysis shows that CO2 emissions could decrease between 63% in the worst case scenario (65% LNG-HDV), and 81% in the best case scenario (85% LNG-HDV). Interestingly, NOx-emissions could be decreased about 93% and PMx-emissions around 96%.

The emission results comparison of the scenarios 2025, 2030 and 2050 indicates that there is a higher reduction of emissions until 2030 than until 2050. The emission between 2030 and 2050 are not

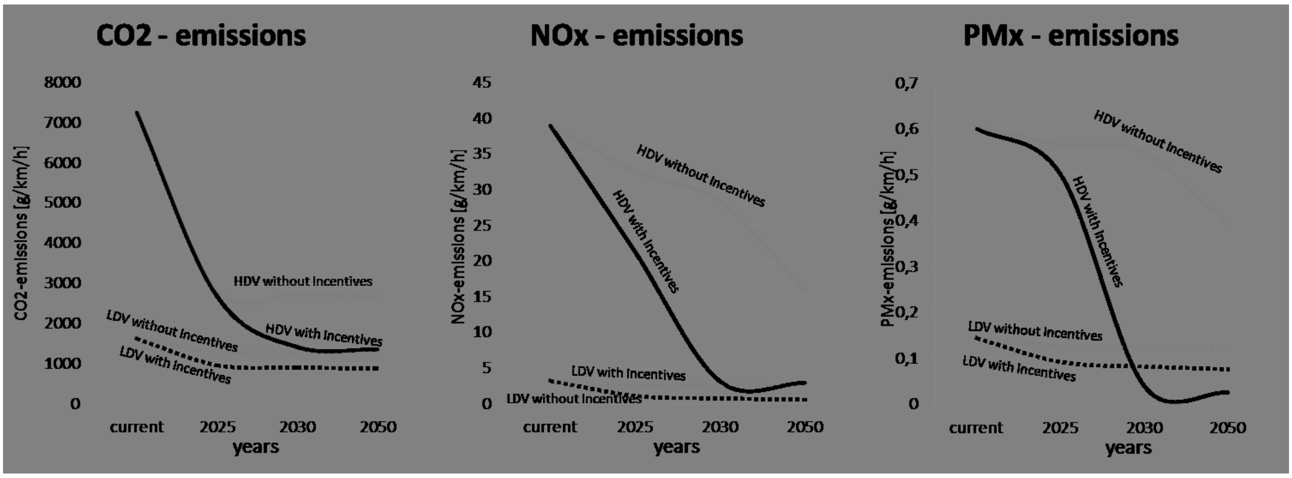


Figure 3.8: Emission-Output related to LDV and HDV vehicles in the selected area of A1 and A7 between 0:00 - 9:30

decreasing proportionally to the amount of used LNG-trucks. For example, the gap between the emissions of the current state and 2025 with incentive plus regulation 35%-LNG-LDVs CO2 emissions is around 41%. Increasing the amount of LNG-LDVs up to 50%, it reduces CO2-emissions around 44% - only 3% less than 2025. Furthermore, increasing the amount of LNG-LDVs up to 65% does decrease CO2-emissions around 46%. The increased amount of HDVs indicates that the gap between 2025 and 2030 is higher than between 2030 and 2050. For example, the scenario with incentive plus regulation does provide 25%-LNG-HDVs in 2025. Within this regulation, 63% CO2-emissions could be reduced. 2030 50%-LNG-HDVs could decrease CO2-emissions up to 80.5% - 18% more than 2025. In addition, the results show that increasing LNG-HDVs up to 85% in 2050 could reduce CO2-emissions by around 81%.

3.4 Discussion

Implementing LDVs and HDVs running on the alternative fuel LNG strongly impacts the total amount of NOx and PMx emissions. The simulation results reveal that increasing the amount of LNG vehicles does not proportionally decrease the produced emissions. A possible explanation for this might be that no average values of CNG emission data are used in the simulation. Instead, emission maps are implemented in SUMO to consider driving dynamics. Notably, even though more vehicles are driving on LNG, the driving behavior did not change. Hence, it could be hypothesized that there is a connection between the driving dynamic and the decrease of the emissions. This hypothesis indicates that intervening in the traffic flow could help to reduce emissions. A known limitation of this study is the used traffic volume within the different emission simulation scenarios. The presented simulation model assumes a similar traffic volume within the different future scenarios. For this reason, future work could incorporate a traffic forecast for the years 2025, 2030 and 2050. A further limitation of the study is the mixture of the different emission models, HBFA and PHEMlight. As indicated by (8), there are considerable differences between emission models. This would actually require an extra evaluation of the presented results. However, at the time of conducting the study the authors were not aware of these differences. Therefore, the authors consider it as important future work.

3.5 Conclusion

The aim of this paper was to examine the environmental impact of adopting LNG as an alternative fuel for light and heavy duty vehicles in a specific area of Upper Austria. The simulation has been carried out in order to visualize and quantify the potential positive effects of introducing LNG on one of the main routes in Upper Austria. The results could provide politicians evidence to support the implementation of LNG. To our knowledge, this is the first study that investigates the effect of LNG on a specific road section in Upper Austria. This study has shown that the amount of CO2, NOx and PMx emissions do decrease with increasing the amount of LNG-vehicles. The results are of practical use, since decreasing particulate matter is a goal of major importance in Austria. The current output data highlighted the effectiveness of introducing LNG fueled vehicles. Even within analyzing the emissions output of all vehicles, a strong influence on particular matters was detected. Future work should address further calibrations of the traffic simulation model, especially with respect to heavy duty vehicles. Furthermore, future work in Upper Austria could consist of investigating additional routes and points of interest for traffic management. On the one hand, routings with VIA-Points at LNG- gas stations could be used for developing either optimized routes for LNG-trucks or the selection of LNG gas station locations. In this context, the influence of LNG on emissions related to different hours and/or different traffic situations could also be analyzed. On the other hand, in order to reduce emissions in certain areas of Upper Austria, the route supporter functionality of SUMO[1] could be used. Moreover, the emissions simulation could be improved by considering altitudes. Further simulations could investigate the impact of LNG-vehicles on noise emissions, e.g. by using the tool Harmonized. This is of specific interest, since LNG has the potential to reduce noise emissions (9). In case decision makers require detailed information in fuel consumption, the simulation also allows to explore the fuel consumption, and to calculate the fuel costs for a specific route.

3.6 Bibliography

[1] *Graph Integration Platform GIP*. http://gip.gv.at/home-en.html, Accessed: 6.2.2017.

[2] *Greenhouse gas emissions from transport*, 2016. http://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-greenhouse-gases, Accessed: 6.2.2017.

[3] *LNG, an alternative to diesel in land transportation*, 2016. http://www.energia16.com/lng-an-alternative-to-diesel-in-land-transportation/?lang=en, Accessed: 6.2.2017.

[4] *PHEM (Passenger Car and Heavy Duty Emission Model)*, 3.1.2017. http://sumo.dlr.de/wiki/Models/Emissions/PHEMlight, Accessed: 6.2.2017.

[5] Arteconi, A and F Polonara: *LNG as vehicle fuel and the problem of supply: The Italian case study*. Energy policy, 62:503–512, 2013.

[6] Arteconi, Alessia, Caterina Brandoni, Dario Evangelista and Fabio Polonara: *Life-cycle greenhouse gas analysis of LNG as a heavy vehicle fuel in Europe*. Applied Energy, 87(6):2005–2013, 2010.

[7] Asfinag: *Asfinag Dauerzählstellen 2016*. http://www.asfinag.at/, Accessed: 6.2.2017.

[8] Behrisch, Michael and Jakob Erdmann: *Comparing Emission and Traffic Flow models of different categories*. In *Models and Technologies for Intelligent Transportation Systems (MT-ITS), 2015 International Conference on*, pages 164–168. IEEE, 2015.

[9] Boris Schmidt: *Die Kühlkette darf nicht reißen*. Frankfurter Allgemeine, 18.04.2013. http://www.faz.net/aktuell/technik-motor/auto-verkehr/lastwagen-mit-lng-die-kuehlkette-darf-nicht-reissen-12149186.html, Accessed: 6.2.2017.

[10] ENGERER, HELLA and MANFRED HORN: *Natural gas vehicles: An option for Europe*. Energy Policy, 38(2):1017–1029, 2010.

[11] EUROPEAN COMMISSION: *COMMUNICATION FROM THE COMMISSION TO THE EUROPEAN PARLIAMENT, THE COUNCIL, THE EUROPEAN ECONOMIC AND SOCIAL COMMITTEE AND THE COMMITTEE OF THE REGIONS on an EU strategy for liquefied natural gas and gas storage*. https://ec.europa.eu/energy/sites/ener/files/documents/1_EN_-ACT_part1_v10-1.pdf, Accessed: 6.2.2017.

[12] HARALD FREY: *Verkehrsaufkommen B20 Untersuchung*. https://s34.at/fileadmin/user_-upload/pdfs/Verkehrsaufkommen_B20.pdf, Accessed: 6.2.2017.

[13] HAUSBERGER, STEFAN and DANIEL KRAJZEWICZ: *COLOMBO Deliverable 4.2: Extended Simulation Tool PHEM coupled to SUMO with User Guide*. http://elib.dlr.de/98047/, Accessed: 6.2.2017.

[14] JANSSEN, ARTHUR, STEPHAN F LIENIN, FRITZ GASSMANN and ALEXANDER WOKAUN: *Model aided policy development for the market penetration of natural gas vehicles in Switzerland*. Transportation Research Part A: Policy and Practice, 40(4):316–333, 2006.

[15] KARL–HEINZ KASTNER, ROBERT KEBER, PETRU PAU and MARTIN SAMAL: *Real-Time Traffic Conditions with SUMO for ITS Austria West*. In KARSTEN LEMMER (editor): *1st SUMO User Conference 2013*, volume Band 21 of *Berichte aus dem DLR-Institut für Verkehrssystemtechnik*, pages 43–53, Berlin-Adlershof, 2013. Deutsche Zentrum für Luft- und Raumfahrt.

[16] KARL–HEINZ KASTNER, PETRU PAU: *Experiences with SUMO in a Real-Life Traffic Monitoring System*. In KARSTEN LEMMER (editor): *Proceedings of the SUMO2015*, pages 1–10. Deutsches Zentrum für Luft- und Raumfahrt, 2015.

[17] KRAJZEWICZ, DANIEL, MICHAEL BEHRISCH, PETER WAGNER, RAPHAEL LUZ and MARIO KRUMNOW: *Second Generation of Pollutant Emission Models for SUMO*. In BEHRISCH, MICHAEL and MELANIE WEBER (editors): *Modeling Mobility with Open Data: 2nd SUMO Conference 2014 Berlin, Germany, May 15-16, 2014*, pages 203–221. Springer International Publishing, Cham, 2015.

[18] KUMAR, SATISH, HYOUK-TAE KWON, KWANG-HO CHOI, WONSUB LIM, JAE HYUN CHO, KYUNGJAE TAK and IL MOON: *LNG: An eco-friendly cryogenic fuel for sustainable development*. Applied Energy, 88(12):4264–4273, 2011.

[19] LINDGREN, MATS and HANS BANDHOLD: *Scenario Planning-Revised and Updated: The Link Between Future and Strategy*. Springer, 2009.

[20] PTV GROUP - EMISSIONSMODELLIERUNG. http://vision-traffic.ptvgroup.com/de/produkte/ptv-vissim/anwendungsfaelle/emissionsmodellierung/, Accessed: 6.2.2017.

[21] THE EUROPEAN PARLIAMENT AND THE COUNCIL OF THE EUROPEAN UNION: *Directive 2014/94/EU of the European Parliament and of the Council of 22 October 2014 on the deployment of alternative fuels infrastructure Text with EEA relevance*. http://data.europa.eu/eli/dir/2014/94/oj, Accessed: 6.2.2017.

[22] VCÖ: *Auf A1 bei Traun sind in Oberösterreich die meisten Pkw unterwegs*. http://bvs.probahn.at/files/pa/1543.pdf, Accessed: 6.2.2017.

[23] Yeh, Sonia: *An empirical analysis on the adoption of alternative fuel vehicles: the case of natural gas vehicles*. Energy Policy, 35(11):5865–5875, 2007.

4 rFpro & SUMO: The Road To A Complete Real-Time Simulation Of Urban Environments for DIL, ADAS and Autonomous Testing

Alain Cottignies, Matthew Daley, Erik Newton, Hannah Parker; rFpro

4.1 Abstract

This paper summarises the initial integration work completed by rFpro and SUMO with the aim of creating a complete real-time simulation of urban environments for DIL, ADAS and Autonomous Testing. A complex traffic network for an rFpro model of public roads in Shanghai, China, is presented. The commitment of rFpro to further enhance the SUMO integration and deliver networks to customers is also summarised. Screenshots from the real-time rendered simulation are provided.

Keywords: DIL, Driver In The Loop, ADAS, Autonomous Driving, Urban Environments, SUMO, Real-time, simulator, sensor testing, low latency, high quality, graphics, visualisation, traffic, ego vehicle.

4.2 Introduction

rFpro has been supplying vehicle dynamics grade simulator software and digital road models for nine years. Engineers and researchers have used real-time simulation to develop, test and validate many areas of a vehicle's performance such as the vehicle dynamics' ride, handling and steering characteristics, chassis control systems, powertrain control system strategies and more recently to develop the critical safety features of ADAS and Autonomous vehicles. Public road driving requires both the driver and their vehicle's control systems to make many decisions about the scenario they are driving in. This means that any simulation that aims to supplement or replicate real-world testing must be able to stimulate the correlated responses from both the human test driver and the vehicle's own sensor network. The primary sense for any driver's decision making (human or machine) is their visual and depth perception of the surrounding area. If simulations are to give accurate, high fidelity results then the 3D area around the driver must be modelled and recreated in real-time to a high degree of geometric and visual accuracy.

rFpro aims to lead the professional simulation marketplace in producing 3D digital road models of the highest quality and accuracy. Adding intelligent simulated traffic to these digital road models is best served by experts from a number of fields. The ego vehicle under test is usually controlled by a human test driver or a driver-model attached to the plant model for the ego vehicle, with the objective of improving understanding of that vehicle and its behaviour in order to develop the desired attributes. Additional dynamic objects in a scene, such as traffic vehicles, cyclists or pedestrians require their own controller to produce realistic movement. The behaviour of this AI traffic should be sufficiently convincing to create a realistic response in both a human test driver and in any ADAS or Autonomous control systems being tested. rFpro have developed and will continue to develop a real-time connection to SUMO (1), in order to control the traffic items in a scene to enhance the surroundings for the live driver and vehicle systems under test.

4.3 Real-Time Simulation -Systems and Hardware Overview

"Driver" or "Sensor in the loop" real-time simulations often follow the same system outline, see figure 4.1. A real-time simulation can be kept simple on a single PC or can stretch over many pieces of hardware, such as the hardware layout shown in figure 4.2.

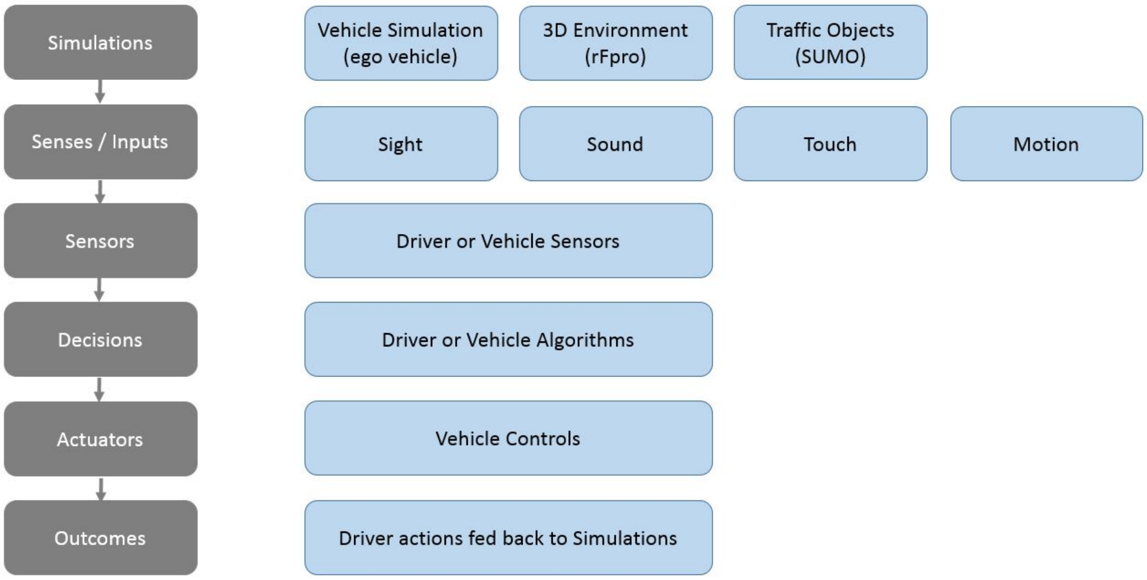


Figure 4.1: System outline for complete real-time simulation

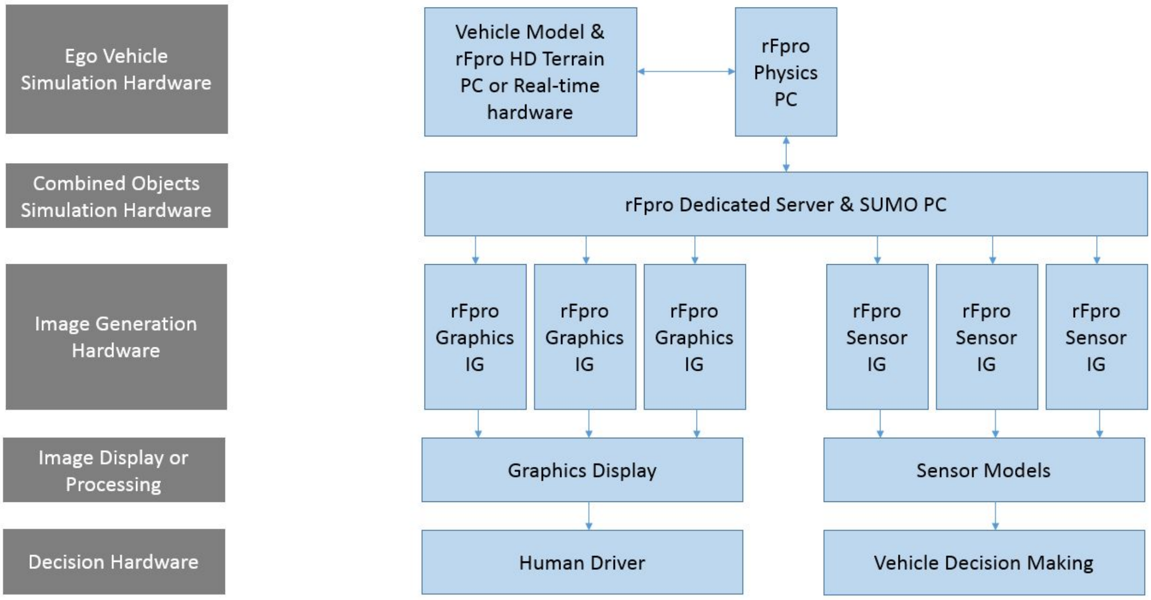

Ego Vehicle Simulation Hardware
Vehicle Model & rFpro HD Terrain PC or Real-time hardware
rFpro Physics PC
Combined Objects Simulation Hardware
rFpro Dedicated Server & SUMO PC
Image Generation Hardware
rFpro Graphics IG
rFpro Graphics IG
rFpro Graphics IG
rFpro Sensor IG
rFpro Sensor IG
rFpro Sensor IG
Image Display or Processing
Graphics Display
Sensor Models
Decision Hardware
Human Driver
Vehicle Decision Making

Figure 4.2: Hardware layout for a real-time simulation running with a multi-channel system

The recent addition of the SENSOR_IG allows rFpro to feed multiple sensor models, synchronised to within 100μs, even in real-time. The most complex customer use case has a total of 28 sensors on the ego vehicle including 6 cameras giving 360 degree vision, medium and long range radar sensors, LiDAR and ultrasound sensors. Figure 4.3 shows how this allows for many of the vehicle systems under test to be exercised via the real-time simulation.

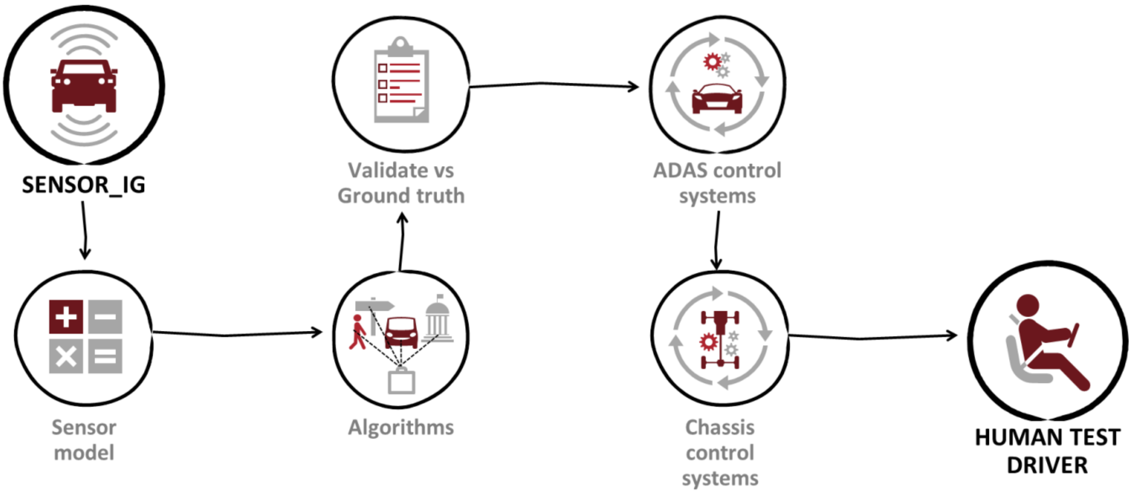


Figure 4.3: Testing out the complete vehicle ADAS components using rFpro SENSOR_IG

4.4 Real-Time Simulation -rFpro and SUMO

rFpro has a plugin architecture which means it can be configured flexibly depending on customer requirements.

In a typical SUMO configuration, the customer will have a Physics plugin to provide a high fidelity (drivable) ego vehicle. This can be an interface to one of the many vehicle dynamics packages which rFpro supports (CarSim, CarMaker etc.) or directly to a customer's own vehicle model. rFpro provides a 3D visualisation for the driver and/or sensor models as well as ground data for the EGO Physics. Using rFpro's plugin API, a user can inject many vehicles and control each object individually to their own real-time location data, but the problem of controlling a large number of vehicles across a complex road network quickly becomes a rather large project. SUMO offers end users a very quick entry point into creating highly complex traffic situations across networks with the added benefit of being a free to use simulation that a user can download and install easily. The most time consuming entry point is the creation of the road network, which rFpro are now able to do and supply alongside the graphical simulation. As part of the work integrating rFpro and SUMO, a new plugin was developed which uses the TraCI API to link to SUMO. The plugin adds an ego vehicle to a predefined SUMO scenario and subscribes to other vehicles and traffic signals within a radius. The general layout of this system is shown in figure 4.4.

Periodically, rFpro sends an updated position of the ego vehicle to SUMO and requests it to run on a single step, typically 10ms. SUMO is synchronised to rFpro, however rFpro will use the latest states it has received from SUMO. rFpro cannot wait for SUMO as it is soft-real-time, and driver immersion with the main physics model is more important. Generally, SUMO can complete its simulation step and return results within the 10ms, but on occasion when this does not happen, rFpro can extrapolate traffic vehicle states from the last received positions. In order to increase the immersive look of these vehicles, the traffic vehicle motions are enhanced by rFpro in order sit cleanly on the ground surface and move in a more natural way, e.g. with vehicle pitch and roll as the vehicle passes over the rFpro ground surface and articulation of the wheels and suspension as vehicles pass over uneven road surfaces. Realistic audio is also rendered to match the vehicles' estimated powertrain behaviour. The traffic vehicle shapes are rendered depending on the vehicle type assigned to each object. The

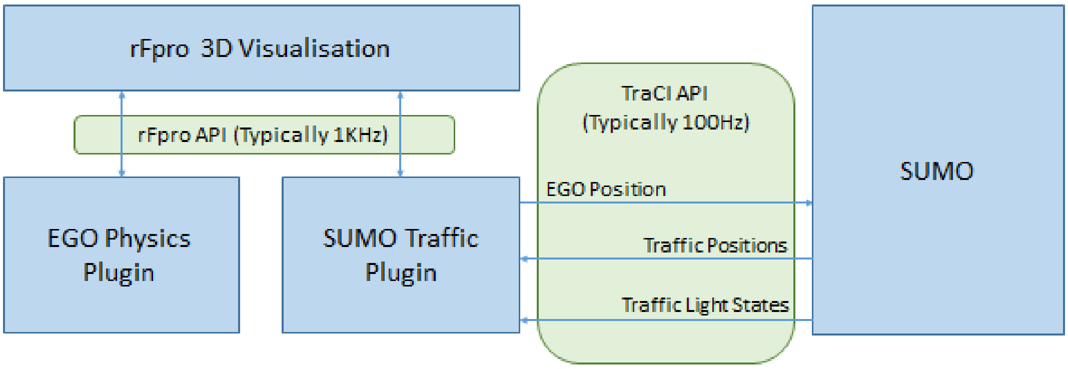


Figure 4.4: Data flows between rFpro, SUMO and the end user's vehicle model

vehicle brake lights and indicators are also fed from SUMO through to the graphical image. The traffic lights are controlled by the network definition in SUMO and are displayed inside the rFpro scene, allowing the human or autonomous vehicle under test to see the same traffic flow control signals as the SUMO traffic. An example of SUMO controlled objects obeying the same traffic lights as a human controlled vehicle is shown in figure 4.5.

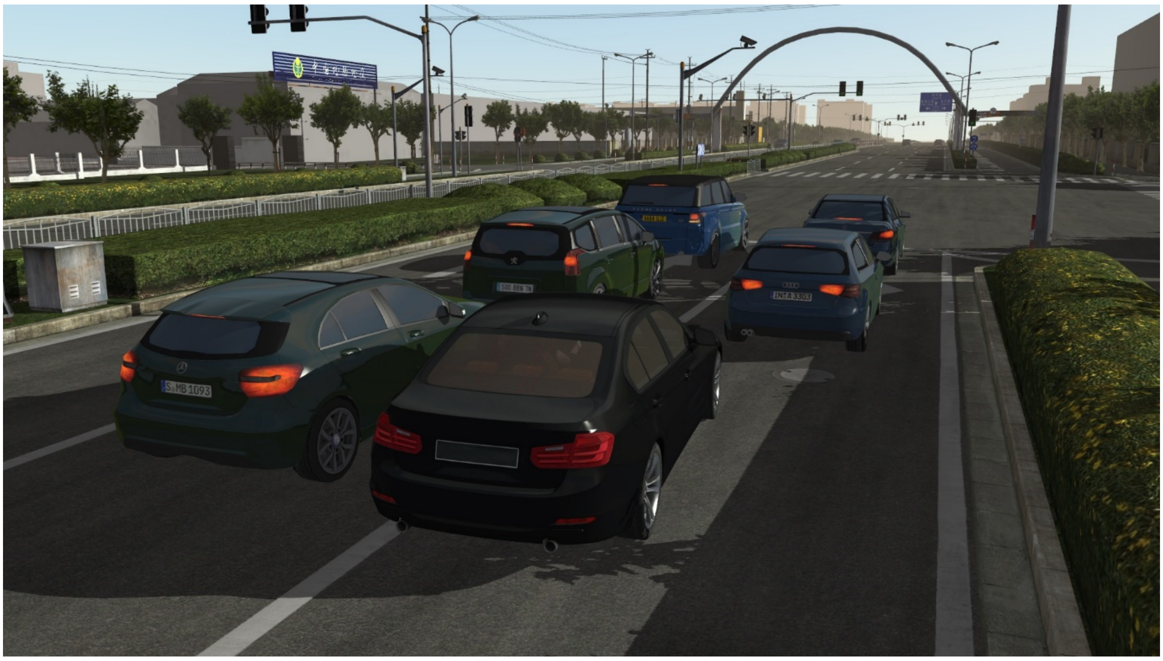


Figure 4.5: Ego vehicle (black) sat with SUMO controlled vehicles at traffic lights

4.5 Technical Integration and SUMO Enhancements

4.5.1 Injecting the Ego Vehicle

The initial problem to overcome on the integration project was to get SUMO to recognise the location of the rFpro ego vehicle. SUMO is generally used for traffic simulations where all vehicles are on a road. Their state is represented by a lane ID and distance along that lane. In the rFpro world, a real driver is free to cut corners or park on the side of the road. Initially this sort of behaviour led to the ego vehicle either snapping to the nearest lane or being ejected from the SUMO simulation entirely. Snapping to the nearest lane would give unrealistic interactions between the ego vehicle and the traffic, e.g. if the driver parked across 2 lanes, only one lane would be physically blocked. Being ejected from the SUMO simulation entirely meant rFpro would not receive any traffic position subscriptions and traffic would disappear from the rFpro visualisation.
Working with the SUMO team the function MoveToXY was added to the TraCI API. This allows the ego vehicle to be fully positioned on or off the SUMO road network and the traffic behave correctly.

4.5.2 Subscription Messages

Once the ego vehicle position was controllable, it became apparent that the subscription messages were not implemented for the C++ TraCI API (only for Python). These were implemented by rFpro and submitted as a patch. Working with SUMO developers, the patches were accepted and now form part of the general SUMO release.

4.6 Urban Environments

SUMO network definitions have been produced for a number of public road models. The most complex to date has been a road network of a combination of 3-lane dual carriageway, multi-lane single carriageway and back streets in Shanghai, China. rFpro have developed in-house tools to semi-automatically generate SUMO net files from their location assets. Most of the lane data is generated automatically whilst junctions require manual editing.
An example of one of the intersections from this model is shown as figure 4.6.

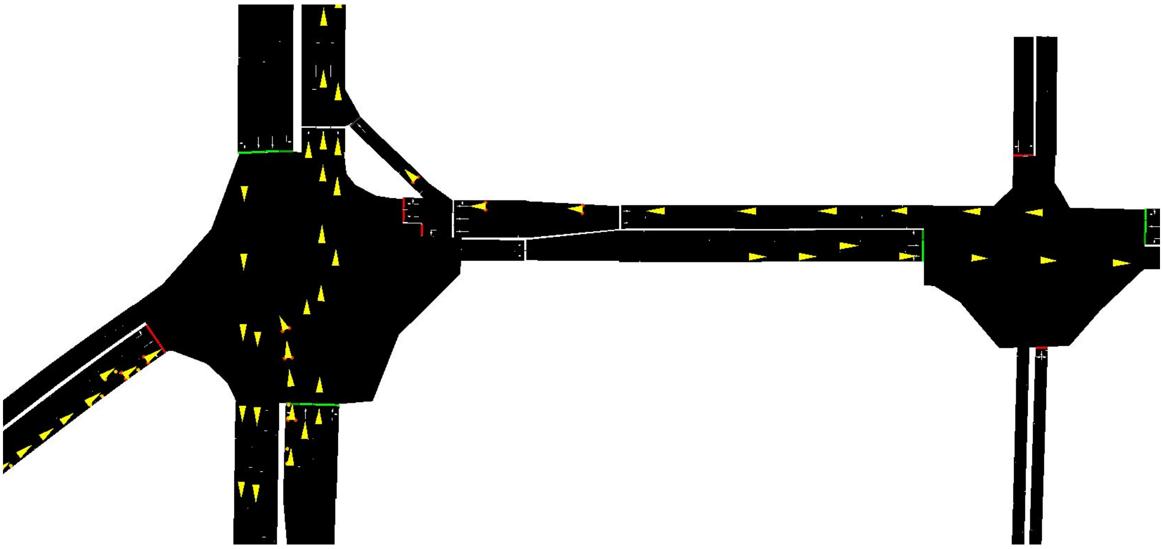


Figure 4.6: SUMO example intersection from the rFpro Shenjiang Rd, Shanghai, China model.

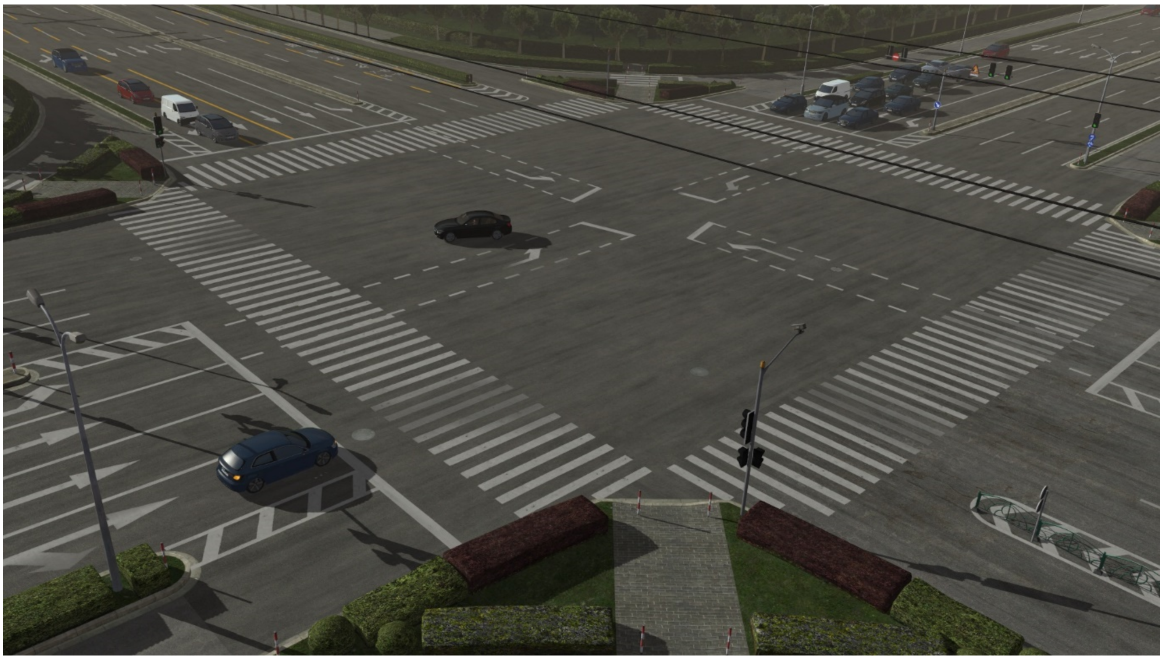


Figure 4.7: rFpro screenshot of an intersection from the rFpro Shenjiang Rd, Shanghai, China model.

The public road library of rFpro and SUMO networks now extends to models in the USA, France, Germany, China and additional routes planned in 2017 for the UK, Germany and Canada. Examples of these roads are shown in figures ref4figure7 and 4.8.

4.7 DIL, ADAS and Autonomous Testing

In order that a driver (either human or autonomous) may react correctly to their surroundings they must become fully immersed in the driving activity and environment. The addition of dynamic objects, such as other traffic vehicles, is an essential part of normal driving. These additional dynamic objects must appear to the driver to both look and react in a natural way. The reaction and control of these objects is the core focus of SUMO. The natural look of these objects has been focused on by rFpro by the use of real-time rendered vehicle models and the additional motion control written into the vehicle rendering pipeline. In a Driver In The Loop (DIL) setup, the rFpro real-time rendering creates a high quality image, with correct time of day shadows, weather and sky conditions, atmospheric fog and haze. The high refresh rate of the image and low latency, coupled with the high quality graphical shaders and lighting are all aimed at creating the highest level of driver immersion possible. 3D rendering is supported for three types of system:

- Full scale, multi-projector systems, possibly around a motion system.
- Head mounted display (HMD) VR displays
- Desktop workstations with 1 or more flat screen displays.

For ADAS and Autonomous Driver testing and validation, the 3D world, along with the dynamic objects controlled by SUMO, can be accessed via a number of different options. For real-time sensor models there is access to 32bit per colour image and depth buffers allowing the direct feeds for camera, LiDAR, Radar and ultrasonic sensor models. For live ground-truth validation, the positions of all of the dynamic objects controlled by SUMO or other systems can be accessed by the users' vehicle control systems. The 3D positions of static objects are also available to objects lists. This area of using simulation for high fidelity sensor testing is rapidly evolving. The integration of high quality

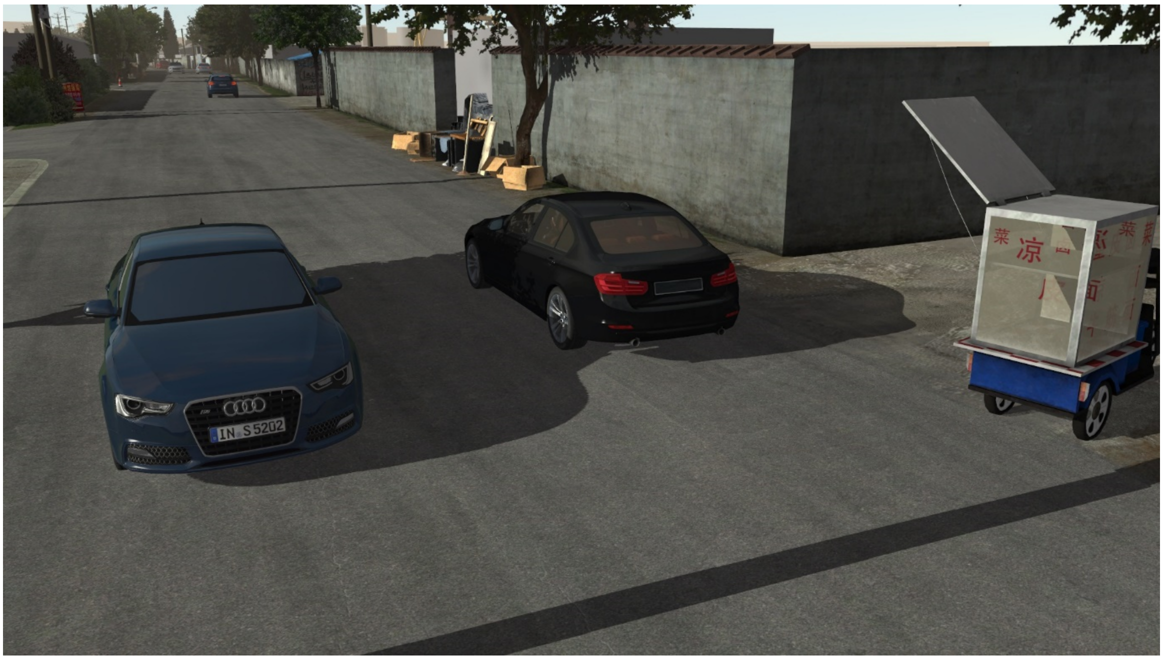


Figure 4.8: rFpro screenshot of a minor street from the rFpro Shenjiang Rd, Shanghai, China model.

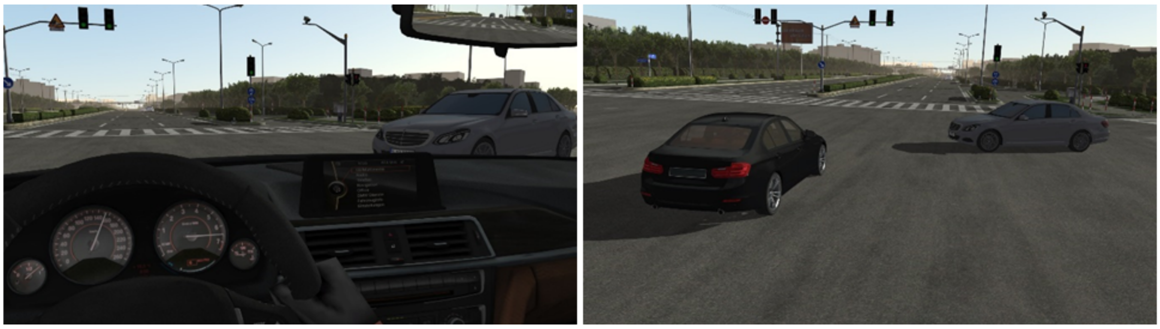


Figure 4.9: SUMO controlled traffic and human controlled ego vehicle at an intersection

products such as SUMO and rFpro creates a very powerful and capable toolset for development engineers to push forward with their research and engineering projects.

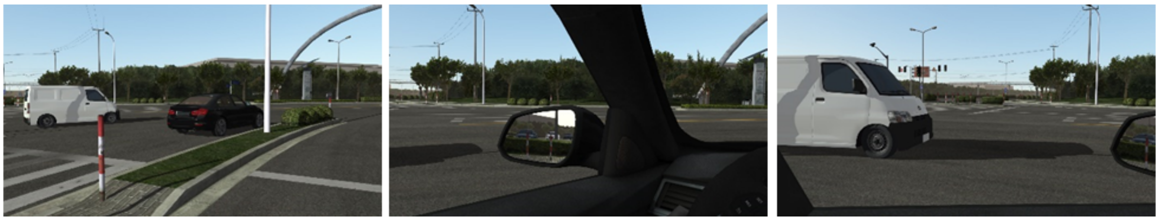


Figure 4.10: SUMO controlled traffic in the "blind spot" of a human controlled ego vehicle at a filter lane

4.8 Example Images and Further Reference Videos

A few static images of ADAS scenarios tested using rFpro and SUMO are contained here in the paper as figures 9 and 10. There are also a number of demonstration videos available to view, such as: SUMO Traffic within rFpro on a public road test route: https://vimeo.com/191720699

Demonstrating four changes of weather in just 10 seconds, in order to validate ADAS and Autonomous behaviour in adverse conditions: https://vimeo.com/191764582

4.9 Conclusions and Future Work

This brief paper offers an introduction to the work done so far to integrate SUMO with rFpro. There has been great progress already in creating 3D environments with a very high degree of correlation to the real test routes they model. SUMO has made possible the addition of large numbers of real-time 3D rendered, smoothed, vehicles with working brake lights and indicators. SUMO's traffic light control has been integrated to rFpro. The combined system has already proved valuable to a number of existing OEM and Tier one automotive customers.

Through SUMO's choice of open source architecture, when desired features are missing it can be rapidly extended and fed back into the core product to benefit all users. The SUMO developers have a very active support network to facilitate this.

rFpro now delivers, as standard, SUMO road networks for all their public digital road models.

There are many areas that rFpro are looking to improve or implement in the near future:

- Additional object types running in dedicated lanes -e.g. Cyclists, Pedestrians, buses. The cyclist and pedestrian animation has already been developed; the implementation of the SUMO position control for these objects will be an exciting next step.
- Saving and loading of the SUMO states to allow the driver to have "instant start" of their testing scenario, rather than waiting for the traffic to be populated.
- Investigating ways of pre-running SUMO from a saved state or start time until a user's chosen testing start time in the rFpro simulation.
- Providing customers with increased control over their chosen network and scenario definitions from the rFpro Admin Console GUI.

There is one area of SUMO development that would benefit real-time simulation:

- Smooth, natural path lane changes of vehicles that choose to swap lanes. At present rFpro set lanechange.duration to 2 seconds. This produces a semi-natural manoeuvre when the vehicles have a high enough forwards velocity, see figure 4.11. When a vehicle is at a low speed, and especially when at sat at traffic lights, a lane change manoeuvre happens as a slide of the vehicle across the lanes, rather than the vehicle driving forwards into the available space in the next lane, see figure 4.12.

4.10 Bibliography

[1] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3&4):128–138, December 2012.

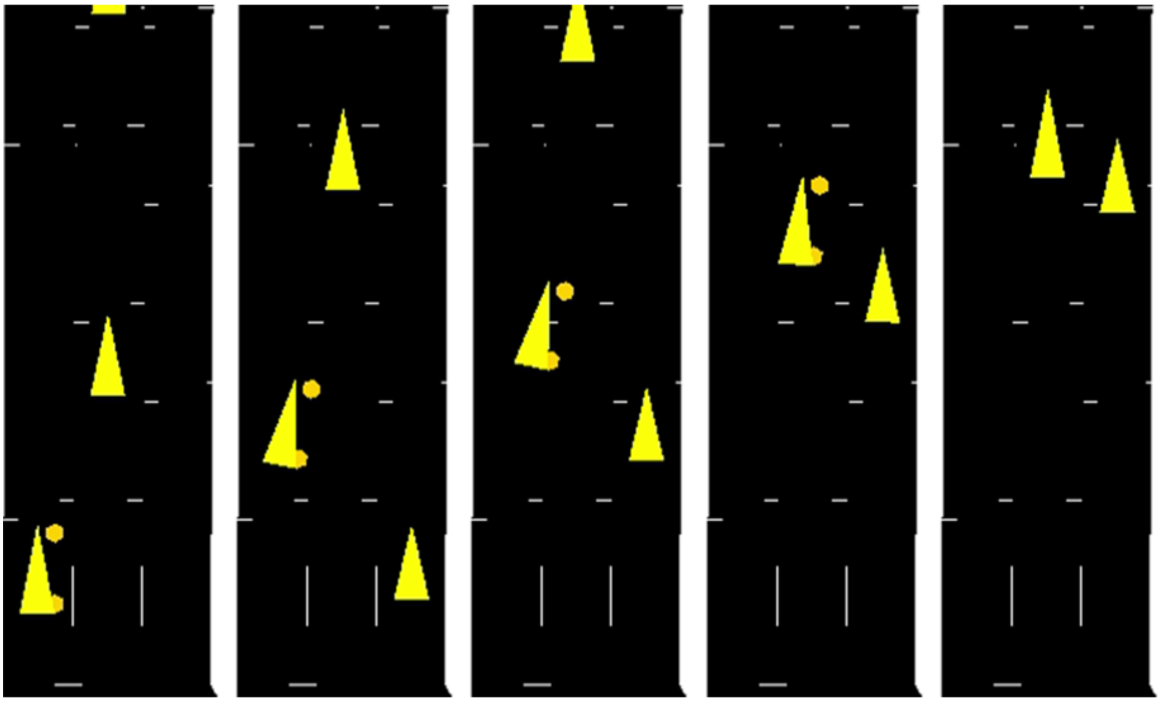


Figure 4.11: A lane change of a vehicle with at speed

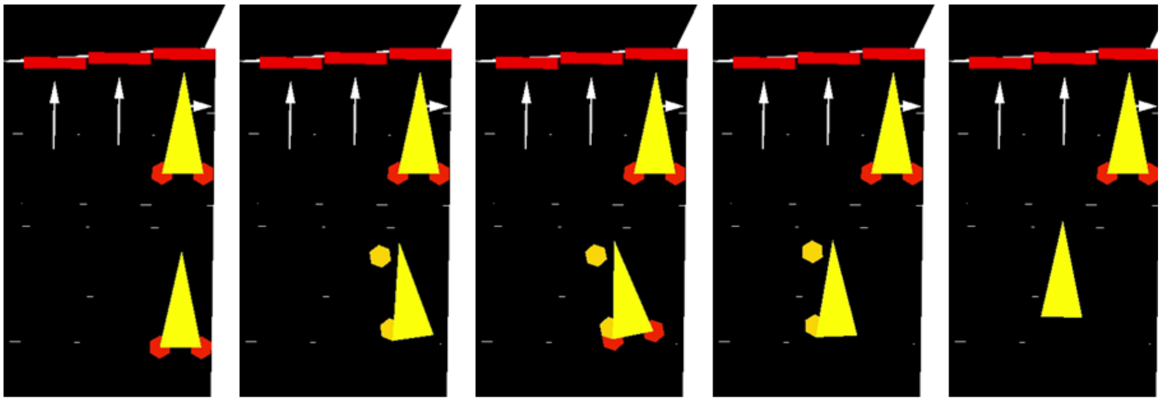


Figure 4.12: A lane change of a vehicle at a set of traffic lights, from stationary.

5 Platooning in SUMO: An Open Source Implementation

Michele Segata; Dept. of Information Engineering and Computer Science, University of Trento, Italy
msegata@disi.unitn.it

5.1 Abstract

Platooning is an application that enhances autonomous driving by means of cooperation. Vehicles, by using wireless communication, exchange data such as position, speed, and acceleration to reduce their inter-vehicle gap and drive in groups, called indeed platoons. The benefits span from improved infrastructure usage to reduced pollution, and from increased safety to a better travel experience. The analysis of a platooning system is complex as it encompasses several research fields, including communication, control theory, traffic engineering, and vehicle dynamics. In addition, even small field operational tests can be extremely expensive in terms of money and time, let alone large scale scenarios. Research studies in platooning can thus highly benefit from simulative analyses, as they overcome the limitations of real world test beds. To this purpose, we developed PLEXE, an extension to the Veins vehicular networking framework that enables the simulation of platooning systems, both from a network and a vehicle dynamics perspective. In this paper we detail the changes PLEXE makes to SUMO to simulate platooning control systems, including realistic engine models for the simulation of vehicle dynamics. In addition, we provide two sample use cases to show its potential.

Keywords: Platooning, Engine Models, Cooperative Adaptive Cruise Control

5.2 Introduction and Background

Autonomous driving has definitely become the hottest topic in automotive in recent years. Self-driving cars can possibly change the way we conceive vehicles today. Besides improving our safety, these vehicles pave the road towards new visionary applications. Autonomous cars do not only relieve the human from driving duties but, for example, we can have the car pick us up in a different place from the one where we left it. Or think about car sharing: usually a user that needs a vehicle needs to fetch the car where the previous user left it. Autonomous vehicles could get rid of this problem.

While self-driving cars can improve safety by avoiding typical human errors such as distractions or drunken driving, they share the same limitation of human beings, i.e., the limited field of view. Autonomous vehicles exploit sensors to collect data and take their driving decisions, but sensors cannot see behind obstacles, just like the human eye. The way to improve the efficiency and the benefits of autonomous vehicles is thus through cooperation: vehicles should share information by means of wireless communication, overcoming the limits of sensors. As an example, take an intersection collision avoidance application. The exchanged information is used to understand whether two vehicles out of each others sight are in collision course (3).

Another example application is platooning(10; 1; 8). With platooning, vehicles share information such as position, speed, and acceleration to form group of cars closely following each other in an autonomous way. The benefits of such an application obviously include safety, but driving at a close distance also reduces the air drag and thus fuel consumption.

The design of platooning systems poses several challenges due to its interdisciplinary nature. The analysis this application is non-trivial, as it requires vehicles, control systems, communication devices, and infrastructure. Simulation-based techniques can thus be highly helpful, but to trust the results the models should be realistic enough. For this reason we developed Plexe (9).

Plexe extends the Veins vehicular networking framework (11) to model platooning systems. The base Veins framework couples the OMNeT++ network simulator with SUMO, providing the research community with a tool that can realistically generate vehicle traces and precisely simulate wireless communication. Plexe takes a step further, implementing platooning control algorithms and detailed engine models within SUMO, enabling platooning studies both from a communication and a vehicle dynamics perspective. Plexe is Open Source and free to download[1].

The requirements of Plexe are

- Not being a new simulator: Plexe extends two well-known network and mobility simulators, so users do not need to learn a new tool;
- Being versatile: Plexe can be used to perform joint network and control analyses;
- Extensibility: users should easily be able to implement additional models within the simulator; and
- Openness: Plexe must be free to download and modify.

In this paper we describe the changes we made to SUMO to realize Plexe, we show two example use cases, and highlight the challenges we encountered.

5.3 Platooning Primer

The fundamental component of platooning is the Cooperative Adaptive Cruise Control (CACC). The CACC is an extended version of a standard Adaptive Cruise Control (ACC) that exploits wireless communication to reduce the inter-vehicle gap. The ACC, instead, relies only on a radar or a lidar. The difference in performance is non-marginal and highlights the gap between selfish, sensor-based vehicles and cooperative vehicles.

As an example, the inter-vehicle gap for an ACC is a constant time headway, i.e., the actual distance is computed as the product of the time headway and the current speed, so the distance increases proportionally with speed. For control stability reasons an ACC must keep a time headway to the front vehicle larger than roughly 1 s to 2 s, which is the same safe distance that must be kept by human drivers (5). By exploiting wireless communication, the CACC developed in (4) is perfectly stable with a time headway of 0.6 s. Each vehicle sends its own "desired" acceleration to the vehicle behind, increasing the reactivity of the CACC. The design in (6), by using data received both from the front and the leading vehicle is stable under a constant spacing policy, i.e., where no time headway is required. In the demonstration, the vehicles drive at a fixed distance of 6 m. For a sensor-based vehicle only, obtaining the information of the leading vehicle is hardly possible.

The principle of operation of a CACC is conceptually simple. The CACC longitudinally controls the vehicle by computing a desired acceleration u that should be applied to maintain a certain inter-vehicle gap. To take its decision the control system obtains data from sensors and via wireless communication: this data can include distance to the front vehicle, position, speed, or acceleration of other vehicles. The acceleration computed by the control system is then given to the lower components for actuation. The engine (or the braking system) requires some time to actuate the

[1] `http://plexe.car2x.org`

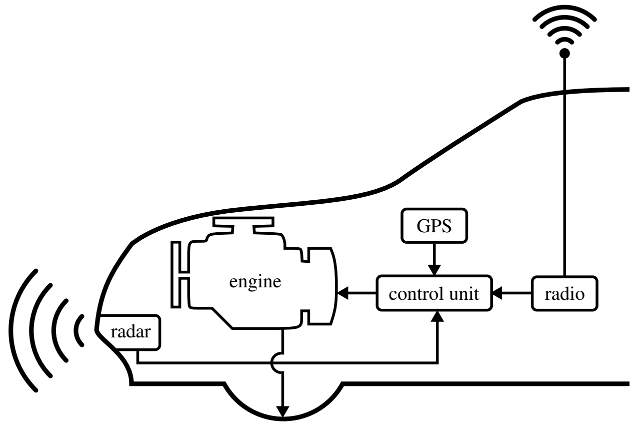


Figure 5.1: Sketch of the control system simulated in Plexe.

command: this time is referred to as actuation lag. Figure 5.1 shows a sketch of the simulated system.

We can model a cruise control as a function $u = C(\boldsymbol{x}, \boldsymbol{r})$, where u is the desired acceleration, $\boldsymbol{x}$ is a set of variables representing the state of the vehicle, and $\boldsymbol{r}$ a set of reference variables. The simplest example is a standard cruise control, i.e., a device that simply maintains a desired speed set by the driver. A very simple control system achieving this goal can be defined as

$$u = C\left(\boldsymbol{x} = \{\dot{x}\}, \boldsymbol{r} = \{\dot{x}_{\mathsf{des}}\}\right) = k_p \left(\dot{x}_{\mathsf{des}} - \dot{x}\right). \tag{5.1}$$

The desired acceleration u is computed as the difference between the desired speed $\dot{x}_{\mathsf{des}}$ (reference value) and the actual speed $\dot{x}$ (state of the vehicle), i.e., the speed error. The speed error is multiplied by a gain k_p, which determines how "reactive" the function is to errors. More sophisticated controllers like an ACC or a CACC follow the same principle, but the sets $\boldsymbol{x}$ and $\boldsymbol{r}$ include additional variables, like acceleration, speed, or position of the own and other vehicles in the platoon.

The desired acceleration u is then passed to engine control for actuation. In control theory nomenclature the equation representing the real system is called the plant. The plant is a function $\ddot{x} = P(u, \boldsymbol{x})$ that computes the real acceleration $\ddot{x}$ given the desired acceleration u and the current vehicle state $\boldsymbol{x}$. One simple but commonly used model is a first order lag (5; 4), which computes the real acceleration by applying a first order low-pass filter to the desired acceleration. Figure 5.2 shows the behavior of such a model.

In a discrete simulator, this is implemented as

$$\ddot{x}_k = P(u, \boldsymbol{x} = \{\ddot{x}_{k-1}\}) = \alpha u + (1 - \alpha)\ddot{x}_{k-1}, \tag{5.2}$$

where

$$\alpha = \frac{\Delta_t}{\tau + \Delta_t}. \tag{5.3}$$

Equation (5.2) computes the acceleration at time step k using the desired acceleration computed by the controller and the acceleration at time step $k - 1$. In Equation (5.3), Δ_t represents the sampling time of the simulator while τ is the engine time constant in seconds, i.e., the delay of the engine. In (5), τ is assumed to be 0.5 s.

This is a simple example, but the engine dynamics can be made more and more realistic by considering further vehicle informations. In Plexe, we implement the first order lag model, as well as a realistic engine model. For brevity, in this paper we do not describe the whole model. The interested reader can refer to (8, Section 2.3) for further details.

The goal of Plexe is to implement cruise control systems and engine dynamics within SUMO. The next section describes how this has been achieved.

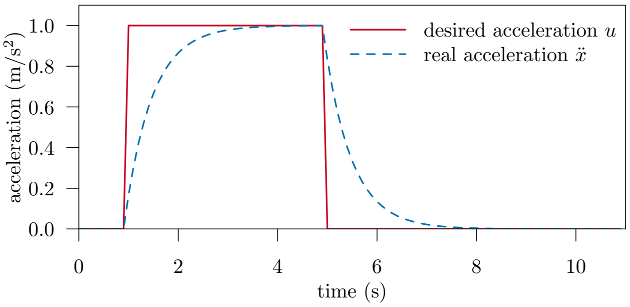


Figure 5.2: Engine actuation lag using a first order lag.

5.4 Implementing Platooning in SUMO

5.4.1 Plexe Structure

This section briefly explains the structure of Plexe to get a general view on how the simulator works. Plexe is built upon Veins, a vehicular networking framework that couples OMNeT++ and SUMO. OMNeT++ takes care of simulating communication between network nodes, while SUMO generate realistic vehicular traffic traces for moving the nodes within the network simulator. Veins, through the TraCI interface, bidirectionally couples the simulators: each vehicle in SUMO has a corresponding network node in OMNeT++, and the node in OMNeT++ moves accordingly. In addition, a node in OMNeT++ can change the behavior of a vehicle in SUMO, for example, by changing the planned route. Veins also offers realistic channel, physical, and medium access layer models for vehicular communication (IEEE 802.11p) (2), as well as higher layer application primitives. Figure 5.3 shows the structure of Veins.

Plexe is built on top of Veins, and in particular it extends both simulators to enable the study of platooning systems. The main changes are applied to SUMO to implement CACCs and engine models. In addition we extend the TraCI interface exploited by Veins for the bidirectional coupling to make the SUMO platooning models accessible within the OMNeT++ modules. The next section describes the implementation details.

5.4.2 Longitudinal Dynamics: A new Car-following Model

The core of the extension is a new car-following model that implements the CACCs, as well as embedding a Krauss car-following model that can "drive" the vehicle when automated controllers are disabled. This enables complex studies where drivers can switch on and off automated driving depending on some conditions. For example, we can imagine a driver to enter the highway, approach a platoon, and only then switch to autonomous driving.

The car-following model is supported by an enriched `VehicleVariables` class. The class handling this data is the `CC_VehicleVariables` class, which stores controller parameters, vehicle state, configuration of the platoon, and user choices.

The specialized car-following model is implemented within the `MSCFModel_CC` class, where "CC" stands for cruise control. The model implements one ACC taken from (5) and three CACCs (6; 4; 7). This is done by keeping in mind that researchers might want to implement and test additional control algorithms, so adding new ones is straightforward.

Algorithm 1 shows the pseudo-code for the new car-following model. In general, SUMO invokes one of the methods that ask the model for the speed to use in the simulation step, for example followSpeed. If the active "driver" is the human-behavioral model, then we simply invoke the same method within the `MSCFModel_Krauss` class. If, instead, the user enables an autonomous controller, the model invokes the method computing the speed to be used in the next simulation step. Within this method, the model computes the desired acceleration for the chosen controller and

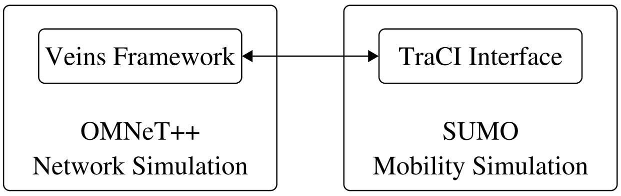


Figure 5.3: Structure of the Veins vehicular networking framework.

then passes this value to the engine model, which considers the actuation lag and computes the real acceleration of the vehicle.

Finally, when SUMO invokes moveHelper to obtain the speed of the vehicle for the current time step, we return the speed value depending on the invoked method. In addition, we update the state of the vehicle.

As SUMO is not designed to simulate driving systems down to the level of detail reached by Plexe, supporting the engine model requires some "hacks". The main problem is that SUMO car-following models are "speed-oriented", i.e., to compute the next speed value each model considers own current speed, front vehicle speed, and distance to the front vehicle.

To compute the next speed value, the engine model requires the current acceleration as well. To account for this requirement, the current acceleration is stored within the `CC_VehicleVariables` class, so the engine model can access it to compute the real acceleration and store the new value. This is, however, not enough as SUMO invokes the model multiple times for each time step. The model should only store the real acceleration value for the actually chosen speed. For example, imagine that SUMO invokes the freeSpeed and the followSpeed methods. The methods will compute their own speed value, each of which is associated with a specific acceleration. If SUMO, for example, uses the new speed computed by followSpeed, then the model should store that specific acceleration value.

This required to track the method invoking the model to store temporary values, and then store only the used ones inside the moveHelper method. Some invocations do not even require storing temporary acceleration values, as they will never be used. One example is the lane-change model that calls the car-following model to understand whether it is worth changing lane. In this case the model needs to know that it is being invoked by the lane changing logic.

The currently available engine models include a first order lag and a realistic engine model. Plexe defines a `GenericEngineModel` class from which all engine models should inherit. The class mainly defines a getRealAcceleration method which, given the current vehicle speed, the current acceleration, and the desired acceleration, computes the acceleration after actuation. The two engine models are implemented into the `FirstOrderLagModel` and the `RealisticEngineModel` classes.

5.4.3 Accessing the Model via TraCI

Users can access the model to enable cruise controllers, set parameters, change and query vehicle status, and so on. This is realized by extending the TraCI interface with a set of APIs. We do not list all the API methods here for the sake of brevity. Further information can be retrieved on the official website.

Besides the standard APIs, we define new TraCI methods to access and easily extend the model. The class defines two generic access points, i.e., the setGenericInformation and the getGenericInformation functions. The idea is avoiding the need to change SUMO core files when extending the simulator. For example, when adding a new CACC we want to be able to pass its parameters via TraCI, but we only want to modify the `MSCFModel_CC` class.

Listing 5.1 shows the prototypes of the two methods used to send data to and retrieve data from the model. TraCI invokes both methods passing the instance of the vehicle for which the action is

Algorithm 1 Pseudo code for the `MSCFModel_CC` class.

```
function followSpeed(vehicle)
    if vehicle.controller == HUMAN then
        return Krauss.followSpeed(vehicle)
    else
        return nextSpeed(vehicle, FOLLOW_SPEED)
    end if
end function
function freeSpeed(vehicle)
    if vehicle.controller == HUMAN then
        return Krauss.freeSpeed(vehicle)
    else
        return nextSpeed(vehicle, FREE_SPEED)
    end if
end function
function nextSpeed(vehicle, invoking)
    $k$ = currentTimestep
    $\boldsymbol{x}$ = vehicle.state
    $\boldsymbol{r}$ = vehicle.reference
    cacc = vehicle.controller
    $u = C(\text{cacc}, \boldsymbol{x}, \boldsymbol{r})$
    engine = vehicle.engineModel
    $\ddot{x} = P(\text{engine}, u, \boldsymbol{x})$
    $\dot{x} = \dot{x}_{k-1} + \ddot{x} \cdot \Delta_t$
    if invoking == FOLLOW_SPEED then
        vehicle.followSpeedTime = $k$
        vehicle.followSpeed = $\{u, \dot{x}\}$
    else
        vehicle.freeSpeed = $\{u, \dot{x}\}$
    end if
    return $\dot{x}$
end function
function moveHelper(vehicle)
    $k$ = currentTimestep
    if vehicle.controller == HUMAN then
        $\dot{x}_k$ = Krauss.moveHelper(vehicle)
        $u_k = 0$
    else
        if vehicle.followSpeedTime == $k$ then
            $\{u_k, \dot{x}_k\}$ = vehicle.followSpeed
        else
            $\{u_k, \dot{x}_k\}$ = vehicle.freeSpeed
        end if
    end if
    $\ddot{x}_k = (\dot{x}_k - \dot{x}_{k-1})/\Delta_t$
    vehicle.state = $\{u_k, \ddot{x}_k, \dot{x}_k\}$
    return $\dot{x}_k$
end function
```

requested. The CCDataHeader structure includes information about the set or get operation. The type indicates the operation, for example the actual parameter the user wants to set, while size indicates number of bytes inside the content (for the setter) or inside the parameters of the request (for the getter). In the case of the setter, the content can be anything, from a basic type such as an integer to a complex data structure.
As an example, if we want to set an integer parameter, size will be set to sizeof(int), while the value can be retrieved by casting the content to an integer.
Concerning the getter, reqParams can be used to parameterize the request. The content is where the method stores the response which, as for the setter, can be of any kind. The method should return the size of the response in bytes.
These two methods provide the users with a straightforward way of extending the communication with the model using the TraCI interface.

5.4.4 Lane Change Model

In addition to the car-following model for longitudinal dynamics, Plexe extends SUMO by adding a lane changer. The aim of the extension is to enable users to tell platooning vehicles on which lane to drive, regardless of what the lane change model dictates. The logic is defined within the MSCACCLaneChanger class, which extends the basic MSLaneChanger provided by SUMO. The MSEdge class instantiates the new MSCACCLaneChanger instead of MSLaneChanger. The new lane changer overrides the methods of the base class, disabling standard behavior when required by the user.
By invoking the TraCI methods in Listing 5.2 it is possible to tell the vehicle to perform a certain action. With setFixedLane the vehicle will move to the required lane as soon as this can safely be done, meaning that the vehicle will avoid collisions with other vehicles. With setLaneChangeAction, instead, we specify what the vehicle should do with respect to the lane change logic. By using DRIVER_CHOICE, the vehicle behaves using standard SUMO lane change logic, i.e., as if a human driver would be in control. With STAY_IN_CURRENT_LANE the vehicle is forced to drive on the current lane, while with MOVE_TO_FIXED_LANE the vehicle tries to change to the lane specified via setFixedLane.

5.5 Use Cases

In this section we show some usage examples to highlight the potential of the simulator. In the first use case, we test different parameterizations of the engine models in an accelerate and brake scenario. In the second use case, we briefly show the differences between two control algorithms implemented in Plexe for a 4-vehicle platoon.

5.5.1 Engine Models

We use a standard cruise control of the form

$$u = \min\left(5\,\mathrm{m\,s^{-2}}, \max\left(-7\,\mathrm{m\,s^{-2}}, \dot{x}_{\mathsf{des}} - \dot{x}\right)\right), \tag{5.4}$$

i.e., a simple controller that maintains a desired speed. The controller never asks the engine (or the braking system) to accelerate stronger than $5\,\mathrm{m\,s^{-2}}$ and to decelerated harder than $7\,\mathrm{m\,s^{-2}}$. These limits are not realistic, but we use them for demonstration purposes only. In the scenario we use a single vehicle that enters the road at a speed of $80\,\mathrm{km\,h^{-1}}$. After one second, the desired speed $\dot{x}_{\mathsf{des}}$ is increased to $130\,\mathrm{km\,h^{-1}}$ and then reset to $80\,\mathrm{km\,h^{-1}}$ after additional ten seconds. We test the behavior of the vehicle in five different configurations. The first two use the first order lag model

```
void setGenericInformation(
  const MSVehicle* veh,
  const struct CCDataHeader &header,
  const void *content
);

int getGenericInformation(
  const MSVehicle* veh,
  const struct CCDataHeader request,
  const void *reqParams,
  void *content
);

struct CCDataHeader {
  int type;
  int size;
};
```

Listing 5.1: Access methods to the `MSCFModel_CC` class.

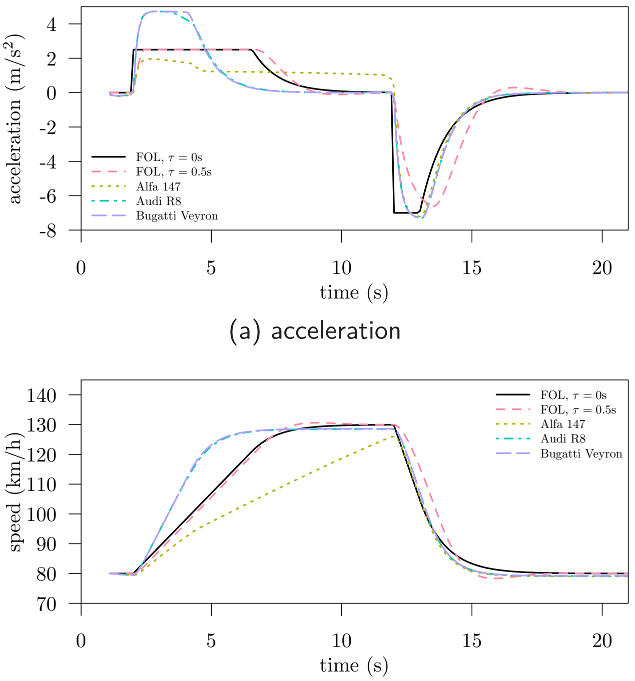
```
void setFixedLane(
  int laneIndex
);

void setLaneChangeAction(
  enum PLATOONING_LANE_CHANGE_ACTION action
);

enum PLATOONING_LANE_CHANGE_ACTION {
  DRIVER_CHOICE,
  STAY_IN_CURRENT_LANE,
  MOVE_TO_FIXED_LANE
};
```

Listing 5.2: Lane changing methods.

with an engine time constant τ of 0 s and 0.5 s. When using this model, we have an acceleration limit of 2.5 m s^{-2}. The remaining three use the realistic engine model, parameterized to reproduce the engine of an Alfa Romeo 147, an Audi R8, and a Bugatti Veyron. The engine parameters for the vehicles are available in the PLEXE repository and are not listed here for brevity.

Figure 5.4 shows the results for the engine test scenario in terms of real acceleration $\ddot{x}$ and speed $\dot{x}$ as function of time. The first noticeable difference is in the behavior of the first order lag model for different values of τ. For $\tau = 0$ s the reaction of the vehicle to commands is instantaneous. The vehicle immediately performs what the controller requires to do, which is clearly unrealistic. By using a time delay $\tau = 0.5$ s the reaction is smoother, and we observe the delay in the actuation, both during the acceleration and in the deceleration phase.

For the realistic engine models each vehicle has its own characteristics. Clearly, as for the first order lag model with $\tau = 0.5$ s, the engine (and braking) reaction is not immediate. Depending on the vehicle we can observe different performance. First of all, we can observe than only the two high-end cars are capable of delivering the required maximum acceleration of 5 m s^{-2}. The Alfa



(a) acceleration

(b) speed

Figure 5.4: Vehicle dynamics for the engine models test.

Romeo, depending on the current speed, reaches a maximum acceleration of 2 m s^{-2}, which drops down to roughly 1 m s^{-2} at 5 s due to gear shifting. In addition, the maximum acceleration slowly decreases over time due to air drag. In the end, the Alfa Romeo is not capable of reaching the target speed before the braking action starts.

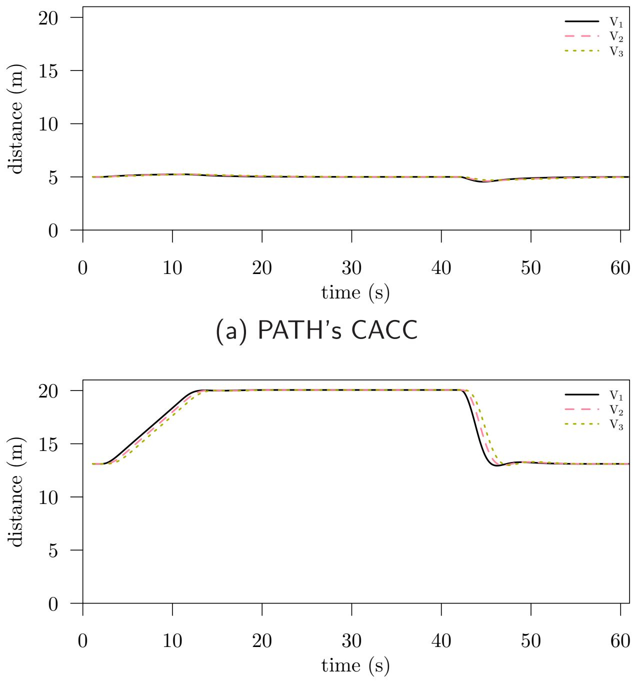
This example shows how we can realistically simulate different vehicle behaviors, which is fundamental in the study of platooning systems. Indeed, one interesting question is how a control system performs when the platoon is made by a set of inhomogeneous vehicles. When such a level of modeling detail is not required, the user can switch to the first order lag model, which, although more simplistic, provides a trade off between a completely unrealistic and a realistic behavior.

5.5.2 Control System Analysis

In this analysis we compare the PATH's CACC (6) and the Ploeg's CACC (4), which are two of the available CACCs. They are well-known CACC algorithms that can be found in the literature, which have been experimentally tested. We choose them for demonstration purpose, but users can choose other algorithms or easily implement their own.

We test a 4-vehicle platoon traveling on a highway. The leader (referred to as V$_0$) is driven by a standard cruise control as in Section 5.5.1, while the three followers (referred to as V$_1$, V$_2$, V$_3$) are controlled by one of the two CACCs. The scenario is similar to the one in Section 5.5.1, with the only difference that the cruise control desired acceleration u is limited to 1.5 m s^{-2}. In addition, after increasing the desired speed to 130 km h^{-1}, the leader maintains this speed for a longer period of time before decreasing it back to 80 km h^{-1}.

The two controllers considered have two different spacing policies. The PATH's CACC uses a constant spacing policy, meaning that the inter-vehicle gap is independent of the speed. In the simulation we set the spacing $d_{\text{des}} = 5$ m. The Ploeg's CACC uses a constant headway policy, i.e., the inter-vehicle gap is $d_{\text{des}} = h \cdot \dot{x}$, for a given headway time h. We set $h = 0.5$ s, which, for the two chosen cruising speeds of 80 km h^{-1} and 130 km h^{-1}, translates into an inter-vehicle distance between 13 m to 20 m,



(a) PATH's CACC

(b) Ploeg's CACC

Figure 5.5: Distance comparison between two CACC algorithms.

including standstill distance[2].

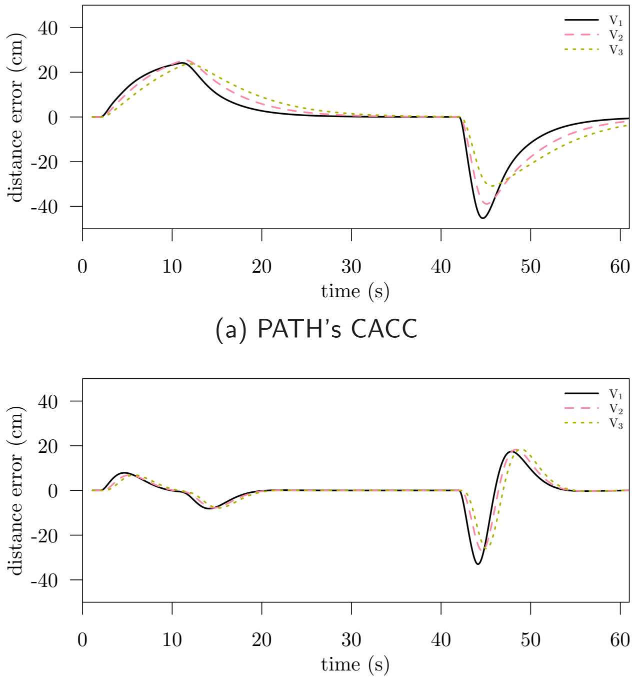
In both cases the engine is modeled using a first order lag with $\tau = 0.5$ s. Finally, the vehicles exchange data by sending wireless broadcast messages ten times a second (the packet transmission rate is 10 Hz). Clearly, the message exchange rate has an impact on the performance of the controller and, if necessary, Plexe can be employed to analyze such an impact. Here we only focus on the SUMO modeling, so we disregard communication aspects.

Figure 5.5 shows the performance of the two CACCs in terms of inter-vehicle distance as function of time. In the graphs, V_i indicates the distance between vehicle V_i and vehicle V_{i-1}. The graphs clearly show the difference in terms of spacing policies. While the PATH's CACC maintains a constant gap around the target of 5 m, Ploeg's CACC adapts the distance to the cruising speed.

It is interesting to compare the controllers in terms of distance error, i.e., how much the actual distance deviates from the target distance. Figure 5.6 plots the distance error in centimeters computed as $d - d_{\text{des}}$. When the error is positive, the vehicle is farther to its front vehicle than it should be. When the error is negative, the vehicles is instead too close. The graphs shows that the PATH's CACC induces a larger distance error, meaning that the constant headway spacing policy improves tracking performance. However, Ploeg's CACC induces some overshooting, while the PATH's CACC smoothly converges to the target distance.

The aim of this section is not decide which control system performs better, but to highlight the potential of the simulator. Using Plexe it is straightforward to perform comparative analysis of platooning control systems. In addition, users can easily extend the simulator to implement new control strategies and compare them with the state of the art controllers already available in Plexe. The official website includes a detailed tutorial showing how this can be done in a few simple steps.

[2] The implementation considers a stand-still distance of 2 m, to avoid the distance to become null in case of a complete stop.



(a) PATH's CACC

(b) Ploeg's CACC

Figure 5.6: Distance error comparison between two CACC algorithms.

5.6 Conclusion

This paper describes in detail how SUMO has been extended within the Plexe framework to implement platooning control systems and detailed engine models. As platooning is becoming more and more a hot topic in the scope of automated driving and vehicular networking, there is the need of a realistic simulator to evaluate this innovative driving solution. Plexe achieves this goal by extending SUMO in an non-invasive way, and permits the user to further modify it to tune the simulator to specific purposes. The paper described the implementation in detail to help users understanding how the extension works. Furthermore, we have seen its potential for two sample use cases, i.e., to observe different vehicle behaviors depending on the chosen engine model and to compare platooning control strategies. The SUMO extension in Plexe provides a valid, open, and free tool for the analysis of future driving technologies.

5.7 Bibliography

[1] Bergenhem, C., Q. Huang, A. Benmimoun and T. Robinson: *Challenges of Platooning on Public Motorways*. In *17th World Congress on Intelligent Transport Systems (ITS 2010)*, Busan, Korea, October 2010.

[2] Eckhoff, D. and C. Sommer: *A Multi-Channel IEEE 1609.4 and 802.11p EDCA Model for the Veins Framework*. In *5th ACM/ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2012), Poster Session*, Desenzano, Italy, March 2012. ACM.

[3] Joerer, S., M. Segata, B. Bloessl, R. Lo Cigno, C. Sommer and F. Dressler: *A Vehicular Networking Perspective on Estimating Vehicle Collision Probability at Intersections*. IEEE Transactions on Vehicular Technology, 63(4):1802–1812, May 2014.

[4] Ploeg, J., B.T.M. Scheepers, E. van Nunen, N. van de Wouw and H. Nijmeijer: *Design and Experimental Evaluation of Cooperative Adaptive Cruise Control*. In *IEEE International Conference on Intelligent Transportation Systems (ITSC 2011)*, pages 260–265, Washington, DC, October 2011. IEEE.

[5] Rajamani, R.: *Vehicle Dynamics and Control*. Springer, Second edition, 2012.

[6] Rajamani, R., H.-S. Tan, B. K. Law and W.-B. Zhang: *Demonstration of Integrated Longitudinal and Lateral Control for the Operation of Automated Vehicles in Platoons*. IEEE Transactions on Control Systems Technology, 8(4):695–708, July 2000.

[7] Santini, S., A. Salvi, A. S. Valente, A. Pescapè, M. Segata and R. Lo Cigno: *A Consensus-based Approach for Platooning with Inter-Vehicular Communications*. In *34th IEEE Conference on Computer Communications (INFOCOM 2015)*, pages 1158–1166, Hong Kong, China, April 2015. IEEE.

[8] Segata, M.: *Safe and Efficient Communication Protocols for Platooning Control*. PhD Thesis (Dissertation), University of Innsbruck, February 2016.

[9] Segata, M., S. Joerer, B. Bloessl, C. Sommer, F. Dressler and R. Lo Cigno: *PLEXE: A Platooning Extension for Veins*. In *6th IEEE Vehicular Networking Conference (VNC 2014)*, pages 53–60, Paderborn, Germany, December 2014. IEEE.

[10] Shladover, S.E.: *PATH at 20 – History and Major Milestones*. In *IEEE Intelligent Transportation Systems Conference (ITSC 2006)*, pages 22–29, Toronto, Canada, September 2006.

[11] Sommer, C., R. German and F. Dressler: *Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis*. IEEE Transactions on Mobile Computing, 10(1):3–15, January 2011.

6 Use of SUMO to assess human exposure and intake to $PM_{2.5}$ near motorways: Preliminary Results

Iván Y. Hernández-Paniagua, Juan C. Zavala-Reyes, Pablo López-Ramírez, Ulises Diego-Ayala, Irma Rosas, Arón Jazcilevich; Centro de Ciencias de la Atmosfera, Universidad Autónoma de México
ivan.hernandez@atmosfera.unam.mx

6.1 Abstract

SUMO is used to estimate exposure of virtual pedestrians to vehicular $PM_{2.5}$ emissions. The pedestrians simulated by SUMO are considered to be performing physical activities such as waiting, walking, commuting, running and cycling. Respiratory $PM_{2.5}$ intake was estimated for each activity by integrating SUMO with a pollution transport and an intake model. Time of residence over the affected area of the footpath resulted in the dominant factor determining intake loading. Therefore, cycling activity resulted in the minimum $PM_{2.5}$ intake (514 ± 68 μg), while waiting resulted in the maximum $PM_{2.5}$ intake (1406 ± 143 μg).

Keywords: Emissions, intake, pedestrians, simulation, SUMO.

6.2 Introduction

The traffic model Simulation of Urban Mobility (SUMO) allows the representation of interaction between vehicles and pedestrians through computer simulations (9). This feature permits investigating population behaviour that not only can be applied to urban transport planning, but also to estimate pedestrian exposure and pollutants intake near motorways, as shown here. This can be achieved by integrating to SUMO, a pollution transport model and a basic physiological intake model. A special environmental concern are $PM_{2.5}$ emissions from motor vehicle emissions, since they represent a threat to population health. Epidemiological studies have shown associations between personal exposure to elevated concentrations of $PM_{2.5}$ and respiratory affections such as increased risk of breathlessness attacks, regular phlegm and wheezing problems (1). Also, population proximity to $PM_{2.5}$ emissions has been correlated with worsening neurological effects (3). A study of respiratory health in adult residents in Mexico City carried out by Cortez-Lugo et al. (4) reported an increase of 33% in cough, and 23% in phlegm ascribed to an increase of 10 μg m^{-3} in the daily average during a two-day personal exposure to $PM_{2.5}$. This suggests that population directly exposed to $PM_{2.5}$ sources are likely to experience adverse respiratory health effects.

Studies on population exposure to $PM_{2.5}$ have been traditionally made using city scale modelling based on continuous area monitoring data and personal measurements over particularly sites or trip routes. Smith et al. (12) estimated population exposure to $PM_{2.5}$ accounting for individual movements during the day in London, with average $PM_{2.5}$ exposures between 8.1 and 8.8 μg m^{-3} for child (5-17 years old) and young adults (18-29 years old), respectively. Although, they observed that the largest exposure periods during the day occurred at indoor environments, the exposure during

daily commute (driving, walking, cycling etc.) was significant enough to be considered. Okokon et al. (10) reported personal exposure averages to $PM_{2.5}$ at 3 European cities, comparing bus, private cars and cycling. However, they did not take into account queuing exposure periods, which, in megacities with large population, may increase exposure. Moreover, physical activities performed along streets with high traffic load may increase accumulated daily personal exposure that is accentuated during rush hours (6).
In this study, we present estimations of personal $PM_{2.5}$ intake at an adjacent street to the campus of the National Autonomous University of Mexico (UNAM), during daily commuting activities. This is accomplished by using SUMO and a fine scale pollution transport model called Mass Conservation and Advection Diffusion (MCAD) by Kanda et al. (8). Pedestrian dynamics data and motor vehicle emissions were obtained using SUMO. Street level $PM_{2.5}$ concentration fields were determined by transporting vehicle emissions using the MCAD. $PM_{2.5}$ intake estimations were calculated from the exposure periods determined with SUMO and population physiological parameters reported in the literature.

6.3 Methodology

6.3.1 Modelling outline

Figure 6.1 shows the proposed approach layout to estimate $PM_{2.5}$ intake. It consists of connecting a model output to the input of next model.

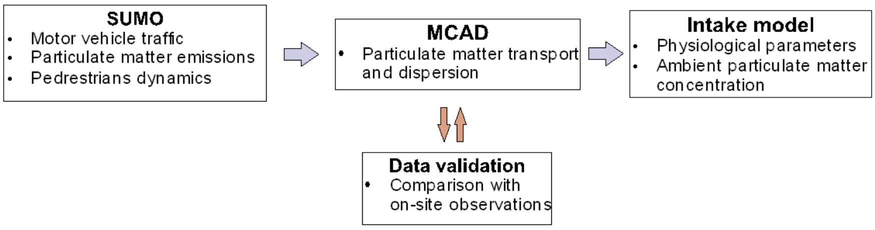

SUMO
• Motor vehicle traffic
• Particulate matter emissions
• Pedrestrians dynamics
MCAD
• Particulate matter transport and dispersion
Intake model
• Physiological parameters
• Ambient particulate matter concentration
Data validation
• Comparison with on-site observations

Figure 6.1: Model system layout: Pedestrian dynamics and $PM_{2.5}$ emissions are obtained with SUMO, street-level $PM_{2.5}$ concentrations are determined with MCAD, and personal intake is estimated using a proposed preliminary intake model.

Based on Fig. 6.1, the following steps are followed to estimate physiological $PM_{2.5}$ intake values:

i). Obtaining the interaction of pedestrians and vehicles.

This can be modelled in SUMO by extending vehicle lanes with footpaths, and providing pedestrians with crossings at the intersections. Although the pedestrian model in SUMO is better suited to work as part of a greater multi-modal simulation while pedestrian flows are calculated from a distribution of person trips (where a person will use several modes of transportation), it is also possible to define individual pedestrian routes. This allows for fine-grained control of simulations at very high spatial and temporal resolution, without the requirement of modelling vehicle and pedestrian demands at a larger scale. In this way, SUMO can be used to evaluate different network designs at very small scale, such as a single pedestrian crossing. As shown later, the emission model available in SUMO can be coupled with a diffusion model to assess pollutant concentrations at different distances from the motorway.
Within SUMO, pedestrians move along special lanes defined as footpaths and may cross between streets, when a pedestrians crossing is available. Footpaths are defined in XML base files containing existing networks that can be modified by user. The pedestrian trips are defined considering attributes

such as type of pedestrian, initial and final edges, etc. The output file for each pedestrian may contain speed and location for any given edge along the route defined in the input (DLR, 2017). For our study site, the pedestrian trips were generated as southwards and northwards trips over footpaths next to the motorway of interest as described in sect. 6.3.2. This was applied for several person profiles, representative of a pedestrian population.

ii). Estimating vehicular emissions.

These can be obtained from on-road measurements near given locations by using portable emission instrumentation (7). SUMO allows the estimation of CO2, CO, HC, NOx, and PM emissions for a large number of motor vehicles classes using HBEFA emission factors (2). Here we focus on $PM_{2.5}$ emissions due to their higher risk factor.

iii). Obtaining near motorway $PM_{2.5}$ concentrations.

The MCAD model is used to determine the transport of $PM_{2.5}$ vehicular emissions, incorporating local meteorology and urban morphology. Briefly, MCAD uses the mass conservation (MC) principle to calculate the velocity field and the advection-diffusion equation (AD) to represent concentration fields, even with complicated geometries. Properties of the atmospheric flow in the turbulent layer due to the obstacles such as buildings were determined experimentally in a wind tunnel, assuming neutral atmospheric stability (8).

iv).Once $PM_{2.5}$ street-level concentrations are determined

exposure periods are estimated by obtaining from SUMO the necessary time for pedestrians to execute a given physical activity (walking, running, etc.).

v). Using the exposure periods

$PM_{2.5}$ intake along the given route is estimated as the integrated exposure of ventilation multiplied by the exposure period in a given microenvironment, and by the concentration of $PM_{2.5}$ at such site. The proposed intake model includes physical and physiological data, such as ventilation rates and speed-physical activity relationships (5) (13).
This procedure was applied to the specific case described below. Preliminary results are presented.

6.3.2 Study site

Pedestrian dynamics were simulated over footpaths of the DelfÃn Madrigal major motorway, adjacent to the UNAM (National Autonomous University of México) campus in the south of Mexico City (Fig. 6.2). The Delfín Madrigal motorway has footpaths on both sides and serves as one of the main access to UNAM for vehicles. It is located beneath two footbridges that connect the University metro station with the main bus station of UNAM as shown in Fig. 6.2 The pedestrian crossing is located southwards just after the first bump.

6.3.3 Vehicular emissions and $PM_{2.5}$ dispersion modelling

The spatial variations of $PM_{2.5}$ at street level were obtained with MCAD (8), considering only local particle emissions from motor vehicles. $PM_{2.5}$ vehicular emissions were obtained using SUMO for traffic flows of 10, 25 and 37 vehicles min^{-1}, with a car/truck ratio of 0.7. Initial wind speed was considered as 2 m s-1 with a prevailing east to west wind occurrence, and a reference height of 20 m determined from the meteorology station, whose location is shown in Fig. 6.2. Urban morphology

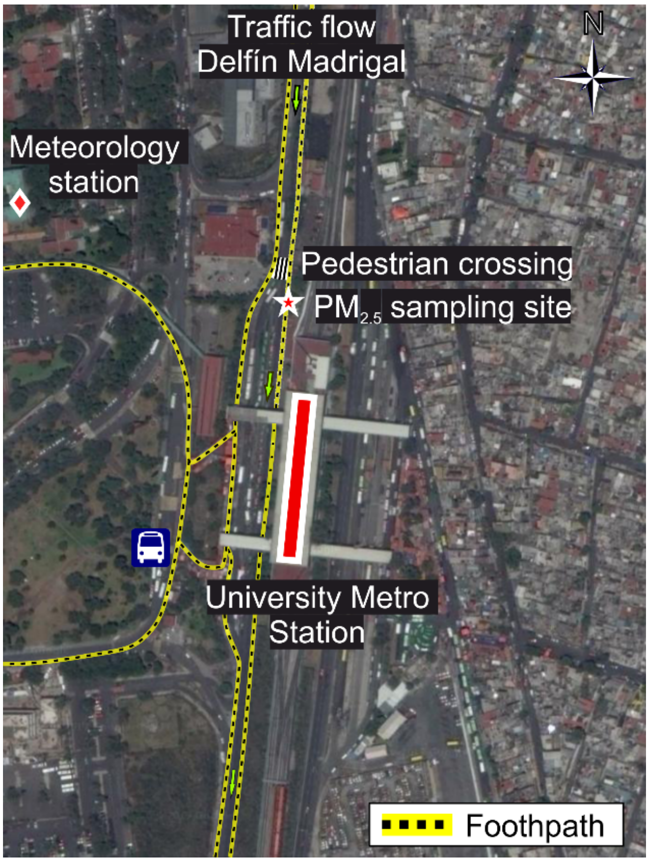


Figure 6.2: Study area for pedestrian dynamics, exposure and $PM_{2.5}$ intake in the south of Mexico City close to UNAM campus. Image source: 2007 LIDAR database, collected by the National of Statistics and Geography[13].

data were obtained from the 2007 LIDAR database, collected by the National Institute of Statistics and Geography (11).

6.4 Results

6.4.1 Exposure periods and $PM_{2.5}$ emissions

Five personal profiles were simulated in SUMO for 5 physical activities performed along selected routes considered representative of daily commuting. Table 3.1 shows the values for the SUMO speed attribute used to estimate the exposure. The highest speeds of 8 km h^{-1} and 13 km h^{-1} were observed for cycling and running, respectively. Those activities were performed by male adults. SUMO estimated the shortest exposure periods for cycling and male adults (114 s) (Table 6.1). The longest periods were those for walking and children (6 years old) (295 s).
The average vehicle flow (vehicles min^{-1}) were determined during a field campaign carried out during 25-27 March 2015. Vehicles were counted and distributed in the following classes: private car, city bus, microbus, vans, taxis and public service trucks. Overall, the largest traffic flow was 37 vehicles min^{-1} during the morning and evening rush hours, and the lowest was 10 vehicles min^{-1} around mid-day. For the rest of the day, 25 vehicles min^{-1} in average were observed. Figure 6.3 shows the corresponding average emission profiles for the traffic flows. SUMO shows that the largest emission peak occurs when the traffic flow is lowest. However, the overall emissions are larger for higher traffic flows in response to larger periods spent within the studied section of the motorway.
Figure 6.4 shows the $PM_{2.5}$ MCAD dispersion profile at street-level of vehicle emissions on the studied segment of the Delfín Madrigal Street. Dominant winds were from East to West. Using the

Table 6.1: Personal and physical activities simulated in SUMO along the selected

Person Activity speed (km h^{-1})	Walk	Commute	Run	Cycle
Adult male	3.5	5	8	13
Adult female	3	4	7	11
Average adult	3	4	7	11
Child age 10	2.5	2.5	4	7
Child age 6	2	2	3	5

Table 6.2: Exposure periods determined with SUMO for pedestrians moving along the Delfín Madrigal motorway.

Person Exposure time (s)	Walk	Commute	Run	Cycle
Adult male	398	221	160	114
Adult female	423	236	176	130
Average adult	409	236	168	125
Child age 10	431	279	195	150
Child age 6	460	295	209	174

intake model, Table 6.3 shows the $PM_{2.5}$ intake estimations for an average adult while performing physical activities over footpaths of the studied motorway. Because of the exposure time, the highest $PM_{2.5}$ intake values were obtained for the waiting activity and the lowest for cycling. This was the case for the motorway segment used for this study.

Table 6.3: $PM_{2.5}$ intake estimations for average adults performing different physical activities along the Delfín Madrigal motorway with 3 different motor vehicle flows. Average intake (μg)

Traffic flow (vehicle min^{-1})	Waiting	Walking	Commuting	Running	Cycling
10	471 ± 53	284 ± 74	241 ± 63	133 ± 45	178 ± 45
25	642 ± 39	607 ± 52	571 ± 68	456 ± 71	410 ± 33
37	1406 ± 143	1326 ± 104	895 ± 95	791 ± 66	514 ± 68

Table 6.4 shows the increase in $PM_{2.5}$ intake for an average adult expressed as percentage, considering 10 vehicles min^{-1} as a reference base. The highest increase in $PM_{2.5}$ intake of 595% was observed for running and 37 vehicles min^{-1}. However, it should be noted that for adults conducting other physical activities, such as commuting or cycling, the predicted increase in average $PM_{2.5}$ intake is similar to the increase in traffic, implying that only during some physical activities there is an effect caused by increase in traffic. This is likely due to the emission dispersion at the study site, which affects differently depending on the height and coordinate in the 3D scenario. These results suggest that among the physical activities studied, running represents the highest health risk in a crossing.

6.5 Conclusion and future work

SUMO was used to estimate pedestrian exposure time to $PM_{2.5}$ vehicle emissions while performing typical physical activities on footpaths of a heavy traffic motorway in Mexico City. $PM_{2.5}$ intake was estimated accordingly. Residence on the affected area resulted in the dominant factor to determine

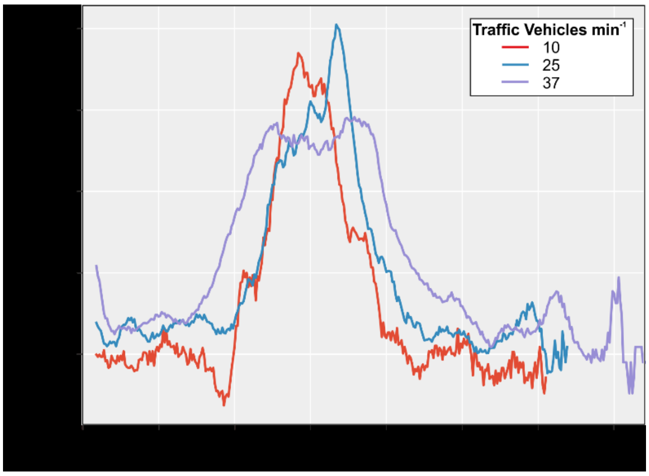


Figure 6.3: SUMO average vehicle $PM_{2.5}$ emissions for the 3 traffic simulated flows at the study site. The peak in the emissions occurs during the acceleration stage once vehicle passed the bump.

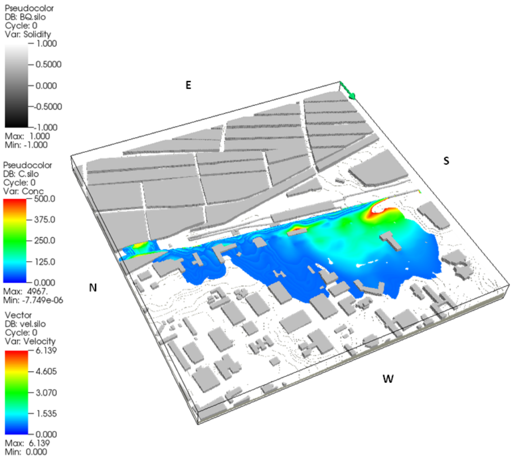


Figure 6.4: $PM_{2.5}$ concentration distribution at 2 m height (centre of domain: north-south). The building blocks are show in grey. Dominant wind direction is shown by green arrow at the top of figure.

$PM_{2.5}$ intake loads. Although, results presented here were obtained for the studied motorway segment, the proposed methodology can be applied for entire motorways and larger urban sections. Further refinements to the system, including more detailed dispersion and vehicular emission models, are currently under development.

6.6 Acknowledgements

This research was supported by the Secretaría de Ciencia y Tecnología de la Ciudad de México, SECITI CDMX 103. The authors acknowledge Dr. Isao Kanda for his assistance in running MCAD, Dra. Patricia Segura for providing comments and physiological information and, Jose Manuel Hernandez, Manuel Garcia Espinosa, Wilfredo Gutierrez Lopez and Alfredo Rodriguez Manjarrez for their technical assistance.

Table 6.4: Percentage of increase in traffic flow and in the average $PM_{2.5}$ intake for the physical activities shown in Table 6.3 Average intake (μg)

Traffic flow (vehicle min^{-1})	Waiting	Walking	Commuting	Running	Cycling
10			Baseline		
250%	136%	214%	237%	343%	230%
370%	299%	467%	371%	595%	289%

6.7 Bibliography

[1] Bayer-Oglesby, Lucy, Christian Schindler, Marianne E. Hazenkamp-von Arx, Charlotte Braun-Fahrländer, Dirk Keidel, Regula Rapp, Nino Künzli, Otto Braendli, Luc Burdet, L-J Sally Liu, Philippe Leuenberger and Ursula Ackermann-Liebrich: *Living near Main Streets and Respiratory Symptoms in AdultsThe Swiss Cohort Study on Air Pollution and Lung Diseases in Adults*. American Journal of Epidemiology, 164(12):1190, 2006.

[2] Bieker, Laura, Daniel Krajzewicz and Andreas Leich: *Local Emissions Monitoring using vehicular communication*. Proceedings of the 2016 SUMO User Conference, pages 165–173, 2016.

[3] Chen, Hong, Jeffrey C Kwong, Ray Copes, Karen Tu, Paul J Villeneuve, Aaron van Donkelaar, Perry Hystad, Randall V Martin, Brian J Murray, Barry Jessiman, Andrew S Wilton, Alexander Kopp and Richard T Burnett: *Living near major roads and the incidence of dementia, Parkinson's disease, and multiple sclerosis: a population-based cohort study*. The Lancet, 389(10070):718 – 726, 2017.

[4] Cortez-Lugo, Marlene, Matiana Ramírez-Aguilar, Rogelio Pérez-Padilla, Raúl Sansores-Martínez, Alejandra Ramírez-Venegas and Albino Barraza-Villarreal: *Effect of Personal Exposure to PM2.5 on Respiratory Health in a Mexican Panel of Patients with COPD*. International Journal of Environmental Research and Public Health, 12(9):10635–10647, 2015.

[5] Gratt, Lawrence B: *Air toxic risk assessment and management: Public health risk from normal operations*. John Wiley & Sons, 1996.

[6] Hertel, Ole, Martin Hvidberg, Matthias Ketzel, Lars Storm and Lizzi Stausgaard: *A proper choice of route significantly reduces air pollution exposure—a study on bicycle and bus trips in urban streets*. Science of the total environment, 389(1):58–70, 2008.

[7] Jazcilevich, Aron, José María Mares Vázquez, Pablo López Ramírez and Irma Rosas Pérez: *Economic-environmental analysis of traffic-calming devices*. Transportation Research Part D: Transport and Environment, 36:86–95, 2015.

[8] Kanda, Isao, Toshimasa Ohara, Taro Nataami, Hiroshi Nitta, Kenji Tamura, Shuichi Hasegawa, Masayuki Shima, Satoshi Nakai, Kazuhiko Sakamoto and Hisashi Yokota: *Development of outdoor exposure model of traffic-related air pollution for epidemiologic research in Japan*. Journal of Exposure Science and Environmental Epidemiology, 23(5):487–497, 2013.

[9] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3&4):128–138, December 2012.

[10] Okokon, Enembe O, Tarja Yli-Tuomi, Anu W Turunen, Pekka Taimisto, Arto Pennanen, Ilias Vouitsis, Zissis Samaras, Marita Voogt, Menno Keuken and Timo Lanki: *Particulates and noise exposure during bicycle, bus and car commuting: A study in three European cities*. Environmental Research, 154:181–189, 2017.

[11] Silván-Cárdenas, José Luis, Juan Andrés Almazán-González and Stéphane A Couturier: *Remote Identification of Housing Buildings with High-Resolution Remote Sensing*. In *Mexican Conference on Pattern Recognition*, pages 380–390. Springer, 2014.

[12] Smith, James David, Christina Mitsakou, Nutthida Kitwiroon, Ben M. Barratt, Heather A. Walton, Jonathon G. Taylor, Hugh Ross Anderson, Frank J. Kelly and Sean D. Beevers: *London Hybrid Exposure Model: Improving Human Exposure Estimates to NO2 and PM2.5 in an Urban Setting*. Environmental Science & Technology, 50(21):11760–11768, 2016. PMID: 27706935.

[13] West, John Burnard: *Respiratory physiology: the essentials*. Lippincott Williams & Wilkins, 2012.

7 SciTraffic: A Macroscopic Simulator for Estimation and Control of Urban Traffic

Christian Portilla, Andrés Acosta and Jairo Espinosa; Universidad Nacional de Colombia
{crportil, afacostag, jespinov}@unal.edu.co
Jorge Espinosa; Politécnico Colombiano Jaime Isaza Cadavid
jeespinosa@elpoli.edu.co

7.1 Abstract

SUMO is a powerful microsimulation package useful for modeling, evaluating and managing urban traffic networks. One of the main advantages of this package is its capacity to integrate with external applications through the TraCI API, allowing to develop and test strategies for managing these traffic networks, where the typical objective consists in minimizing congestion, usually through adaptive traffic lights control. From a control perspective, SUMO can act as the plant to be controlled and the external application as the control system. Model-based Control and specifically Model Predictive Control (MPC) has many advantages over other control strategies, including the explicit consideration of constraints and the fact that it utilizes an explicit model of the plant in order to predict its behavior. This article presents SciTraffic: a new tool aimed to develop model-based control strategies for urban traffic, which takes advantage of the topological information of the SUMO traffic network to build a simplified model for control purposes. SciTraffic has been developed in the project Modelling and Control of Urban Traffic in the City of Medellín (MOYCOT) - second stage for the evaluation of different traffic light control strategies in the city. SciTraffic is implemented in the Matlab®programming language, thanks to its ability for rapid prototyping, and utilizes tools developed previously in the project such as TraCI4Matlab and SUMOLib4Matlab. Thus, SciTraffic takes advantage of the well-structured object-oriented hierarchy of the SUMO traffic networks, and includes models for multi-modal traffic flow such as cars, Bus Rapid Transit (BRT), bicycles and pedestrians. Furthermore, SciTraffic interacts with SUMO using TraCI4Matlab for parameter estimation and visualization, and can run independently or simultaneously with the SUMO GUI for enabling the visual comparison of both models.

Keywords: Macroscopic traffic model, multimodal traffic simulation, software integration.

7.2 Introduction

Traffic congestion is nowadays one of the most common problems of modern cities. It is becoming more evident that an increased congestion brings environmental impacts like air pollution and noise. Moreover, the productivity of a city is affected since people are spending more time in traffic jams, increasing their levels of stress and worsening their quality of life. For this reason, governments, researchers and the private sector are making efforts to improve mobility, leading to strategies that can be classified into three key action areas (2; 3): substitution, modal shift and efficiency increase. The first and second areas relate to the area of transportation and urban planning, where strategies related to the land use and policies for encouraging alternative modes of transport can be found. Some of these strategies involve long-term projects and building new transportation infrastructure.

In contrast, solutions focused on increasing the efficiency of existing transportation infrastructure require a lower intervention and can be achieved in the short term. These solutions are commonly supported by technological advancements and are related to the Intelligent Transportation Systems (ITS) paradigm. ITS has received an increasing interest in the last decade, motivated by the goal of efficiently managing a traffic network and reducing accidents, supported by Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure communications (V2I). In this regard, managing a traffic network is concerned with taking the best decisions to reduce traffic congestion based on the knowledge of characteristics of the city such as the spatial and temporal distribution of activities that require trips, the available transportation infrastructure, the means for measuring the traffic state and taking decisions, and the behavior of the drivers. Some examples of technologies that benefit from ITS include adaptive traffic lights control, dynamic routing and Advanced Driving Assistance Systems (ADAS).

Traffic simulation plays a key role for the successful implementation of traffic management strategies, since it offers a controlled environment where any scenario can be tested without modifying the actual infrastructure. The more detailed the models implemented in the simulation and the more accurately their calibration, the more realistic the simulation scenario and the more reliable the traffic management strategies tested on it. Microscopic traffic models can offer such a level of detail by characterizing each driver-vehicle unit, taking into account car following models for the longitudinal behavior and lane change models for the lateral behavior. Currently, there are many software packages for microscopic traffic simulation. Among them, the Simulation of Urban Mobility (SUMO) (4) package results ideal not only for research but also for the development of commercial applications. Thanks to its open-source nature and its high integration capacity, SUMO allows to develop and validate new traffic models and to test traffic management strategies, which can be implemented in external platforms and integrated through the Traffic Control Interface (TraCI). Furthermore, SUMO includes a mesoscopic model that make simulations from 10 to 100 times faster.

On top of the microscopic traffic simulators, traffic management strategies can be developed and tested. In this work, the interest is on Model-based Predictive Control (MPC) systems for controlling actuators located in a large area of a city, which often require real-time or near real-time operation. Therefore, the prediction models used by these MPC systems have to be computationally efficient. For these reason, macroscopic models are more suitable for MPC traffic lights control (5). In this sense, it is necessary to implement macroscopic models quickly, taking advantage of some physical and operational characteristics that have already been described at the microscopic level.

In this article, a macroscopic traffic simulator intended for traffic management decision-making systems is presented, namely SciTraffic. SciTraffic is developed in Matlab, thanks to its simplicity for rapid prototyping and its wide set of toolboxes for estimation and model-based control systems. Furthermore, SciTraffic is based on the TraCI4Matlab and SUMOLib4Matlab packages, allowing to build suitable object-oriented hierarchies and simplifying visualization. SciTraffic is tested by implementing the macroscopic Berg model and visualizing the occupation of the edges in SUMO in a simplified scenario. This article is organized as follows: Section 7.3 describes SciTraffic and its architecture, section 7.4 presents simulation results in a simplified scenario, showing the interaction with SUMO, finally, section 7.5 concludes the article.

7.3 SciTraffic architecture

SciTraffic is a macroscopic traffic simulator aimed to simplify the estimation and control schemes for traffic management. SciTraffic integrates with the SUMO simulator in a closed-loop scheme, where SUMO is the plant and the main actuation systems are the traffic lights. As showed in the UML diagram of figure 7.1, the integration of SciTraffic and SUMO is made through two packages previously developed in the MOYCOT project (1):

- **TraCI4Matlab.** This package is the main interface between SUMO and SciTraffic, used for parameter estimation and visualization of the macroscopic simulation. Parameter estimation is done by running the microscopic SUMO simulation and measuring the required traffic flows and queue lengths with TraCI4Matlab and storing the results in the SciTraffic network. After that, SciTraffic perform an optimization-based parameter estimation for calibrating the macroscopic model. Additionally, TraCI4Matlab is used for visualizing the macroscopic simulation, setting the colors of the edges according to information like their occupation and the length of the queues.
- **SUMOLib4Matlab.** This package is an implementation of the sumolib library for Java/Matlab, used for building an object hierarchy from the SUMO XML network file. Thus SUMOLib4Matlab constructs the macroscopic network object, where the macroscopic simulation takes place. Note that this approach takes advantage of the powerful importing capacity of SUMO from Open Street Maps (OSM) for the rapid set up of real-world traffic networks.

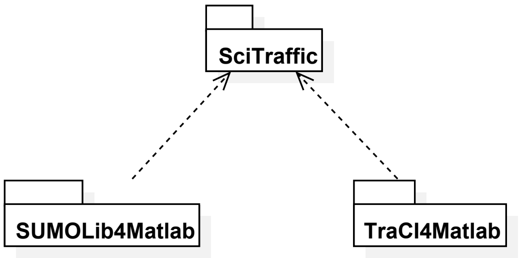


Figure 7.1: UML package diagram of SciTraffic

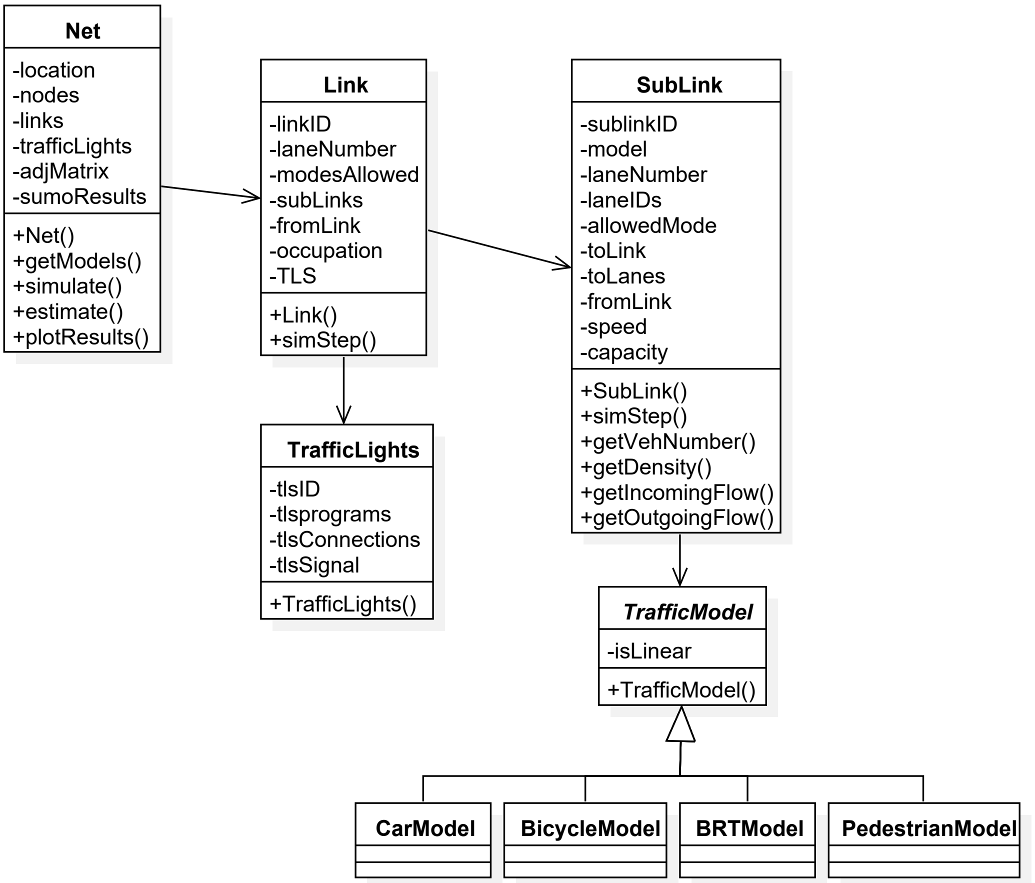


Figure 7.2: UML diagram showing the main classes implemented in SciTraffic

Figure 7.2 shows the main classes implemented in SciTraffic. The Net class wraps the corresponding class from the SUMOLib4Matlab package, with additional methods for simulating and visualizing it and estimating its model parameters. The Link and Sublink classes wrap some attributes of the Edge and Lane classes of SUMO. In this case, an object of the Link class comprises at least one object of the Sublink class, which is a group of lanes sharing the same transportation mode. Note that the Sublink class has the simStep method for performing one simulation step and has an association with a subclass of Model, since each transportation mode implements its own traffic model. These classes are explained in detail, as follows:

- **Net** The Net class is the root class representing a macroscopic traffic network. This object is constructed with the help of the network parser of SUMOLib4Matlab. In this process, the Net class organizes its own structure for supporting the macroscopic models and instantiates its child objects, being the nodes, the links and the traffic lights. Additionally, this class constructs a simplified graph of the network which can be used for linear models and visualization in Matlab; and a container for the results of the microscopic simulation run obtained through TraCI4Matlab. These results are used for calibrating the macroscopic model. The Net class includes the following methods:
  - getModels: Used for instantiating the macroscopic models implemented in SciTraffic.
  - simulate: This method performs the simulation with options for specifying the initial and end times, and for enabling visualization of the macroscopic simulation or the macroscopic and the microscopic simulations simultaneously.
  - estimate: This method performs a microscopic simulation run in SUMO and stores the results of interest in the sumoResults property. After that, estimate performs a calibration of the parameters of the macroscopic model according to the results obtained in the microscopic simulation.
  - plotResults: This method is used for visualizing several macroscopic variables after the simulation, including the flows on each link and the queue lengths.
- **Link** This class represents a road between two nodes which, unlike the edges of the SUMO network, does not contain information regarding its shape. In addition to the number of lanes, the Link class has another property for identifying the sublinks, which is an important property for macroscopic models. The Link class has some additional properties related to the macroscopic model including the occupation and the associated traffic lights system. The simStep method performs a simulation step on the link, where different macroscopic models can be implemented for each transportation mode.
- **Sublink** A sublink is a set of lanes grouped by the transportation mode they allow. This class represents the most basic unit in the macroscopic network hierarchy. This class has additional properties related to the allowed transportation mode, such as the speed and the capacity. Moreover, the Sublink class has methods for obtaining important variables that determine the traffic dynamics and the interaction with the other sublinks connected to it.
- **TrafficLigths** This class allows SciTraffic to get all information about traffic lights, which influence directly the traffic network performance. The TrafficLigths class has only the constructor method TrafficLigths() which instantiates an object with the following properties:
  - tlsID: This property contains the identifier of the traffic light signal, which is a string associated to the intersection identifier.
  - tlsPrograms: This property contains information about predefined stated on the traffic lights.

- `tlsConnections`: Because there is one traffic light signal per lane, this property contains information about connections between lanes and their turn options.
- `tlsSignal`: This property contains the traffic lights state for each lane, i.e., whether it is in green, red, etc., at a time t.

In order to illustrate the link concept, Figure 7.3 shows two nodes a and b which represent two intersections connected and isolated. Thease nodes (a and b) are linked through two edges in opposite directions, i.e., link (a, b) represent the road from a to b and vice versa.

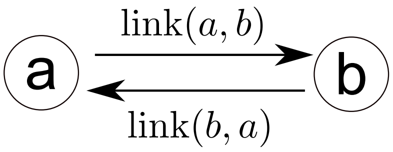


Figure 7.3: Links between node a and node b

However, each link could have exclusive lanes for only one transportation mode, thus, it is possible to find links with multi-modal flow as showed in Figure 7.4. In this particular case, the link (a, b) contains three sub-links, since it has lanes that only allow the transit of cars, BRT or bicycles. Therefore, the `Sublink` class is of great importance because it facilitates the implementation of differentiated traffic flow models for each transportation mode, regardless the fact that they belong to the same link.

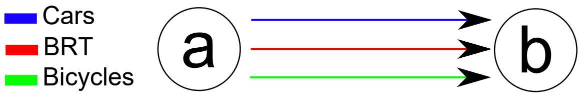


Figure 7.4: Multi-modal traffic flow to link (a, b), showing three sublinks: blue for cars, red for BRT and green for bicycles.

In the next section simulation results in a simplified scenario are presented.

7.4 Simulation results and discussion

In order to test the macroscopic modeling process in SciTraffic, a simple urban traffic network is proposed, which is showed in Figure 7.5. This urban traffic network is composed of four traffic light intersections: i_1, i_2 , i_3 and i_4 and eight input and output nodes: o_1, o_2, ... o_8. The edges between nodes and intersections are called links. These links are identified so that the edge from o_1 to i_1 is the link (o_1, i_1). Links are 100 m long, have two lanes per link and allow only cars. Traffic lights have a cycle time of 60 s where green and red time are the same.
This case of study allows to evaluate and to compare the congestion level on the network through macroscopic (using a macroscopic model) and microscopic (using SUMO) simulation. A macroscopic simulation is possible because SciTraffic is a versatile simulator where several macroscopic mathematical models can be included. In this paper, the macroscopic traffic model proposed by Berg et. al. (6) was chosen and included into SciTraffic. The Berg model has two states: the number of vehicles waiting in the queue and the storage capacity of the link (given in number of vehicles). In this particular case, the most important state is the storage because it determines the congestion level. Hence, the storage of the link (u, d) at time step $k + 1$ is given by:

$$S_{u,d}(k+1) = S_{u,d}(k) - \sum_{o_j \in O_u} n^{\text{in}}_{o_j,u,d}(k) - \sum_{o_j \in O_d} n^{\text{out}}_{u,d,o_j}(k) \qquad (7.1)$$

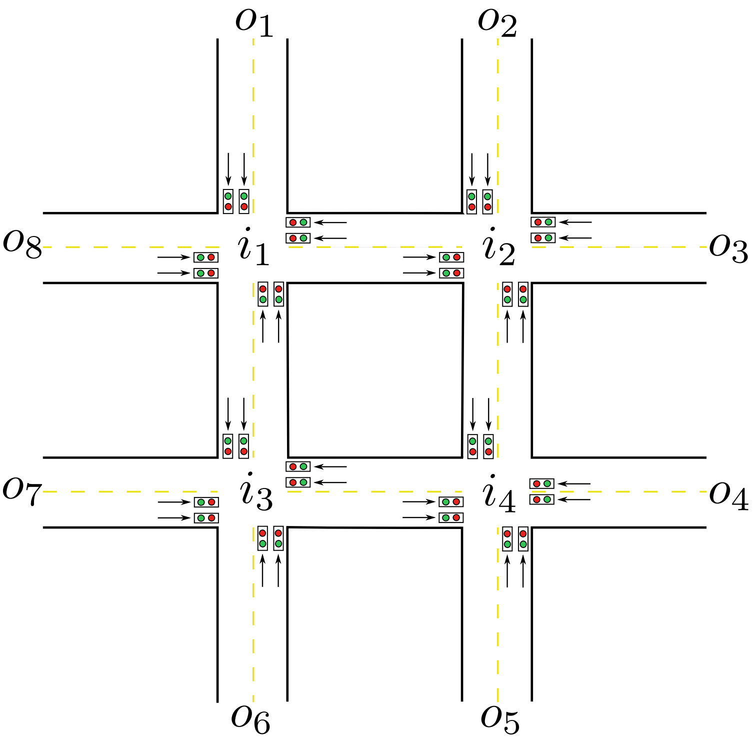


Figure 7.5: Urban traffic network scheme

where $S_{u,d}(k)$ is the storage at time step k (previous step), $\sum_{o_j \in O_u} n^{\text{in}}_{o_j,u,d}(k)$ is the sum of all vehicles entering to link u, d during $[Tk, T(k+1)]$, and $\sum_{o_j \in O_d} n^{\text{out}}_{u,d,o_j}(k)$ is the sum of all vehicles leaving from link u, d during $[Tk, T(k+1)]$, where T represents the sample time $T = 5$ s. Furthermore, O_u is the set of incoming links to link u, d and O_d is the set of outgoing links from link u, d. In the borders o, the vehicles entering to the links are obtained from SUMO trough TraCI4Matlab until time step k i.e., every 5 s. For internal nodes, the vehicles entering to the links are the same vehicles leaving from upstream links. The number of vehicles leaving a link are given by the traffic light signal and the parameters: free-flow speed, vehicle average length, lanes number, saturation flow and turning rates. These parameters are unknown and depend on the specific setting of the network, therefore, they must be measured or estimated. In order to simplify, the parameters of the Berg model are the same as those typically reported in the literature (6), including the free flow velocity, the average length of a car and the maximum outgoing flow from a link (saturation flow). Note that there are no turns at intersections, then turning flow parameters are not considered.

The urban traffic network presented in Figure 7.5 was built in SUMO using SUMOLib4Matlab (see Figure 7.6).

The network showed in Figure 7.6 is available for microscopic simulation. However, it is possible to modify some graphical characteristics of this network in real time, including macroscopic information about congestion level. The congestion level can be described taking into account the color convention of Figure 7.7. If one link is empty (0% occupation) then this link will look green. The color of this link can gradually change to red, representing a full (100%) occupation.

From (7.2) it is possible obtain the occupation at each time step k, as follows:

$$0\text{ccupation}(k) = \frac{100 - S(k)}{S(1)} \tag{7.2}$$

where $S(1)$ is the initial condition of storage. Generally, initial conditions are set so that the traffic network is empty and a maximum storage, i.e. equivalent to the link capacity (maximum number of vehicles that fit in one link).

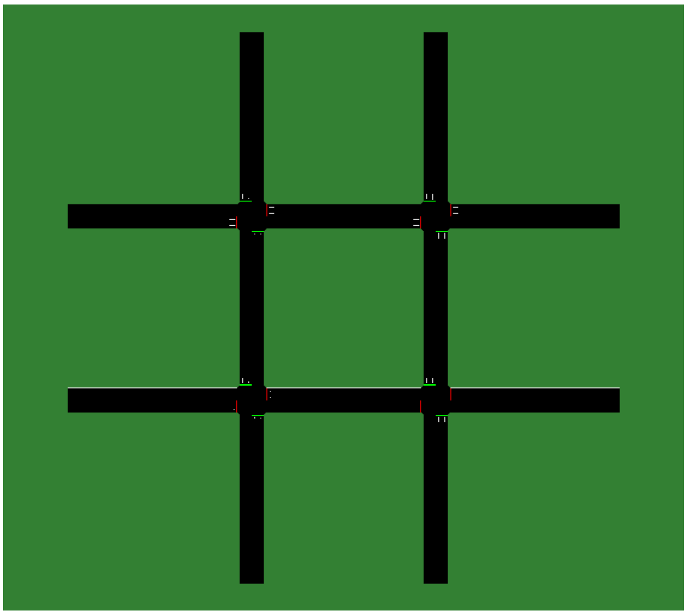


Figure 7.6: Urban traffic network in SUMO

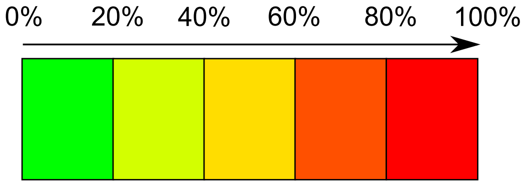


Figure 7.7: Color convention for denoting congestion levels (occupation)

In order to obtain a dynamic visualization of the network in SUMO according to the congestion level, it is mandatory to set up adaptive travel time in both SUMO and SciTraffic. This option (adaptive travel time) it is an easy way of coloring a street. The color convention showed in Figure 7.7 has to be included in the SUMO GUI like a gui-settings-file with extension *.xml*. On the other hand, SciTraffic uses a TraCI command namely *traci.edge.adaptTraveltime* for each link for every time step. Figure 7.8 shows the network in SUMO and SciTraffic for different times and congestion levels. Note that the network is simulated microscopically and macroscopically simultaneously.

Figure 7.8 shows the levels of congestion. Thus, when a large-scale traffic network is simulated, it is possible to evaluate and to monitor the congestion zones regardless of the size of the network. The proposed traffic network has 24 links, meaning that at each sample time (5 s), a set of equations for each link is solved to find a theoretical occupation. Furthermore, the simulation time is 3.600 s but the computational burden is different if the same network is simulated using SUMO, SciTraffic (without graphical interface) or both (As shown in Figure 7.8). The computation time was 54.4 s, 32.2 s, and 99.84 s for SUMO, SciTraffic (without graphical interface) and both at the same time with graphical interface). It is clear that the macroscopic simulation is the most efficient with respect to computational burden, next the microscopic simulation and finally the simultaneous stimulation (microscopic and macroscopic). Note that simultaneous simulation takes more time than the sum of macroscopic and microscopic times, it is due to communication resources required by TraCI4Matlab to establish the congestion levels using colors. However, it is important to clarify that SUMO and SciTraffic are not required to run simultaneously, i.e. SciTraffic only uses the SUMO GUI for visualization and can run independently, since it builds the traffic network topology using SUMOLib4Matlab.

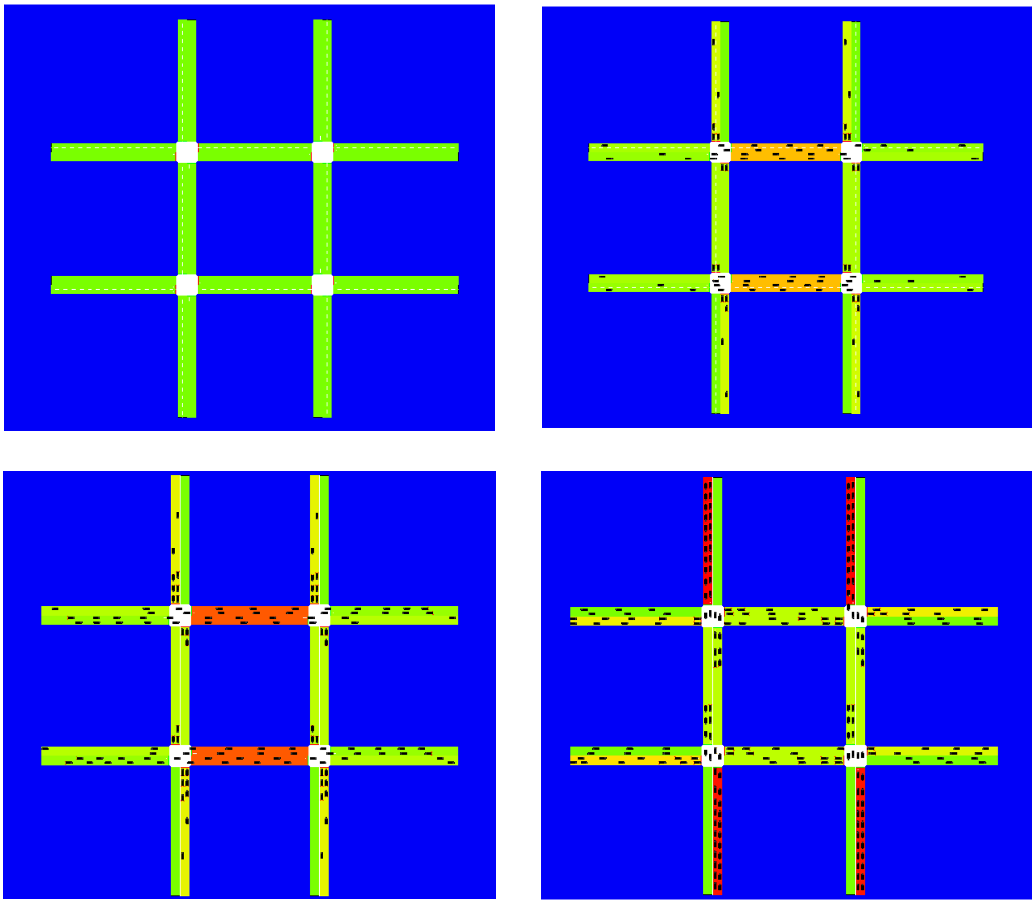


Figure 7.8: SciTraffic screenshots - occupation of the traffic network with different congestion levels

7.5 Conclusions

This article presented a new macroscopic urban simulator, namely SciTraffic, which allows to include any macroscopic model and can run simultaneously with the SUMO microscopic simulation. SciTraffic takes advantage of network characteristics developed in SUMO and the advantages offered by Matlab to increase versatility in the development of traffic control systems.

SciTraffic was tested through a simple case study, where satisfactory results were obtained. However, the results presented in this article are only a test among the many possible applications.

In order to continue this work it is necessary to include a parameter estimation of the Berg model to improve the results, including the estimation of turning flows at intersections. Furthermore, the Berg model can be used for control purposes, allowing to approach real time control strategies. Finally, the performance of SciTraffic should be evaluated for increasing demand and scenarios and comparing the results with the mesoscopic model of SUMO in order to evaluate how this mesoscopic model could be eventually included as a prediction model in the control architecture, since it could offer an improved accuracy.

7.6 Acknowledgments

This work was supported by Proyecto Colciencias 111856934640 contrato - FP44842-202-2015: Modelamiento y Control de tráfico urbano en la ciudad de Medellín Fase 2. Convocatoria 669.

7.7 Bibliography

[1] Acosta, Andrés, Jorge E. Espinosa and Jairo Espinosa: *Developing Tools for Building Simulation Scenarios for SUMO Based on the SCRUM Methodology*. In *Proceedings of the 3rd SUMO User Conference SUMO2015*, pages 23–35, Deutsches Zentrum für Luft - und Raumfahrt e.V., May 2015. Deutsches Zentrum für Luft - und Raumfahrt e.V.

[2] Banister, David: *The sustainable mobility paradigm*. Transport policy, 15(2):73–80, 2008.

[3] Hidalgo, Darío: *transporte sostenible para américa latina: situación actual y perspectivas*. Documento de respaldo foro de transportes sostenible FTS de América Latina 2011, pages 22–24, 2011.

[4] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3&4):128–138, December 2012.

[5] Lin, S., B. De Schutter, Y. Xi and H. Hellendoorn: *Integrated urban traffic control for the reduction of travel delays and emissions*. IEEE Transactions on Intelligent Transportation Systems, 14(4):1609–1619, December 2013.

[6] van den Berg, M., A. Hegyi, B. De Schutter and J. Hellendoorn: *A macroscopic traffic flow model for integrated control of freeway and urban traffic networks*. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, pages 2774–2779, Maui, Hawaii, December 2003.

8 Impact of Autonomous Vehicles in an Urban Environment Controlled by Static Traffic Lights System

François Vaudrin, Jakob Erdmann and Laurence Capus

François Vaudrin, Civil Engineer, Laval University, Québec, Canada
Faculty of Sciences and Engineering
Université Laval, Québec, Canada

Jakob Erdmann, Ph. D.
German Aerospace Center (DLR)
Institute of Transportation Systems, Berlin-Adlershof, Germany

Laurence Capus, Ph. D.
Associate Professor, Faculty of Sciences and Engineering
Université Laval, Québec, Canada

francois.vaudrin.1@ulaval.ca *Jakob.Erdmann@dlr.de* *Laurence.Capus@ift.ulaval.ca*

8.1 Abstract

Autonomous vehicles are the subject of intensive research and are now targeted by many car manufacturers and others. One may think that these cars will be part of the landscape in a few years as electric vehicles. If we assume that autonomous cars will comply with safety rules and speed limits, this may have an impact on traffic fluidity.

Moreover, the establishment of autonomous cars will be gradual and will take several years, if not decades. Infrastructure and existing traffic light systems will also continue to be used by governments even with the addition of autonomous vehicles. These new vehicles will therefore have to coexist with the current systems for several years.

As part of this research, we propose to measure the impact of autonomous cars in an urban environment controlled by static traffic lights system like those used in cities. To carry out this research, we used the SUMO simulator and modeled a network of few intersections in the city of Québec in Canada. The purpose of this paper is to present the work in progress in order to better understand the effects of autonomous cars on urban travel.

Keywords: SUMO, Autonomous cars, TLS

8.2 Introduction

Autonomous Vehicles (AVs) are the subject of intensive research and are now targeted by many car manufacturers and others. Also, several mainstream companies, such as Google, Toyota, Nissan, and Audi, are developing and testing their own prototypes (10). Those who have confidence in the benefits of AVs believe that this technology will considerably reduce traffic problems, accidents and polluting emissions (6). Moreover, the implementation of the AVs will take place over several years, if not several decades. The arrival of AVs in cities is expected to begin in the 2020s or 2030s, but the real effects to improve transport and better traffic flow will occur later, possibly in years 2040s to 2060s. Beneficial impacts on safety and congestion are likely to appear between 2040 and 2060 (7).

There are several factors that need to be considered when introducing new technology, including existing systems and infrastructure. For example, there were approximately 265 000 traffic signals in the United States in 2004 (12). These infrastructure have required significant investments. It can therefore be assumed that the current Traffic Lights System (TLS) will continue to be used for several years even with the arrival of AVs.

Given the importance of TLS, we question whether the addition of AVs will have a tangible effect on the fluidity of traffic in urban areas. The purpose of this paper is to measure the impact of AVs in an urban environment controlled by static TLS. To carry out this research, we used the SUMO simulator and modeled a network of few intersections in the city of Québec in Canada. The purpose of this paper is to present the work in progress in order to better understand the effects of AVs on urban travel. The next section refers to the related works, the third focuses on the methodology, the fourth on the results, the fifth is the discussion, the sixth is relate to future works and the last concludes the paper.

8.3 Related Work

There is a lot of speculation about the impact of AVs on transportation (9). Some believe they will reduce road congestion, greenhouse gas emissions and revolutionize transportation. Others are of the opinion that this technology will have negative impacts because it will add vehicles to the network (8).

One of the factors that makes AVs perform better than normal cars with drivers is because the distance between vehicles can be reduced (headway). This allows a better distribution of available space. Although the AVs technology allows the reduction of the distance between vehicles (Figure 8.1), the rules in force in several countries required to respect a minimum distance between each car (14). It is also believable that the authorities will ask AVs to respect these distances for safety reasons. In addition, it is possible that the minimum distance between vehicles should be increased to reduce the risk of collisions of autonomous cars.

The other factor that advocates AVs is the ability to move in groups (platooning). However, this is only possible when all vehicles are autonomous and the addition of a single human conductor can significantly reduce the performance of all vehicles (4). Some studies are being carried out with a view to establishing a coordination between autonomous cars and traffic lights (13). This implies, however, that there are changes to the existing TLS and that there is a sufficient number of AVs.

One must also consider the behaviour of human drivers in the model. For example, there is an unwritten rule at least in Québec (Canada) that police generally tolerate exceeding the speed limit of 20 km/h. This practice appears to be corroborated by the results of a survey carried out in 2010

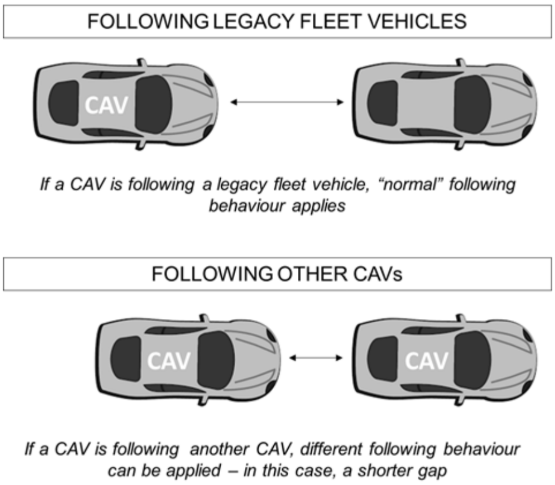

FOLLOWING LEGACY FLEET VEHICLES
CAV
If a CAV is following a legacy fleet vehicle, "normal" following behaviour applies
FOLLOWING OTHER CAVs
CAV
CAV
If a CAV is following another CAV, different following behaviour can be applied – in this case, a shorter gap

Figure 8.1: Headway from (1)

in Québec. On a sample of urban streets with a speed limit of 50 km/h, the survey shown that 85% of vehicles exceed the speed limit (2). Moreover, between 2010 and 2012, approximately 400 000 speeding offences were issued in areas where the speed limit was 60 km/h or less. The majority of offence reports (63%) concern excesses between 21 and 30 km/h above the speed limit. Finally, the result of a survey indicates that most drivers report exceeding the speed limit (11). If we take for granted that the AVs will move with careful attention to speed limits, this could affect traffic, vehicle speeds and perhaps create congestion.

Will these impacts be positive, negative or negligible? It is not easy to answer this questions because one must reproduce human behaviour with a computer and there are other factors that may also influence the results as the effectiveness of TLS.

8.4 Methodology

8.4.1 Mixing of AVs and Normal Vehicles

We did a simulation in a busy area of Quebec City (Canada). The metropolitan area has a population estimate of 800 000 peoples. The network consists of 13 intersections including 7 TLS static and covers 1.5 km. The traffic flows and the programming of the TLS comes from the data provided by the city of Québec. Data were measured at peak hours in the afternoon, the number of lanes varied, and parking is permitted at certain roadside locations. There are no detection loops and pedestrians are not considered in the model. The duration of each simulation is 3 600 seconds. Performance measurements are based on the waiting time of vehicles during the simulation. The Krauß-model with some modifications is the default model used in SUMO. This model let vehicles drive as fast as possibly while maintaining safety gap (always being able to avoid a collision if the leader starts braking within leader and follower maximum acceleration bounds) (3).

The simulations consist of separating the vehicles into two categories (AVs and normal). The only hypothesis that we have posed for now is that the AVs will more respect the speed limits and that the speed deviations will be lower compared to normal vehicles. We have arbitrarily defined a standard deviation from the speed limit of 0.05 for AVs and a standard deviation of 0.10 for normal vehicles. It is understood that other parameters could also be considered (headway, acceleration, deceleration, car-following models, lane-changing, etc.).

There are 4 043 vehicles circulating on the network during each simulation and departures are randomly allocated for each category of vehicle. We performed 51 simulations with the SUMO-GUI. This allows to visualize if there are particular problems during each simulation and if the results are consistent with what was observed during the simulations. For example, the SUMO GUI allowed us to validate that faster vehicles exceed the slowest in some simulations.

The first series of simulations relates only to normal vehicles. The speeds are increased by 5 km/h at each simulation (25 km/h, 30 km/h, ..., 65 km/h). The second simulation series consists of mixing AVs and normal vehicles. The proportion of AVs is 64% (2 580) and normal vehicle 36% (1 463). We repeat the same tests by increasing only the speed of the AVs with normal vehicles moving at the speed limit (speedFactor = 1). For the third series of simulations, we reversed the proportion of AVs (36%) and normal vehicles (64%).

8.4.2 TLS Impact on the Average Speed of Trips

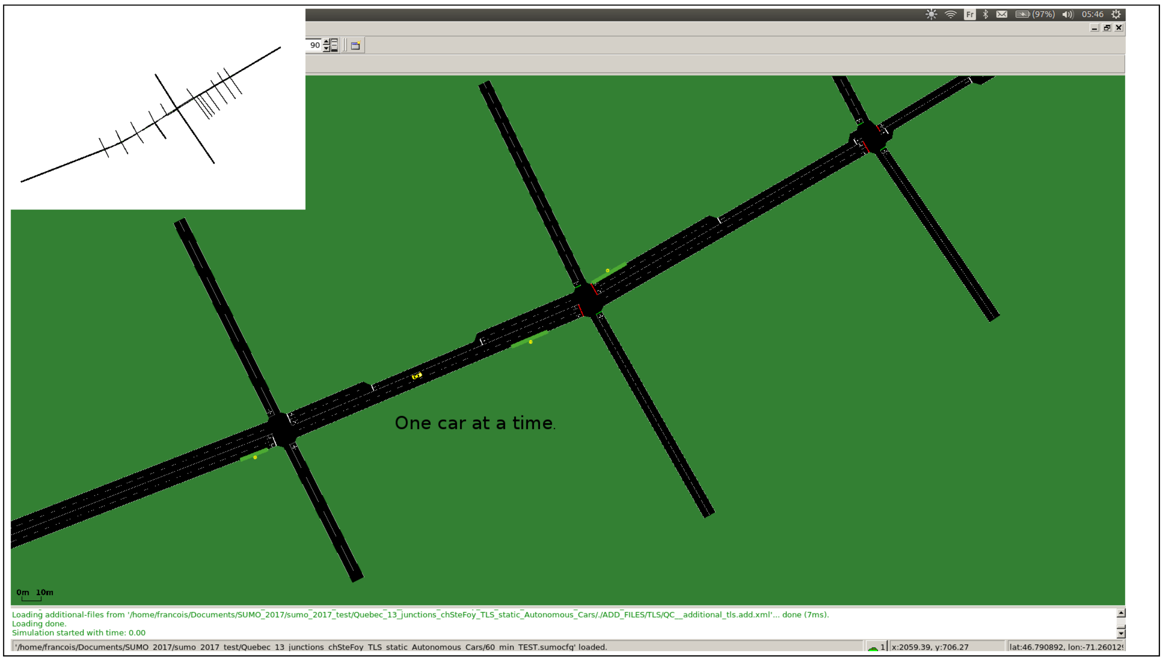
The results of previous simulations did not give significant differences as we will see in section 4. We then decided to continue the investigation by checking the impact of the TLS on the average vehicle speed during a trip. For example, if a vehicle travels in an unobstructed straight line, the difference between the expected speed and the measured speed should be minimal. Consequently the standard deviation should be close to zero. If the standard deviation is large, then the impact of TLS is significant.

To measure this, we generate random paths of a single vehicle (without external interactions) over the same road network and TLSs (Figure 8.2). We then generate trips in SUMO. Each time a vehicle finishes its route, SUMO generates another route randomly and so on until the end of the simulation. We also varied the speedFactor during simulations. Then, we calculate the average speed travelled for each trip. This speed is calculated by dividing the length of each path by its duration. By randomly repeating the paths during the simulations, we obtain an average speed for each speedFactor. We then examine whether there is a correlation between the speedFactor and the average speed. There were more than 10 000 simulations (trips) during the experiment. These simulations were performed using a Python script in a terminal.

8.5 Results

8.5.1 Mixing AVs and normal vehicles

The Figure 8.3 shows the results of mixing AVs and normal cars. For the first test (blue line), the difference between minimum and maximum waiting time is about 15% (400 minutes). Minimum waiting time is obtained when vehicles are moving at a speed comparable to the speed limit (speedFactor = 1.0). One explanation is that these TLSs have been programmed according to the speed limit and it is logical that the minimum waiting time is obtained when the average speed approaches this speed. We also see a peak between the speedFactor 0.85 and 0.95. We have no explanation for this peak. The second test (orange line) showed results comparable to the first test. The results of the third test are similar to those of the previous tests. It does not matter if AVs are



```
Loading additional-files from '/home/francois/Documents/SUMO_2017/sumo_2017_test/Quebec_13_junctions_chSteFoy_TLS_static_Autonomous_Cars/./ADD_FILES/TLS/QC__additional_tls.add.xml'... done (7ms).
Loading done.
Simulation started with time: 0.00
'/home/francois/Documents/SUMO 2017/sumo 2017 test/Quebec 13 junctions chSteFoy TLS static Autonomous Cars/60 min TEST.sumocfg' loaded.
```

Figure 8.2: SUMO network

added to normal cars, the results are comparable in all cases in this experiment. It should not be forgotten, however, that this is a particular environment with TLSs programmed at fixed times with no interaction between vehicles and TLSs.

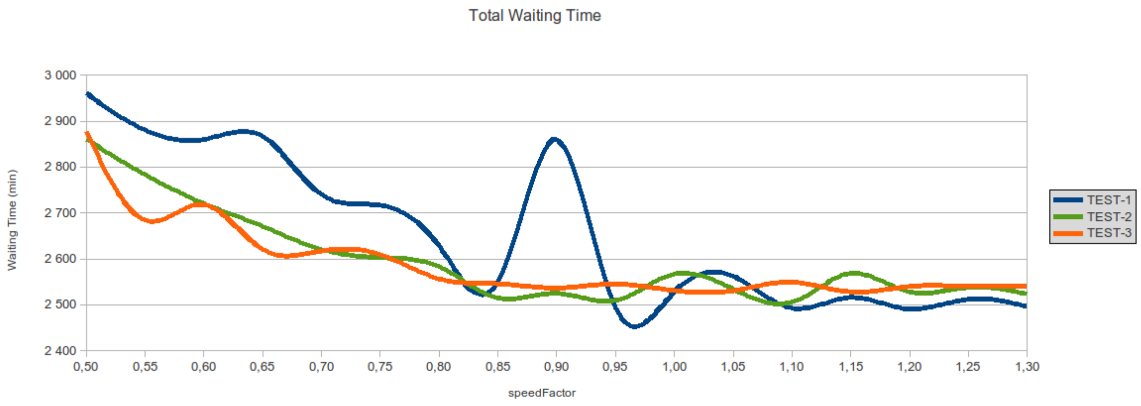


Figure 8.3: Total Waiting Time AVs vs Normal Vehicles

8.5.2 Simulation of Single Random Paths

This section aims to validate the influence of TLS on average vehicle speed and indirectly on waiting times. For each series of simulations, the speed factor remains constant. During the first simulation (Table 8.1), there were 627 random trajectories and SUMO calculated the speed of each of these trips (length / duration). For example, for the initial speedFactor 0,5 (25 km/h), the speed average is 5,5 m/s (19,8 km/h), for an initial speedFactor 1,2 (60 km/h), the speed average is 11,1 m/s (40 km/h),and so on for others speed factor. We also increased the speed factor up to four times in order to check if the speed tends towards a limit, which is indeed the case. When the speed factor is greater than 2.0 (100 km/h), the average speed tends towards a maximum value (55 km/h or 15,3 m/s) as shown in Figure 8.4.

To evaluate the correlation between speed limit and average speed, we use the Pearson correlation r factor (equation 1). When r is between 0.5 and 1.0, this indicates that the correlation is very high between the two variables. The r correlation factor obtain is 0.99 for data between 0,5 and 1,5 (25 km/h to 75 km/h). This is an almost perfect correlation. We have also done simulations by increasing the number of vehicles (10, 100, 500) and the correlation was similar.

$$r = \frac{\sum_{i=1}^{k}((F_i - \bar{F})(Savg_i - S\bar{a}vg))}{\sqrt{\sum_{i=1}^{k}(F_i - \bar{F})^2}\sqrt{\sum_{i=1}^{k}(Savg_i - S\bar{a}vg)^2}} \tag{8.1}$$

$r = Pearson's\ correlation\ factor$
$k = the\ first\ 11\ rows\ in\ table\ 1\ (between\ SpeedFactor\ 0,5\ and\ 1,5)$
$F_i = speed\ Factor$
$\bar{F} = speed\ Factor\ mean$
$Savg_i = speed\ average$
$S\bar{a}vg = Speed\ average\ mean$

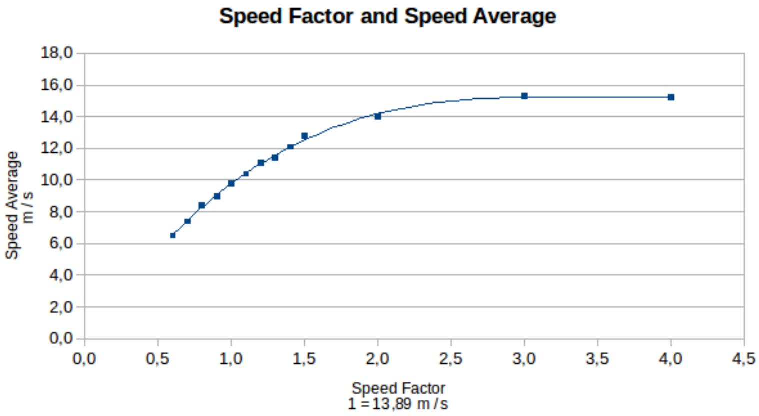
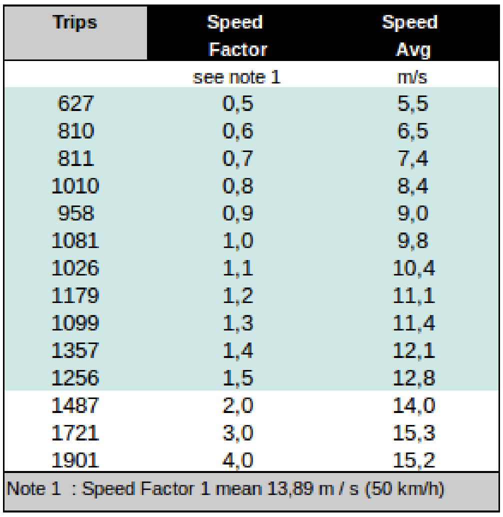


Figure 8.4: Speed Average

Table 8.2 show the relative average speed compare to the speed factor. Figure 8.5 shows that if the speed increases, the relative average speed decreases. This means that the effect of TLSs is accentuated when the speed increases. If there was no TLSs, the difference would be less between these speeds.

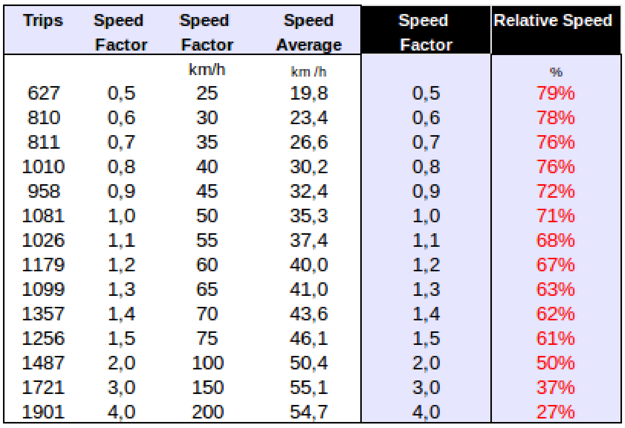
We also calculated the standard deviation of the speed (equation 2). The purpose of this test was to check whether the standard deviation varies with speed. The results are shown in Table 8.3 and Figure 8.6.

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(s_i - \bar{s})^2}. \tag{8.2}$$



Trips	Speed Factor	Speed Avg
	see note 1	m/s
627	0,5	5,5
810	0,6	6,5
811	0,7	7,4
1010	0,8	8,4
958	0,9	9,0
1081	1,0	9,8
1026	1,1	10,4
1179	1,2	11,1
1099	1,3	11,4
1357	1,4	12,1
1256	1,5	12,8
1487	2,0	14,0
1721	3,0	15,3
1901	4,0	15,2
Note 1 : Speed Factor 1 mean 13,89 m / s (50 km/h)		

Table 8.1: Speed Average



Trips	Speed Factor	Speed Factor	Speed Average	Speed Factor	Relative Speed
		km/h	km /h		%
627	0,5	25	19,8	0,5	79%
810	0,6	30	23,4	0,6	78%
811	0,7	35	26,6	0,7	76%
1010	0,8	40	30,2	0,8	76%
958	0,9	45	32,4	0,9	72%
1081	1,0	50	35,3	1,0	71%
1026	1,1	55	37,4	1,1	68%
1179	1,2	60	40,0	1,2	67%
1099	1,3	65	41,0	1,3	63%
1357	1,4	70	43,6	1,4	62%
1256	1,5	75	46,1	1,5	61%
1487	2,0	100	50,4	2,0	50%
1721	3,0	150	55,1	3,0	37%
1901	4,0	200	54,7	4,0	27%

Table 8.2: Relative Speed

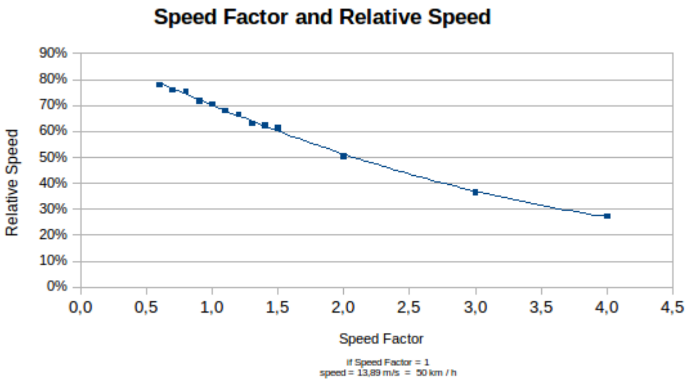


Figure 8.5: Relative Speed

$\sigma = standard\ deviation$
$s_i = measuring\ speed\ (trip\ length\ divided\ by\ duration)$
$n = number\ of\ trips$
$\bar{s} = speed\ average$

The standard deviation is mainly influenced by the length of each route and the number of TLS met during the trip. If a path is long and there are few TLSs, then the speed should therefore be more constant and the standard deviation lower. Furthermore, if a path is short and it crosses several TLSs, its standard deviation should be larger. We note that the standard deviation increases when the speed increases. This suggests that the higher is the speed, the higher is the impact of TLS on the average speed.

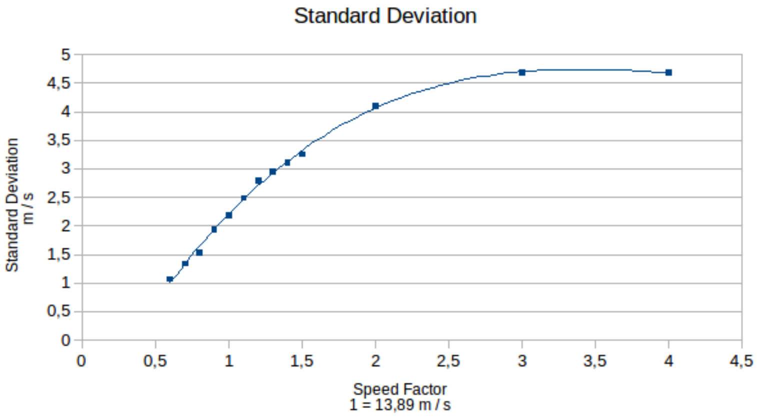
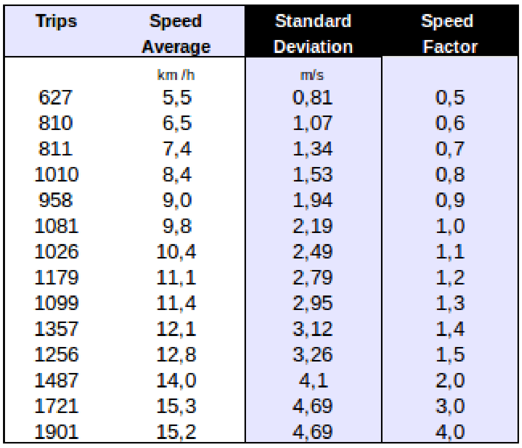


Figure 8.6: Standard Deviation



Trips	Speed Average	Standard Deviation	Speed Factor
	km /h	m/s	
627	5,5	0,81	0,5
810	6,5	1,07	0,6
811	7,4	1,34	0,7
1010	8,4	1,53	0,8
958	9,0	1,94	0,9
1081	9,8	2,19	1,0
1026	10,4	2,49	1,1
1179	11,1	2,79	1,2
1099	11,4	2,95	1,3
1357	12,1	3,12	1,4
1256	12,8	3,26	1,5
1487	14,0	4,1	2,0
1721	15,3	4,69	3,0
1901	15,2	4,69	4,0

Table 8.3: Standard Deviation

8.6 Discussion

Our model has been simplified and it is possible that the addition of other parameters give different results. We believe, however, that the conclusions would be the same because it does not appear that the addition of AVs decreases the total waiting times. We recall that we used a model based on real data and that the programming of TLS in the model is similar to that existing in the field. The results of the Figure 3 are similar both for normal vehicles and mixing AVs and normal vehicles. Gaps in general are less than 5% for total waiting times. We consider that these gaps are not significant due to the uncertainty associated with the modelling.

Our model also seems to show that TLS statics have more influence on waiting times than the addition of AVs. Our explanation is that even if one walks, takes a bicycle or drives a fast car, one must stop and wait at the red lights. In other words, static TLS are independent of vehicles travelling on roads. One might think that the AVs will favour the movements of vehicles platooning but this will be useless if the traffic lights are not synchronized. And this is virtually impossible to predict in a chaotic situation or congestion.

Maybe AVs could slow down other vehicles and improve relative speed, but we think that this would have little impact on total waiting time. Furthermore, if the TLS are incorrectly programmed (12), the addition of AVs will not make them more efficient. Naturally, the situation could be different with efficient adaptive systems in real time.

Finally, although that most drivers exceed the speed limit, this seems to have little impact on total waiting times. The total waiting time is similar even when vehicles travel faster than the speed limit.

8.7 Future Works

SUMO offers the possibility to vary the speed and its standard deviation (speedDev). This allowed us to partially model the behavior of AVs. It might be useful to get closer to the behaviour of AVs using other parameters. It would also be interesting to do a similar experiment in other contexts, for example on a highway.

Otherwise, the Institute of Transportation Engineers (5) recommend collecting traffic data to program static TLSs. The presence of AVs should be considered as trucks, buses, public transport, pedestrians or cyclists. It may also be necessary to modify the car-following-model for AVs.

8.8 Conclusion

We wanted to study the impact of AVs on the fluidity of traffic in an urban environment controlled by static TLS systems. We mixed AVs and normal vehicles to see if this was affecting the total waiting times. Subject to the limitations of our AV model, it does not appear that the presence of AVs has a significant influence on the improvement of waiting times. We found that TLS seems to have a much greater impact than AVs on waiting time. It seems that it is not the AVs that influences the effectiveness of TLS but rather the TLS that influence the average speed of all vehicles, including AVs. We emphasize that this is a TLS "static" model with fixed sequences such as those used in most cities. Maybe the effect could be felt more on highways because AVs could possibly induce vehicles to move in groups (platoon) and help to regulate traffic flow.

Acknowledgements

We thank the German Aerospace Centre (DLR) for their continual support and availability. We would also like to thank the City of Québec for its collaboration in this research.

8.9 Bibliography

[1] Atkins Ltd: *Research on the Impacts of Connected and Autonomous Vehicles (CAVs) on Traffic Flow*. Summary Report on the behalf of the Department for Transport, United Kingdom, 2016.

[2] Bellalite, L.: *Étude des conditions optimales correspondant aux différentes limites de vitesse en milieu urbain*. Final Report, Laboratoire application et recherche en aménagement, Sherbrooke University, Canada, 2011.

[3] DLR: *Definition of Vehicles, Vehicle Types, and Routes, Default Krauss Model Description*. http://sumo.dlr.de/wiki/, last consulted january 2017.

[4] Friedrich, Bernhard: *The Effect of Autonomous Vehicles on Traffic,Autonomous Driving, Technical, Legal and Social Aspects*. Springer Open, Germany, 2016.

[5] Institute of Transportation Engineers: *Traffic Signal Timing Manual*. Washington, 2009.

[6] Keen, Andrew: *The future of travel: How driverless cars could change everything*. http://www.cnn.com/2013/05/14/business/bussiness-traveller-futurecast-driverless-car, 2013, last consulted january 2017.

[7] Litman, Todd: *Autonomous Vehicle Implementation Predictions: Implications for Transport Planning*. Victoria Transport Policy Institute, TRB, Washington, 2015.

[8] NHTSA: *Accelerating the Next Revolution In Roadway Safety, Federal Automated Vehicles Policy*. National Highway Traffic Safety Administration, US Department of Transportation, 2016.

[9] Polzin, Steve: *Implications to Public Transportation of Automated or Connected Vehicles*. Center for Urban Transportation Research, University of South Florida, 2013.

[10] Prateek Bansal, Kara M. Kockelman and Amit Singh: *ASSESSING PUBLIC OPINIONS OF AND INTEREST IN NEW VEHICLE TECHNOLOGIES: an Austin perspective*. TRB, 2016.

[11] SAAQ: *Évaluation de la campagne vitesse 2013, « Ralentir, faut le dire »*. Société de l'assurance automobile du Québec (SAAQ), Évaluation de la campagne vitesse 2013, « Ralentir, faut le dire », Final Report, SOM Research & Surveys, Canada, 2013.

[12] Tarnoff, Philip J. and Javier Ordonez: *Signal Timing Practices and Procedures, State of the Practice*. University of Maryland, Center for Advanced Transportation Technology, Maryland, 2004.

[13] Tettamanti, Tamás, István Varga and Zsolt Szalay: *Impacts of Autonomous Cars from a Traffic Engineering Perspective*. Periodica Polytechnica Transportation Engineering, 44(4):244–250, 2016.

[14] Wagner, Peter: *Traffic Control and Traffic Management in a Transportation System with Autonomous Vehicles*. Institute of Transportation Systems, German Aerospace Centre (DLR), Autonomous Driving Technical, Legal and Social Aspects, Springer Open, Berlin, Germany, 2016.

9 Personal Rapid Transit simulations with SUMO

Joerg Schweizer and Federico Rupi; Dept. of Civil, Environmental, Chemical and Materials Engineering (DICAM) - University of Bologna, Italy
{Joerg.Schweizer, Federico.Rupi}@unibo.it

9.1 Abstract

Personal Rapid Transit (PRT), also known as podcars, is an emerging public transport technology which consists of fully automated, electric vehicles, running on exclusive, light-weight guideways (rails or other passive guiding structures), which can be elevated, under-ground or at ground, if protected. Typical PRT vehicles have room for 4-6 passengers. PRT vehicles are available on demand and drive the passengers directly and non-stop to a pre-selected destination. There are only a few low-capacity PRT systems in public operation today. However with the advancement of technologies for driverless cars, also the capacities of PRT may increase and PRT could find wider application fields and markets in urban transport.

This work has two aims: (i) to explore the SUMO micro-simulation framework for analyzing and evaluating the performance of PRT networks in terms of transport capacity; (ii) to demonstrate a novel PRT vehicle redistribution system based on flow predictions. In the present framework, SUMO is used to perform the low-level vehicle tasks, such as acceleration, deceleration, diverge and merge operations. Also platooning is achieved by configuring specific vehicle attributes. A PRT vehicle is emulated by a SUMO electric vehicle type. The edges are made accessible for PRT vehicles only and zipper-type junctions are used to join the edges. The vehicle type parameters are adjusted to mimic the dynamic characteristics of real PRT vehicles. A change in the control parameters allows also to let vehicles closely follow each other, hence to create platoons. High-level functions, such as stations and vehicle management, are implemented in python and control the vehicles through SUMO-Traci. The vehicle management optimizes flows, such that the waiting time of passengers at stations is minimized. The novel vehicle management algorithm is based on vehicle and passenger flow prediction, and aims at avoiding flow oscillations in the network.

The performance of the PRT simulation are demonstrated by means of a test track, where the capacity limits are identified. The influence of some control parameters of the vehicle management are also shown. Finally some critical implementation issues are highlighted and suggestions for further improvements are made.

Keywords: PRT, Personal Rapid Transit, PRT, automated transit networks, ATN, microsimulation, vehicle management, multi-modality, platooning.

9.2 Introduction

As the nature of Personal Rapid Transit PRT and PRT developments are little know, this introduction gives first a brief overview on PRT characteristics as well as on past and present PRT systems.

Thereafter the present work is motivated by explaining why SUMO can help to quantify the benefits and costs of PRT applied on an urban area.

9.2.1 The basic concept of PRT

PRT is a public transport network based on small-size, fully automated vehicles running on exclusive guideways. Clearly, all vehicles run all the time fully automated, neither other vehicle types nor manually driven vehicles are allowed during regular operation. PRT offers passengers the opportunity to travel individually or in small groups to a pre-defined station, without the need to transfer between vehicles. The electric, 4-6 passenger vehicles arrive on-demand or they are already available at dedicated stations, similar to a taxi stand. Each trip is direct and ideally non-stop. The latter is possible because PRT stations are typically off-line, such that vehicles only need to stop when they reached their final destination station.

PRT vehicles run on a network of narrow guideways. The small minimum guideway curvature (typically less than $10m$) and the small cross section (less than $1m^2$) allow to route the guideways through a city, either underground in narrow shafts, elevated, or through buildings.

Those characteristics have been shown to optimize an urban transit system in terms of travel time, and trip costs (1). In brief, it is argued that large vehicles may no longer make economical sense if the driver can be replaced by computers and sensors: a large number of small, low-cost, mass produced vehicles running on a seamless network do offer a faster and more convenient service than a few large-size, expensive because manually assembled vehicles, running on line-oriented services. Consequently, the much lighter infrastructure of PRT guideways reduce construction costs with respect to tram or light rail systems.

9.2.2 A brief history and state of PRT development

The idea of PRT is not new. Already during the 60th, several independent PRT studies have been carried out in different parts of the world, most notably the work from Donn Fichter in 1964 (7), and the studies ordered by the Urban Mass Transportation Administration (UMTA) in 1966, known as the HUD studies. During the 70th and 80th, a dozen of PRT projects aimed at building PRT models and prototypes.

The following PRT projects demonstrated at least a full scale, operating model with multiple vehicles: (i) The Computer-Controlled Vehicle System (CVS) has been a one-second-headway, 1000kg, four passenger-vehicle PRT system developed in Japan beginning in 1968. A 1000-vehicle network was simulated, and a full-scale test facility began operating in 1972 in a suburb of Tokyo with 4.8 km of guideway and 60 vehicles. (ii) The Kabinentaxi PRT has been developed by the two firms, Messerschmitt-Bölkow-Blohm (MBB) and DEMAG. The testing on a full-scale facility near Hagen, Germany, began in May 1973. The test facility has been expanded several times. In 1976 the test track had a length of 1.9 km with six stations. The twenty-four vehicles were running with headways of $3s$ at speeds up to 36km/h and logged over 640,000 vehicle-km between 1975 and 1978. In 1978 the vehicles completed an endurance test of 10,000 continuous vehicle hours. A large variety of tests on maintenance and human factors were also performed in preparation for offering the system for deployment in cities. (iii) In late 1970, Matra received its first contract to develop a PRT system called Aramis from the French agency DATAR. Fullscale testing with four-seat vehicles began in April 1973 at Orly International Airport and by summer 1974 the first phase of proof testing of the basic concept was finished. Aramis featured platoon formation, where ultrasonic and optical sensors achieved vehicle spacing of 30cm. (iv) The Morgantown system was built at the Morgantown campuses of West Virginia University (WVU), starting in 1970, by a consortium led by Boeing Vertol as a government-funded experiment in PRT systems. The track with a length of 5.8km and 5 stations entered operation in 1975. The site includes 73 vehicles, each with room for 20-passengers. Due to the large size of the vehicles the system runs in Group Rapid Transit (GRT) mode; that is, the vehicle stops at a station if any of the passengers wants to alight. The rail-based electric vehicles have power pick-ups and run up to 50km/h at 15s headway. (v) Raytheon and the Northeastern Illinois Regional Transportation Authority have developed a prototype PRT. The system, PRT 2000™employs four-passenger vehicles with linear induction motors, running on rails with headways as short as 2.5 seconds at 48km/h.

The reasons why the earlier PRT systems did not find a wider market are manifold. With most projects there have been some combinations of technical problems, design errors, escalating development costs, lack of political and financial support, lobbying by and funding competition with conventional transport companies, safety certification issues and bulky track designs. The book from C. Burke (4) describes the reasons behind the failure of many PRT developments, B. Latour (12) gives a deeper insight into the techno-socio-political development of the Aramis project and many interesting details on the Kabinentaxi project can be found in the biography of the lead engineer, Klaus Becker (3).

From the early PRT projects, only the Morgantown system is in operation to date. The daily ridership is approximately 16,000 passengers, fares of 50 c$ cover 60% of costs and system reliability is 98.5% (measured from 1975 to 2005). In 2014, the control and communication system of the Morgantown PRT has been modernized.

In recent years, the following PRT system have been developed and are in public operation until to date: (i) In 2010, the Dutch company 2getthere has completed a PRT system in the newly built "sustainable" city of Masdar City, UAE. The 1.4 kilometers long track connects a car-park with the University campus. There are ten battery powered, four-passenger vehicles circulating at a maximum speed of 40km/h and a minimum headway of 6s. The vehicles with rubber tires are actively steered along "virtual routes", where the position of the vehicle is verified relative to magnets embedded in road surface. The system is operating 18h a day and system availabilities exceeds 99.6% every year. (ii) The Heathrow airport PRT system, constructed by ULTra LTD, became operational in 2011, connecting Terminal 5 with a business car park. The system consists of 21 vehicles, a total of 3.8 kilometers of one-way guideway, and three stations. The vehicles with rubber tires are battery powered, actively steered by laser scanning of guideway borders. The system transports in average 800 passengers per day with an availability of 99.7%. (iii) The SkyCube PRT in Suncheon Bay, South Korea features a bi-directional guideway of 4.6km and 40 vehicles with seating capacity for 6 to 9 passengers. Electric vehicles have power pick-ups and run on rails at a speed of 45km/h and a headway of 4s.

9.2.3 What are the differences between PRT and driverless cars?

PRT and driverless cars (automation level>4 cars) have many characteristics in common: both systems are fully automated, provide individual transport, allow empty vehicle trips and hence the possibility to offer on-demand, vehicle sharing services.

The main functional differences are: (i) PRT vehicles use only exclusive guideways and do not mix with any other transport mode; (ii) PRT network topology is usually very simple (i.e. single lane, intersection with only two incoming, one outgoing guideway); (iii) the PRT guideway is standardized throughout the network and plays a more important role than roads do for driverless cars – the guideway contains sensors and communication devices and serves as passive guidance (or at least as a protection to prevent vehicles from leaving the track) and the guideway can also provide electric power to the vehicles. Furthermore, PRT is a public transport and all vehicles are shared.

Some systematic advantages of the PRT approach are obvious: the automation of PRT is much simpler compared with the automation of cars, moving in complex and uncertain environments. This is probably the main reason why PRT systems are in public operation many years before driverless taxis will be available. Simplicity and known operating environments is also necessary to guarantee predefined safety levels, shorter headways and ultimately higher capacities, even if roads

were dedicated to driverless cars only (10). The exclusive right of way on PRT guideways allows a full control over all traffic flows, and the predictability of trip times becomes more precise with respect to mixed road traffic. The latter plays an important role in avoiding traffic jams and in providing the right amount of vehicles at the stations (as we shall see later in this article). Power pick ups can replace batteries, which may be more cost effective at urban areas with higher vehicle densities.
The systematic disadvantages of PRT is that the guideway infrastructure needs to be build from scratch. It is obvious that a PRT network is unlikely to reach the density of the road network, because of the high investment costs (see (19)), geometric conditions or visual impact. Due to their standardized size, PRT vehicles can transport persons and freight of a certain volume and weight, but PRT does not nearly allow the wide range of vehicles as the road does. However, it remains an interesting question whether in urban areas with high vehicle densities, an area-covering PRT network where stations are place in an acceptable distance to most origins and destinations, and thus absorbing a large part of motorized individual traffic, would become financially viable.

9.2.4 Why micro-simulating PRT with SUMO?

It has been shown in a European stated preference survey that the majority of urban car users would be willing to change to public transport if a PRT-type service were provided (22). As PRT could compete with the private car, a city wide PRT network would have a significant impact:

- in-town parking is no longer needed, as 100% of vehicles are shared by design
- for shorter distances, walking and cycling would be encouraged as motorized traffic and traffic lights could be largely removed from road level
- the citizen's health and live would improve as there may be more green space, less toxic emissions, less noise pollution, and less stressful because no perceived danger on the roads.

However, such major changes are difficult to assess as mobility is a complex system with many interdependencies. It is further needed to quantify and compare the benefits and costs with the present situation and with alternative solutions. SUMO could help to simulate and consistently compare scenarios of before during and after the implementation of PRT networks in a city. Besides the impact analyses (i.e. air-pollution, energy use and noise) some particular issues are:

- what kind of infrastructure would be needed to absorb a large part of motorized traffic (capacity)?
- what would be the changes in door-to-door trip time with respect to the present situation?
- how much would pedestrians and cyclists benefit from such a system?

The capacity issue has often been seen as an obstacle for PRT systems to become effective and economically feasible. As PRT vehicles carry typically only 1-2 passengers, the time headway must be very small in order to allow acceptable transport capacities. This is the main reason, why PRT has been shown to be economically feasible for special applications, see (19) as example.
Furthermore, any PRT network will not start as fine-meshed, city-wide network with distributed demand patterns. Instead, it is more likely that a PRT project becomes financed if it can start as an initially small network which can handle high traffic volumes, as to recover investments (for instance by connecting major activity centers).
In the past, the following strategies have been followed to overcome the capacity problem:

- Group rapid transit (GRT) with larger vehicles; during peak hours such vehicles operate like a conventional public transport line, while during off-peak hours the same vehicles would serve as PRT. This approach solves the capacity problem, but it reduces service quality during peak hours. In addition, vehicles and infrastructure would be heavier compared with PRT.

- increase network density, similar to the road network. This approach is most likely too costly and in most cities not feasible due to space constraints.
- Short, sub-second time headways would be solution of the future. Currently a safety certified technology is not available.
- Platooning of vehicles with the same destination (or intermediate destinations). Such a formation of short trains would boost capacity (18). Dynamic platooning (platooning while running) is technically challenging, while coupling of vehicles in stations would be possible with state of the art automated train coupling systems.

This work focuses on the implementation and simulation of PRT features in SUMO. The main aim is the implementation of some basic PRT features in SUMO and the evaluation of a novel PRT vehicle redistribution system. Furthermore, a simple platooning strategy is described and tested to increase capacity. The aim is not to model precisely the PRT vehicle dynamics, because those will differ dependent on the concrete PRT technology.

9.3 PRT technologies

This section provides a short overview on important technical concepts that need to be modeled by SUMO. Again the focus in this work is not on precisely matching PRT dynamics but on the vehicle management.

9.3.1 A brief overview on PRT vehicle control

Automatic PRT vehicle control systems have been studied since the 70$^{\text{th}}$ (5). In the past decade, automated vehicle research has been focused on Automated Highway Systems (AHS) and more recently on self-driving cars, see (8) for a comprehensive overview.
PRT needs a control system for short time headways to meet simultaneously criteria for safety, string stability comfort and speed transitions. In literature, there are three main longitudinal control-strategies: (i) constant distance separation, where vehicles are ordered to maintain a fixed distance to the vehicle in front; (ii) constant time separation, where vehicles are ordered to maintain a constant time headway and (iii) constant safety separation, where a vehicle must always keep a sufficient distance to the lead vehicle, such that no crash occurs in an emergency case.
The bulk of research on automatic vehicle control deals with constant time headway spacing policy, which is usually adopted by AHS systems. see for example (24). Nevertheless, constant safety considerations play a role in the design of control laws for platoon-join manœvers (14). Fewer literature can be found on longitudinal controllers that follow the constant safety separation policy. For PRT systems, constant safety spacing policy is important as conventional rail legislation may apply. The well known "brick-wall criteria" (the train must not crash into a train in front, even though it stops suddenly) could be a requirement, at least for the first PRT implementations.
One approach is the use of safe trajectory generators, see (25). Another approach to minimum safe distance is to use a feedback controller, where the minimum distance criteria is implicitly guaranteed by a non-linear function in the feedback-loop of a controller as proposed by (16). However, for the current work, the default SUMO car-following model, called Krauss model, is used, see (11).

9.3.2 A brief overview on PRT stations

PRT stations play an important role in the simulation as they unite different functions:

- access to the PRT by persons: stations connect to the road network of SUMO, such that persons can walk (or cycle) to take a PRT vehicle.

- alighting/boarding procedure: the stop controls arriving vehicles, let passengers alight and board persons who wait at the platform.

- programming of vehicles: the stations "programs", with the support of the vehicle management, the departing vehicle. This means the vehicle's destination station, route to this destination and precise departure time is set. Departing vehicles can be occupied or empty.

- Starting of vehicles: programmed vehicles are started to leave the station at the programmed departure time.

- Platooning. If a stop is allowed to create vehicle platoons (stop is in platoon mode), it can couple vehicles with identical destination.

PRT stations are usually off-line, such that only those vehicles must stop which end their trip at this station. All other vehicles bypass the station on the main line. Off-line stations allow non-stop origin-to-destination service and to avoid capacity bottlenecks at stations.

There are different possible layouts for PRT stations, each with its own performance characteristics, advantages and disadvantages with respect to particular PRT technologies. For an extended discussion, see (9; 21). The layout with the highest throughput and highest space efficiency (passengers boarding per time per area) is the serial-line station with alighting and boarding zones as shown in Fig. 9.1.

9.3.3 A brief overview on PRT vehicle management

The PRT vehicle management is a centralized service that optimizes traffic flows on a PRT network. In particular, this means to guarantee that the vehicle flow into all stations corresponds to the passenger flows, demanding vehicles at each station. A second flow-control task is to avoid link flow exceeding capacity limits by appropriate deviations, but this is not subject of the present study. The redistribution problem of PRT vehicles has been addressed in literature since many years, see (9) and theoretical performance limits have been found (13; 20; 23). The problem of dispatching a given number of vehicles to a given number of stations at certain time instance is a high dimensional problem and can only be solved heuristically in real-time operation conditions. In practice, most algorithms resort to a sequential allocation of vehicles, taking into account the travel time C_{od} and the current travel demand-level at destination d, where o is the station of origin of the vehicle and d is the potential destination station.

The demand level at station d may express the total waiting time, the maximum waiting time or the number of waiting passengers. The principle strategy is to send a vehicle to the station d with the highest demand from the nearest station o (by minimizing C_{od}). One can speed up the optimization time by reducing the number of feasible destination stations, see (6).

However, in larger networks, the time delay between the instance of vehicle scheduling and the instance of arrival at the destination station is not negligible. For example it can happen that a destination station with high demand during scheduling time, is in low in demand, when the vehicle eventually arrives, because other vehicles may have arrived before and decreased the number of waiting passengers. The consequence is that the vehicle flows on the network become unstable and begin to oscillate between stations. This problem has been addressed in literature and two solutions have been proposed (2):

- Instead of considering only the demand at the destination station, the "demand surplus" is taken into account. The demand surplus has been defined as
  the number of passengers waiting + number of passengers expected to arrive during triptime
  - vehicles already at this station - vehicles (occupied or empty) on their way towards this station

- Instead of scheduling empty vehicles only at their departure station, vehicles may be rerouted also on their way at predefined intervals, if necessary; this is to react to changes in demand surplus during the vehicle trip.

For the vehicle management of the present work, a third approach has been developed in order to optimize and stabilize the vehicle redistribution, as suggested in (15): in addition to the surplus at the destination station (as suggested in (2)), also the estimated future person-flows and scheduled vehicle-flows are considered.

9.4 The SUMO components of PRT simulation

Different software "components" have been created and developed which allow an realistic simulation of PRT networks.

- PRT vehicles in which persons ride to their desired destination. Empty vehicles will be directed to stations with hight demand.
- PRT stations, which are special edges of the SUMO network, where persons can access the PRT vehicles.
- PRT vehicle management, which optimizes the routing of the entire fleet. The most important function of the vehicle management is the empty vehicle routing.

The functionality of PRT stations and the PRT vehicle management are implemented in Python code. The code is provided as a plugin of the SUMOPy simulation suit (17). PRT vehicle are controlled only at PRT stations through the TRACI Interface, see below for details. Persons are currently not controlled interactively but follow predefined plans, see Sec. 9.5 for a brief description of the demand generation.

9.4.1 PRT vehicles

PRT vehicles are represented as E-vehicles with a special parametrization. As mentioned previously, the present study is focused on PRT simulation which means small-size PRT vehicles, where all passengers have a unique destination.
As stated earlier, the PRT vehicle's control algorithm is approximated by SUMO's Krauss model. As all PRT control algorithms are proprietary, the Krauss model serves as a good approximation as important parameters, such as maximum values for speed and acceleration as well as reaction time can be specified according to technical data provided by the PRT suppliers.
Vehicles can run as individual vehicles or in a platoon. Whether a vehicle runs in individual or platoon mode is determined by the stop, short before the vehicle departs, as described in Sec.9.4.2. Individually running vehicles and platoon leaders have the same set of parameters. Instead, the control parameters of platoon followers (all vehicles behind the leader in a platoon), have a different set of control parameters in order to closely follow (in the ideal case physically touch) the vehicle in front. Table 9.1 shows the vehicle attributes used for individual vehicles, leaders and followers. The values, from (20), have been chosen match parameters from real PRT systems. Values for the follower mode have been tweaked manually such that vehicles follow closely and smoothly their vehicle in front.
Macroscopic control instructions like start or stop, or setting a new destination, or setting follower mode is performed in stations. Between stations, vehicles run without any further interaction through TRACI.

9.4.2 PRT network and stations

The PRT network is implemented as a uni-directional, single-lane network with access restictions (at present the vClass “custom1”). Edge speed limits have been set to $20 m/s$ as to allow all PRT vehicles to run at full speed. The PRT edges are joined by zipper-type nodes. Typically a node has either 2 ingoing edges and 2 outgoing edges or vice versa.

The PRT station, as shown in Fig. 9.1, has been realized by a single link with two lanes: the leftmost lane is accessible for pedestrians (and bikes if desired) while the second lane is accessible by PRT vehicles only. The pedestrian lane is connected to the surrounding road network and allows persons, (and bicycles), to access the station platform.

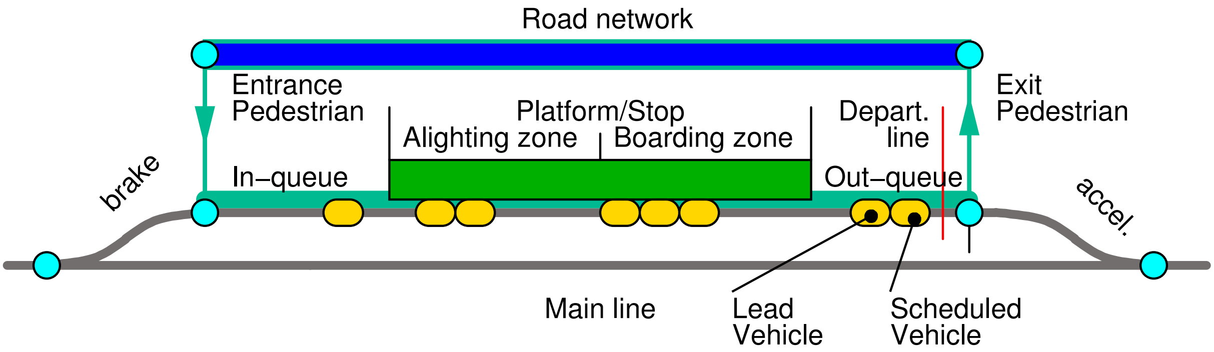


Figure 9.1: Scheme of a PRT station as implemented in SUMO.

Table 9.1: Vehicle type attributes of a PRT vehicle. Values are given when vehicle runs individually or as a leader of a platoon (left column) and for a vehicle when it runs as a follower in a platoon (right column).

Attribute	value as leader	value as follower
vType name	PRT	
Length $[m]$	3.5	
Width $[m]$	1.6	
Height $[m]$	1.7	
Passengers	1	
Capacity	1	
Max. speed $[m/s]$	20	
Speed factor	1	2
Speed dev.	0.01	
Max. accel. $[m/s^2]$	2.5	3.5
Max. decel. $[m/s^2]$	2.5	
Reaction $[s]$	0.8	0.2
Driver	1	0
Min. gap $[m]$	0.5	0.01
boarding time $[s]$	8	
loading time $[s]$	90	
vClass	custom1	
Emission type	HBEFA3/zero	
Impatience	-1000	
GUI shape	evehicle	

The station platform (alighting and boarding zone) is realized by a SUMO stop elemen and has generally room for N_B vehicles. Places where vehicles stop and passengers alight or board are called *berth*. The total platform length is $L_S = N_B L_B$ meters, where L_B is the length of one berth (typically a few centimeter greater than a vehicle length). In such a station, $\frac{N_B}{2}$ passengers can alight and board simultaneously. An efficient PRT station requires queues at the entrance and exit of the station are necessary in order to quickly move in vehicles, as soon as all passengers have alighted, and to move out vehicles, as soon as all passengers have boarded. Therefore, if the station should work at capacity limits, both buffers should have at least room for $\frac{N_B}{2}$ vehicles. In general, standard queuing theory, based on average vehicle flow rates, can be applied to determine the minimum buffer length.

In a serial-line station, entering vehicles stop first at the alighting zone where passengers exit the vehicle. Then the empty vehicles are forwarded to the boarding zone, where passengers board.

Passengers do wait at berths of the boarding zone. When a PRT vehicle stop at a berth then one passenger, who is waiting at the correponding berth will board, as the passenger capacity of a PRT vehicle is set to one. If multiple persons wait at the same berth then it is currently not possible to control which person is boarding the vehicle. This means a FIFO person queue at stations cannot be inforced and realistic person waiting time cannot be determined.

When passengers have boarded, the vehicle moves on to the “departure line” near the exit of the station (see Fig. 9.1). At the departure line the vehicle stops and waits to get started. The vehicles receives instructions from the vehicle management on the destination stop, which is the destination of the passengers, and an optimized departure time instance. In case vehicles in the boarding zone are waiting and occupied vehicles are behind, then these waiting vehicles are pushed out after a defined time-out.

Before a vehicle reaches the departure line, it gets scheduled from the vehicle management. Scheduling means that the vehicle receives its optimal destination station d_{opt} and its optimal departure time T_{opt}, see procedure in Sec. 9.4.3. When a vehicle eventually arrives at the departure line it stops and waits until time has reached the the optimum scheduled departure time T_{opt}. Then the vehicle accelerates and starts his actual journey. This particular vehicle handling is necessary to ensure a predefined departure time. Empty vehicles can also get scheduled when they are waiting in the boarding zone, see pulled empty vehicle scheduling in Sec. 9.4.3.

If enabled, PRT stations can perform platoons by concatenating vehicles with identical destinations. This concatenation takes place in the out-queue of a station: examining the out-queue backwards, if a scheduled vehicle has a vehicle with the same destination in front, the it will become a follower, otherwise it will remain a leader. In platoon mode, all vehicles in the out-queue will start together instead of waiting at the departure line. This means the precise timing is lost.

9.4.3 PRT vehicle management

Within the present simulation framework, the vehicle management provides the following principal functions:

schedule pushed occupied vehicles: when a vehicle is occupied, and ready to leave from a PRT station, then the vehicle management determines the route to the desired destination of the passenger and the best departure time.

schedule pushed empty vehicles: when a vehicle is empty and is forced to leave a station, then the vehicle management determines the route to the optimal destination station and the best departure time.

schedule pulled empty vehicles: when stations are in need for empty vehicles due to high demand, then the vehicle management locates the optimum station from where to draw an empty vehicle and determines its optimal departure time.

In this paragraph, the identification of the optimal destinations (in case of empty trips) and the optimal departure time are explained for these 3 cases. As mentioned above, the vehicle redistribution makes decisions based on the estimated future person-flows and scheduled vehicle-flows, in addition to the demand surplus at the destination station. For this purpose different quantities are measured in time-intervals of duration T_S, which is in the order of tenth of seconds. This means the time interval k refers to the time period $kT_S \leq t < (k+1)T_S$.

The demand surplus matrix $\mathbf{D}[d,k]$ represents the passenger surplus at station d at time interval k which is the number of persons at station d minus the sum of the already scheduled vehicles to station d at time interval k minus half of the vehicle available at the station during interval k. Instead of the number of waiting persons, the total waiting time at a station would be preferable. But as mentioned above, the wait times of passengers are not currently not measurable reliably.

The person flow matrix $\mathbf{W}[d,k]$ represents the number of persons arriving during time interval k at station d, while the vehicle flow matrix $\mathbf{F}[d,k]$ represents the number of arriving vehicles during time interval k at station d.

The future person flow matrix $\hat{\mathbf{W}}[d,k+\kappa]$ must be estimated. For the present study it is assumed that the person flow into station d does not change significantly in the future, hence $\hat{\mathbf{W}}[d,k+\kappa] = \mathbf{W}[d,k]$ for any positive κ. In more sophisticated models, one could consider demand evolutions based on past experiences or bookings through cell phones. The scheduled vehicle flow matrix $\hat{\mathbf{F}}[d,k+\kappa]$ represents the arriving vehicles at station d in κ time intervals ahead. As shown further below, the scheduled vehicle flow matrix is determined during the scheduling process. The principle how $\hat{\mathbf{F}}[d,k]$ is generated shall be explained by an example: if a vehicle departs at instance k at station o, heading for station d then it would arrive at station d during time-step $k + C_{od}/T_S$ and increasing the vehicle flow during this time step by $\frac{1}{T_S}$. This means the scheduler adds $\frac{1}{T_S}$ to the flow estimation $\hat{\mathbf{F}}[d,k+C_{od}/T_S]$. Doing so for each scheduled vehicle, the flow estimation matrix $\hat{\mathbf{F}}[d,k+\kappa]$ will eventually be completed as κ approaches zero.

The task of the scheduler can be seen as a control function: The difference between the person flow and the vehicle flow must be controlled to zero and simultaneously, the remaining waiting persons must be controlled to zero. Similar to a linear control function, the objective function for station d at κ time-intervals in the future, is formulated as

$$z(d,\kappa) = \Big(\hat{\mathbf{F}}[d,k+\kappa] - \hat{\mathbf{W}}[d,k+\kappa] - K_D\mathbf{D}[d,k]\Big)\exp(-\lambda\kappa T_S) \tag{9.1}$$

where the constant K_D is a weight to balance the flow difference with the demand surplus. The travel time weight $\exp(-\lambda\kappa)$ with the positive time constant λ is introduced to penalize remoter destinations with higher travel times κ. This purely heuristic function is subject to future improvements. Note that in a steady state, the vehicle flows equal the passenger flows and the demand surplus is zero or close to zero.

Regarding the algorithm, the vehicle management schedules periodically the *lead vehicle* in each PRT station $o \in \mathcal{S}$. The lead vehicle of station d is the last vehicle which is located in the boarding zone or out-queue and that has not yet been scheduled (see also Fig. 9.1). If a lead vehicle has been scheduled, then the vehicle behind becomes the new lead vehicle of that station. Every $2s$ the vehicle management tries to schedule lead vehicles from all stations. Assuming a scheduling starts at time time t (or interval $k = t/T_S$) then the scheduling algorithm repeats the following processes until there are no longer lead vehicles in any of the stations:

schedule pushed occupied vehicles: For all station of origin $o \in \mathcal{S}$ where the lead vehicle is occupied, schedule the lead vehicle as follows: The vehicle's destination station d is determined by the desired destination of the person in the vehicle. With a fixed destination d, the departure delay $\Delta \in [\Delta_{\min}, \Delta_{\max}]$ minimizes the objective function $z()$, thus

$$\Delta_{\text{opt}} = \text{argmin}_{\Delta} z(d, C_{od}/T_S + \Delta)$$

with the search interval $[\Delta_{\min}, \Delta_{\max}]$. This means the scheduled departure time becomes $T_{\text{opt}} = t + \Delta_{\text{opt}} T_S$. The earliest possible departure time is $t + \Delta_{\min} T_S$ should be large enough as to allow the vehicle to reach the departure line, while the latest possible departure time $t + \Delta_{\max} T_S$ should not be low enough, such that the station capacity does not get compromized.

schedule pushed empty vehicles: For all station of origin $o \in \mathcal{S}$ where the lead vehicle is empty, schedule the lead vehicle as follows: Determine the optimal destination $d \in \mathcal{S}, d \neq o$ and the optimal number of delay intervals $\Delta \in [\Delta_{\min}, \Delta_{\max}]$ by minimizing

$$d_{\text{opt}}, \Delta_{\text{opt}} = \text{argmin}_{d,\Delta} z(d, C_{od}/T_S + \Delta)$$

Optimum departure time becomes again $T_{\text{opt}} = t + \Delta_{\text{opt}} T_S$.

schedule pulled empty vehicles: The strategy identifies first the stations without demand. Such stations are typically depots. Then the empty lead vehicle trips gets scheduled, similar to a pushed empty vehicle scheduling. Thus, for all station of origin $o \in \mathcal{S}$ without waiting passengers and where the lead vehicle is empty and waiting in the boarding zone, schedule the lead vehicle as follows: Determine the optimal destination $d \in \mathcal{S}, d \neq o$ and the optimal number of delay intervals $\Delta \in [\Delta_{\min}, \Delta_{\max}]$ by minimizing

$$d_{\text{opt}}, \Delta_{\text{opt}} = \text{argmin}_{d,\Delta} z(d, C_{od}/T_S + \Delta)$$

with optimum departure time $T_{\text{opt}} = t + \Delta_{\text{opt}} T_S$.

9.5 Test scenario

A scheme of the test network is shown in Fig. 9.2. There are 10 PRT stations located in three zones: zone and 3 are peripheral zones with moderate traffic flows and zone 2 is zone in central location with the highest traffic flow.

The number of persons traveling between the different zones (see Fig 9.2a) are given by the OD matrices in Tab. 9.2. Two demand scenarios are tested: a low demand scenario (a) and a high demand scenario (b). Both demand patterns represent an asymmetric demand scenario: Stations in zone 1 have more departures than arrivals, in zone 2 there are more arrivals than departures and stations in zone 3 there are only arrivals. Such an asymmetric demand is challenging for the vehicle redistribution because it must reequilibrate the vehicle flows between the station with different demand levels, such that all passengers are served in all stations.

Table 9.2: Zone-to-zone OD matrices for a 1h time period. (a) Low demand scenario, (b) High demand scenario.

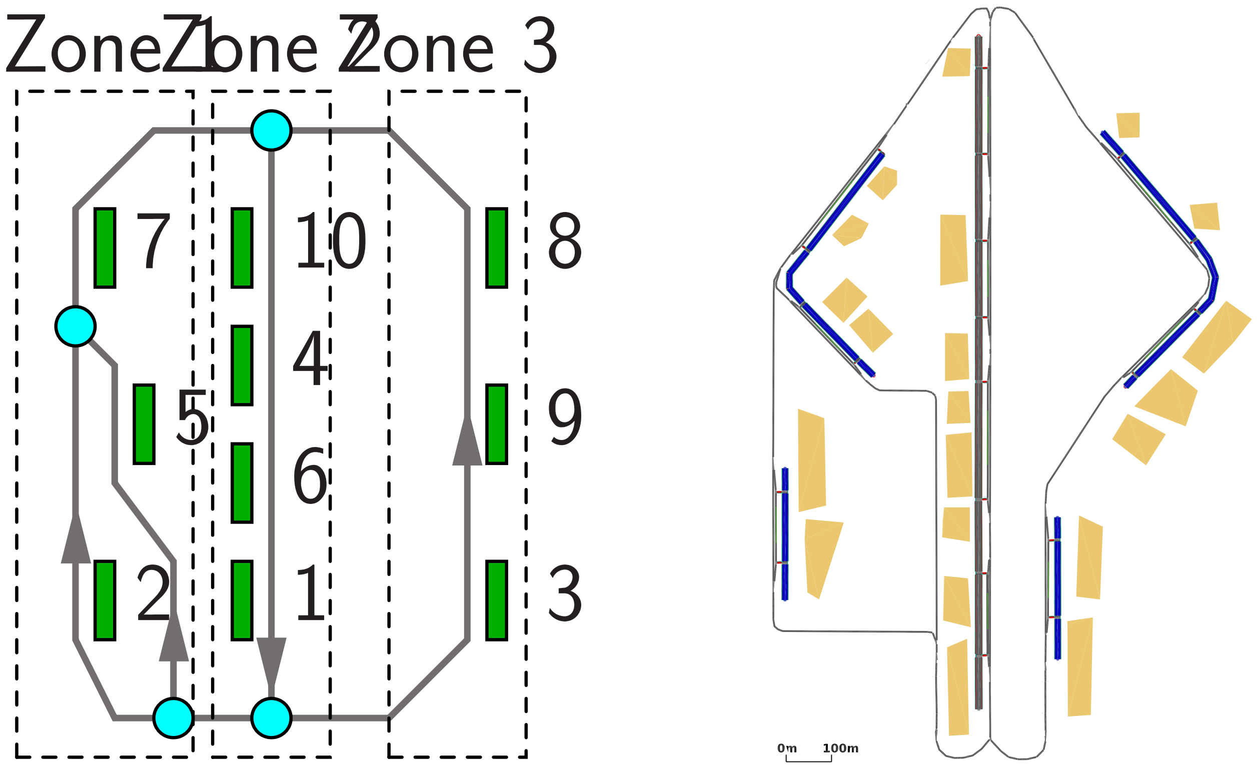
Zones	1	2	3
1	0	800	0
2	400	0	0
3	0	800	0

(a)

Zones	1	2	3
1	0	1200	0
2	600	0	0
3	0	1200	0

(b)

The entire PRT simulation is embedded in an activity based demand model, where door-to-door-plans are generated for each person of the virtual population. The virtual population is generated from the OD matrices in Tab. 9.2. This means for each zone-to-zone flow in the OD matrix, the corresponding number of persons are generated. Each person receives a "home"-building in the zone of origin and



(a) Scheme of a PRT test network.

(b) Entire SUMO test network.

Figure 9.2: PRT test network: (a) a scheme with station IDs and zones, and (b) the detailed SUMO network including access roads and buildings to the right.

an "activity" building in the zone of destination. The persons are distributed over the buildings according to the area of the respectove buildings. Once the buildings of origin and destination are specified for each person, the door-to-door plan, which consist of several walk and ride stages, are generated: each plan starts on the pedestrian network-edge which is closest to the home building; the person then walks to the nearest PRT stop, where she/he waits for a PRT vehicle at a randomly chosen berth position on the platform; the person rides on an PRT vehicle to the stop which is nearest to the activity building; the person walks from the PRT stop to the activity building.

9.6 Simulation results

For all simulations the control constants for the vehicle redistribution have been $T_S = 10s$, $K_D = 0.01$, $\lambda = 0.005/s$, $\Delta_{\min} = 1$ and $\Delta_{\max} = 3$, and all stations are not in platooning mode, unless stated otherwise. These parameters have been found by manual optimization: the parameters have been tweaked such that the total number of person waiting at stations is minimal. It turned out that simulation results are not very sensitive to parameter changes. Of course, only a systematic parameter search would be able to find an optimum set of parameters. The travel times C_{od} are free-flow, shortest path travel times from station o to station d. In a future implementation, also real-time measurements of link travel times could be used to update C_{od}.
First the low demand scenario has been simulated for $1h$ with 250 vehicles. The vehicle flows in the most critical part of the network is shown in Fig. 9.3. The flow in the central parts is 1847 vehicles per hour. Station 7 showed with 405 departing vehicles the highest number of passenger boardings.
The number of waiting passengers over time at each of the 10 stations is shown in Fig. 9.4. One can see that the vehicles are able to prevent a constantly increasing number of waiting passengers at all stations, which means that the 250 vehicle are sufficient to cope with the demand. Simulations with

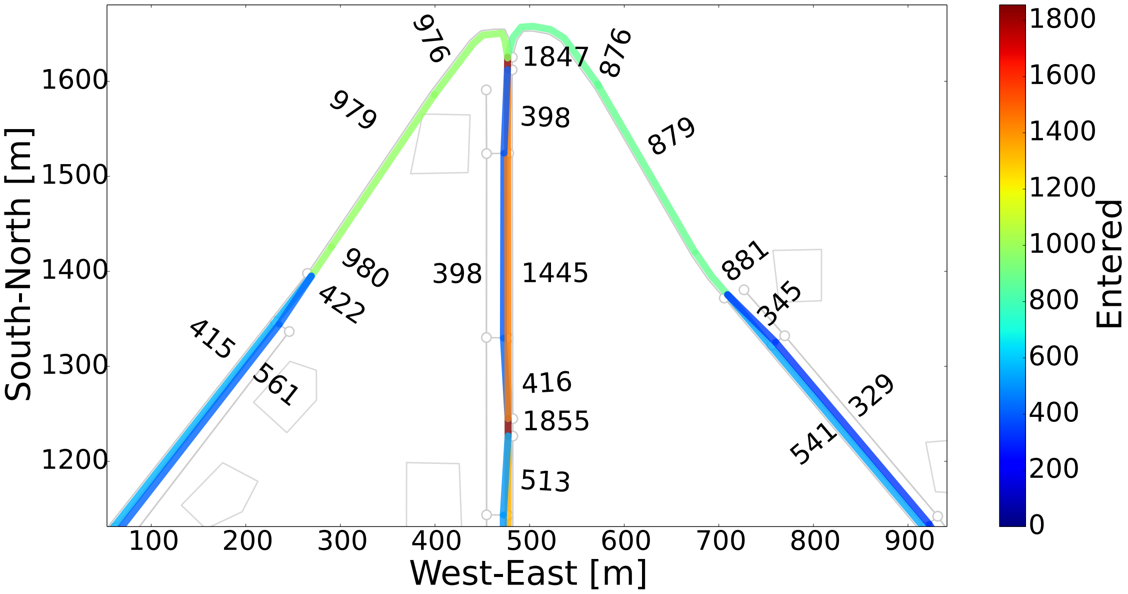


Figure 9.3: Vehicle flows with low demand scenario at the entrance of the central network branch, the most critical point.

less vehicles (200) resulted in an increasing number of waiting passengers and simulations with more vehicles (350) resulted in an excessive number of empty trips. In reality vehicle depots should absorb vehicles if too many are in circulation and release vehicles, when needed. However, depots are not included in the current simulation.

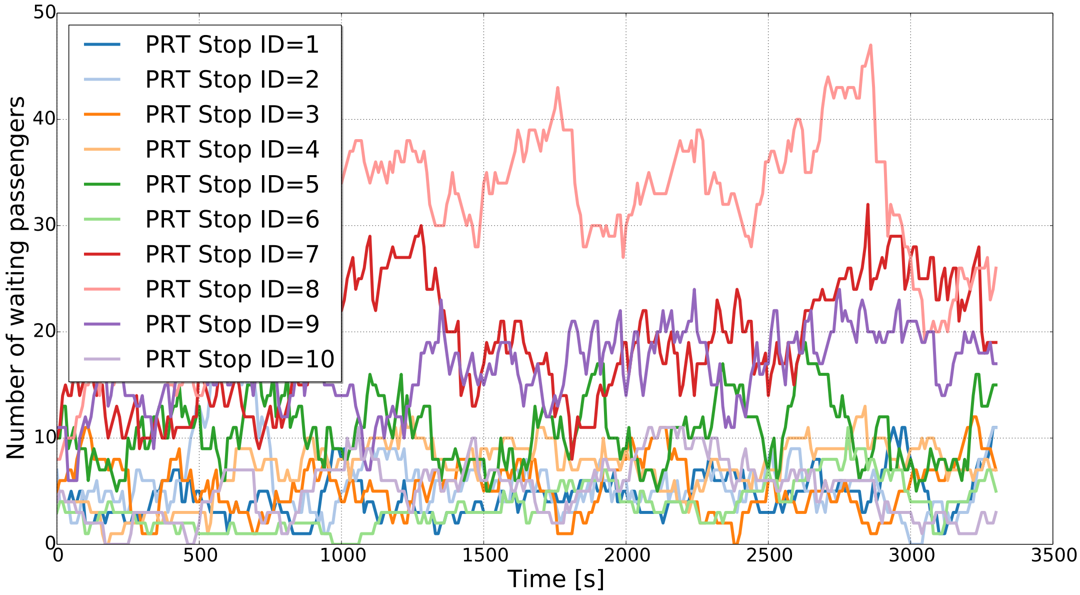


Figure 9.4: Number of waiting persons at PRT stations over time during low demand scenario.

Fig. 9.5 shows the in-flow results at station 8: This is number of waiting passengers over time t as well as the vehicle flows $\mathbf{F}[8,k]$ and passenger flows $\mathbf{W}[8,k]$ over $k = \mathrm{int}(t/T_S)$. The flows appear to be discretized at different flow-levels, because the flow averaging takes place over $T_S = 10s$ only. This means the different levels simply indicate the few persons or vehicles which arrived during one interval. Note that the average flow of vehicles (350/h) is near the average flow of passengers

(310/h). The vehicle flow is slightly higher, because not all vehicles which left the station have been occupied. For instance, a vehicle waiting at the boarding zone for a passenger can get pushed out by occupied vehicles following behind. Figure 9.5 shows also the scheduled flow $\hat{\mathbf{F}}[8,k]$ over $k = \int(t/T_S)$.

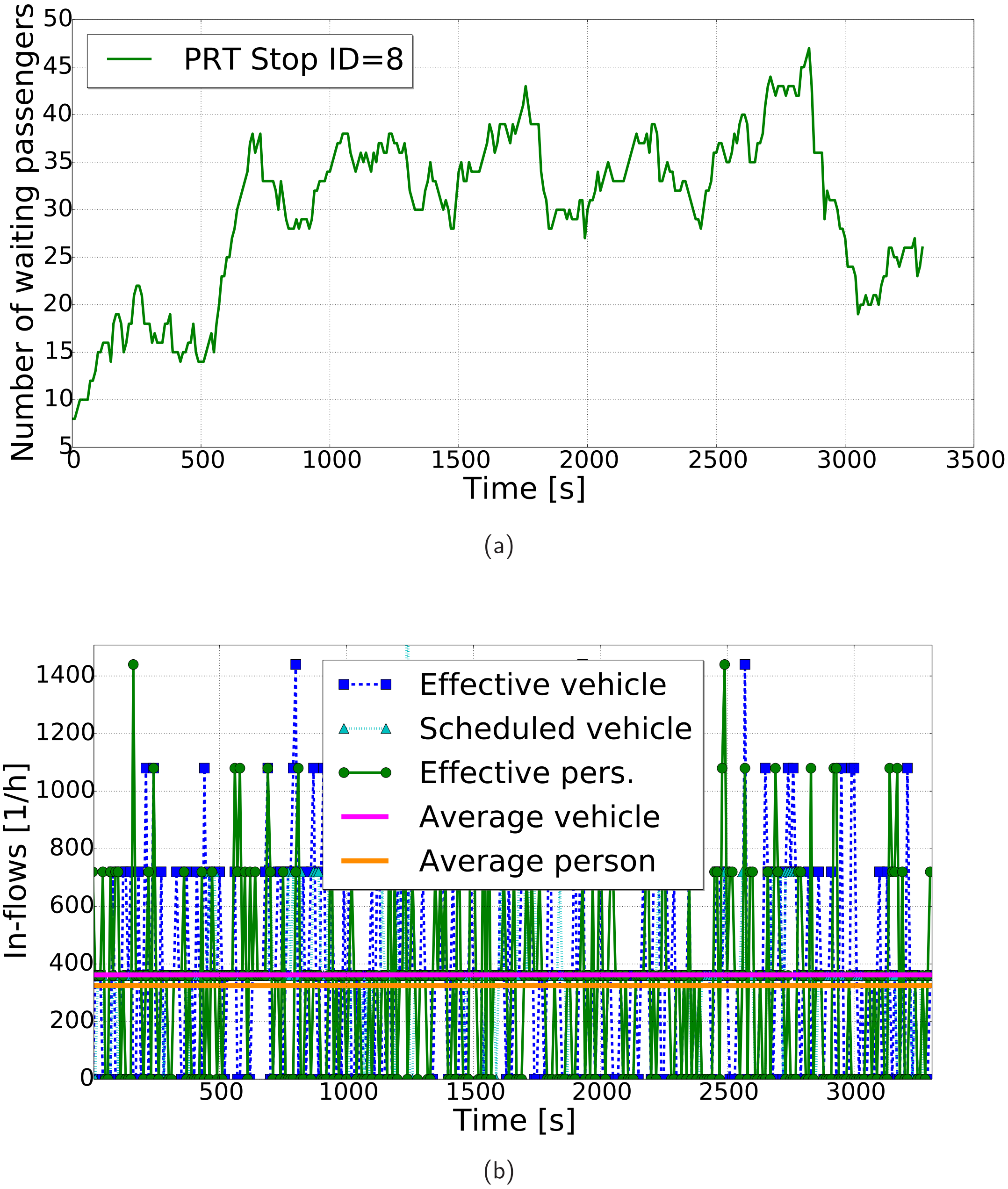


Figure 9.5: (a) Number of waiting persons over time at station 8. (b) Effective and scheduled vehicle flows as well as person flows into station 8.

The flows scheduled by the vehicle management and the effective flow at the stations can be better compared at a station with low demand, such as station 3, see Fig. 9.6. In particular, it can be seen that scheduled and effective vehicle arrival times do not completely coincide. The effective arrival times of the vehicles seem scattered around the scheduled arrival times. These delays, which are in the range of $30s$, can be attributed to the asynchronous nature of the vehicle control strategy and the assumption of free-flow speeds when determining the travel time matrix C_{od} on which scheduling is based.

Note that at this low demand station, the average vehicle flow is significantly higher than the average passenger flow. The reason is that if only few passengers are waiting and are sparsely distributed along the boarding area of the platform then the vehicles were often unable to pick up the passengers. With a future SUMO versions it will be possible to control persons through TRACI. This means it is possible to make a person walk to a nearby waiting vehicle, so that the vehicle will not leave the station empty.

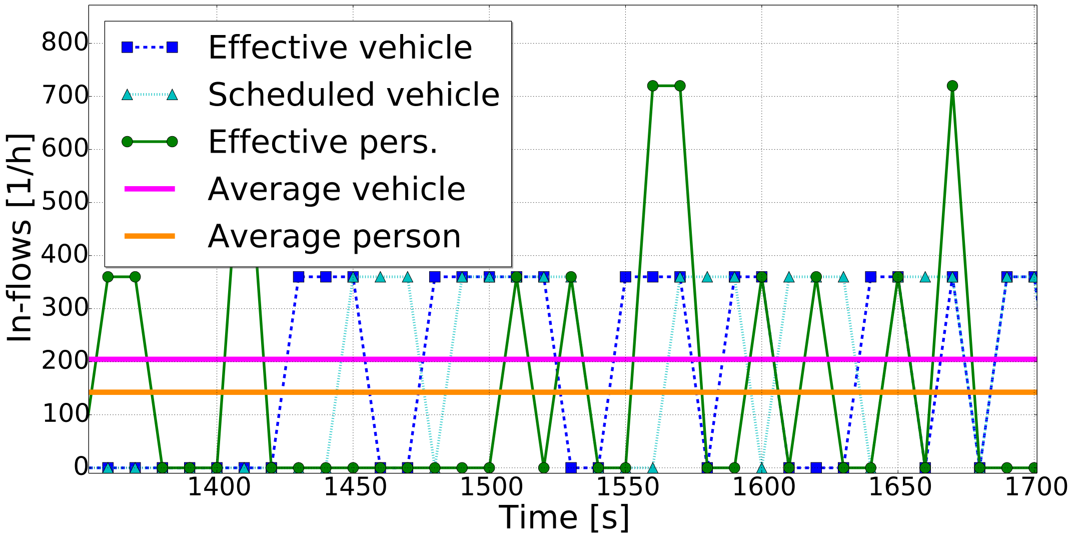


Figure 9.6: Comparison of scheduled vehicle flow $\hat{\mathbf{F}}[3,k]$, effective vehicle flow $\mathbf{F}[3,k]$ and passenger flow $\mathbf{W}[3,k]$ at station 3, $k = \mathrm{int}(t/T_S)$.

The control parameter K_D balances the contribution of the flow difference and the demand surplus to the objective function of a particular station, see Eq. 9.1. In order to demonstrate the influence of the control parameter K_D, the simulation of the low demand scenario is repeated with $K_D = 1$, which is a significant increase by 100 times. The number of waiting passengers at station 8 is shown in Fig.9.7. There are noticeable oscillations in the number of waiting passengers compared with Fig.9.5a where the flow component had a stronger influence on the objective function (with $K_D = 0.01$). Such oscillations are negative because the station will generate excessive empty vehicle trips if a swarm of vehicle does no longer find enough passengers (in particular during the declining phase of the cycle). This oscillating number of waiting persons confirms, that a vehicle redistribution becomes instable if decisions are predominantly based on the demand surplus. Hence, the flow component in the objective function appears to have stabilizing effect. The optimization of K_D is subject to further studies.

The high demand scenario has been simulated with 350 vehicles. As shown in Fig. 9.8 the number of waiting persons is slightly increasing for all stations, which means the vehicles struggle to serve the demand. The vehicle flow in the central zone reaches $2536\mathrm{veh/h}$. A further increase in the number of vehicles leads to a congestion and a break down for the vehicle flows in the network. This means the given demand is at the limit of what can be handled by the PRT network based on individually running vehicles.

As an attempt to further increase line capacity, the stations have been switched to platoon mode. This means vehicles in the output queue of stations with identical destination stations get concatenated to a platoon, as shown in Fig. 9.9a. However, with 350 vehicle running, congestions occurred at the entrance of the central line. Note that for the same scenario no congestions appear when vehicles run individually. The reason is apparently that zipper-type nodes are not designed for handling platoons and as soon as platoons approach nodes with a certain traffic density on competing branches then vehicles slowed down and congestion occurred, see Fig. 9.9b. A further systematic problem is that

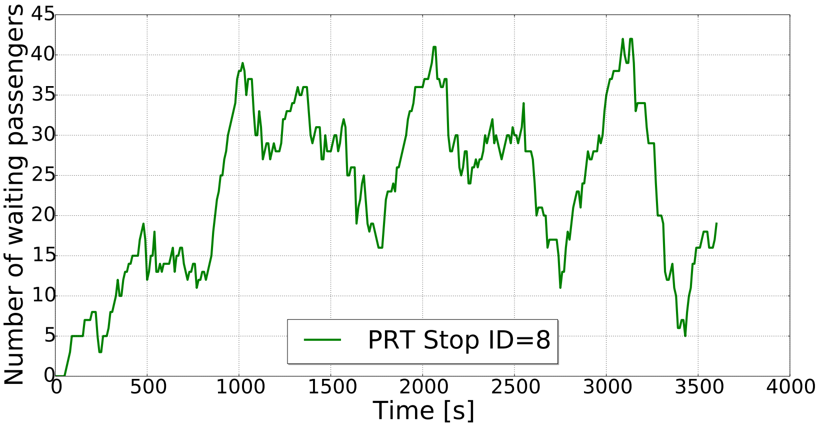


Figure 9.7: Number of waiting passengers at station 8, with control parameter value $K_D = 1$.

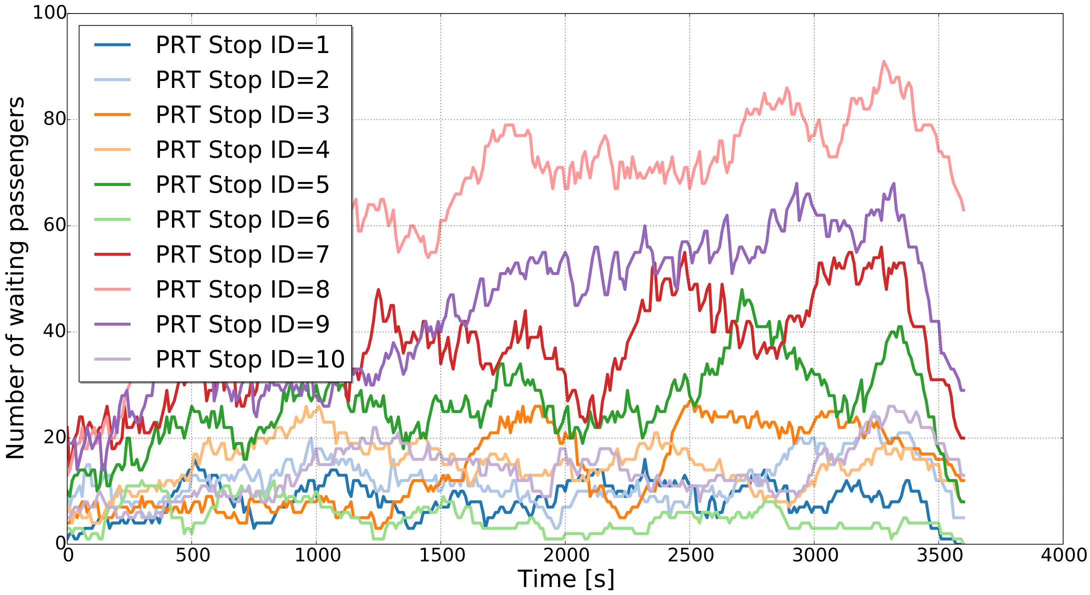


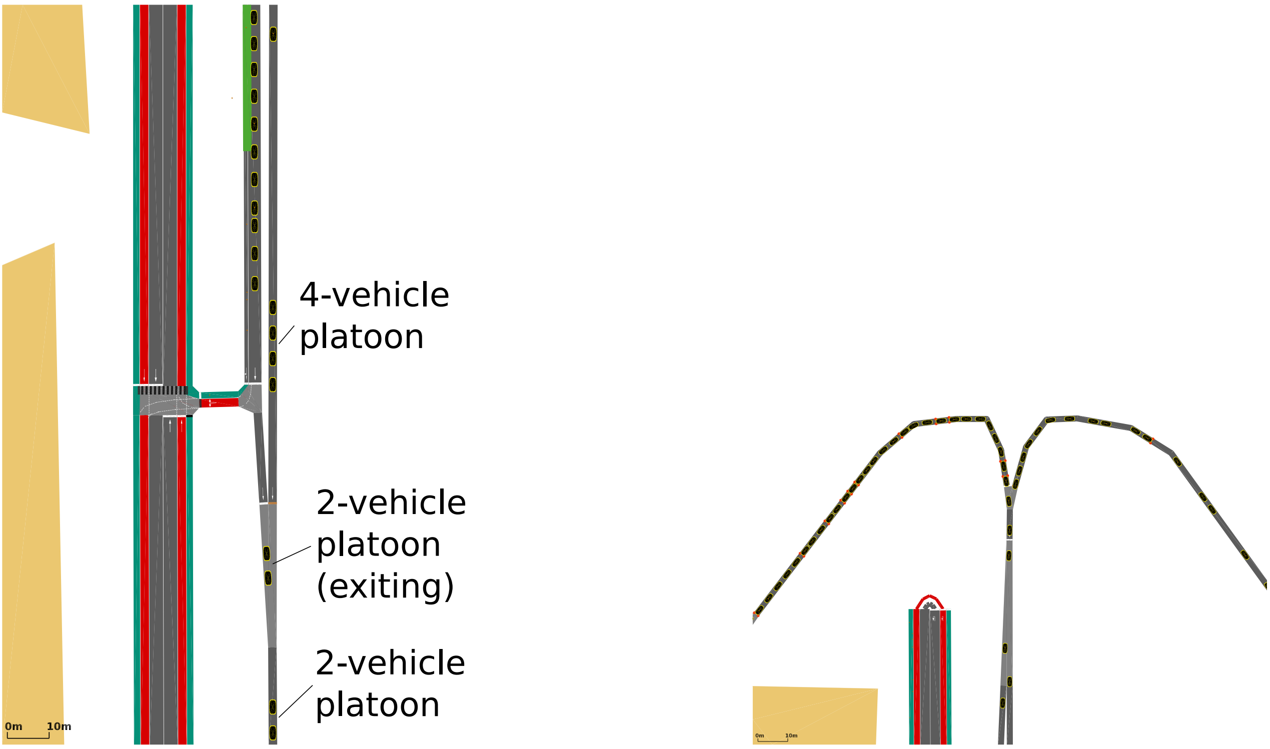
Figure 9.8: Number of waiting persons at PRT stations over time during high demand scenario.

platoon length are generally low (2-4 vehicle length platoon in the test network). Longer platoons would result in a higher line capacity.

9.7 Conclusions

An extension to SUMO has been described that allows the simulation of PRT networks, where shared PRT vehicles carry individual persons from their departure station to their destination station. The particular characteristics and functions of PRT vehicles and PRT stations have been explained. In fact PRT vehicles are controlled through TRACI in stations only. A novel vehicle redistribution algorithm, based on vehicle- and passenger-flow prediction has been devised.

The performance of the vehicle management has been demonstrated by means of a test- network under asymmetric demand conditions and with two different demand-levels. The number of vehicles in circulation and the control parameters of the vehicle redistribution have been shown to be critical to the overall performance. An attempt to increase line capacity be creating platoons of vehicles with identical destination has not been successful because congestions have been created at a critical



(a)

(b)

Figure 9.9: Simulation with platoons. (a) Platoons with different length at the station exit and on the main line. (b) Congestion that occurred with platoons at the entrance of the central line.

zipper type node.
The present work had merely the scope to explain and demonstrate the PRT features of the developed SUMO extension. There is much room for further improvements. Some of the critical issues and possible solutions are listed here:

- The vehicle redistribution's parameters need to be optimized in a systematic way, maybe with the help of macroscopic models. Also the sensitivity of the system to changing demand and different numbers of vehicles should be assessed.

- The vehicle redistribution's objective function itself could be optimized, taking into account the total waiting time of passengers instead of the number of passenger waiting. The recording of waiting times at stations will be possible with SUMO version 0.29.

- The real-time measurements of link travel times could be used to update the travel time matrix used in the vehicle redistribution algorithm.

- The proper functioning of vehicle depots (absorption and release of vehicles) need to be tested and optimized.

- The simulation with platoons requires specialized nodes: The vehicles at incoming edges should be controlled through TRACI with the objective to merge platoons smoothly. Furthermore, predictive scheduling, as used to control flows into stations could also be employed to avoid congestions at critical merge points.

- Platoon length could be increased by creating platoons to central station on a shunt yard-type structure.

9.8 Bibliography

[1] ANDERSON, J.E.: *Transit system theory*. Lexington Books, Lexington, MA, 1985.

[2] ANDRÉASSON, INGMAR: *Vehicle Distribution in Large Personal Rapid Transit Systems*. Technical Report 1451, Transportation Research Board, Washington, D.C, 1994.

[3] BECKER, KLAUS: *Träumen, erfinden, die Welt verbessern*. Haag + Herchen, 2002.

[4] BURKE, CATHERINE G.: *Innovation and Public Policy; The case of Personal Rapid Transit*. Lexington Books, D. C. Heath and Company., 1979.

[5] CAUDILL, J. and S.P THOMAS: *Fundamentals of Personal Rapid Transit*, chapter AGT vehicle kinematics: a dynamic study. Lexington Books, 1978.

[6] DASZCZUK, W.B., J. MIELŚCICKI and W. GRABSKI: *Distributed algorithm for empty vehicles management in personal rapid transit (PRT) network*. J. Adv. Transp., 50:608–629, 2016.

[7] FICHTER, D.: *Individualized Automated Transit and the City*. Technical Report, B. H. Sikes, 1430 East 60th Place, Chicago, Illinois 60637, 1964.

[8] HOROWITZ, R. and P. VARAIYA: *Control design of automated highway system*. IEEE Proc., 88(7), July 2000.

[9] IRVING, J.H.: *Fundamentals of Personal Rapid Transit*. Lexington Books, Lexington, MA, 1978.

[10] J., SCHWEIZER: *A technological comparison of Automated Highway Systems (AHS) and Individual Automated Guided Transport (IAGT)*. In *Metodi e tecnologie dell'ingegneria dei trasporti*, pages 84 – 85, S.N. – ITA, 13 - 15 dicembre 2005. s.n. Abstact.

[11] KRAUSS, STEFAN: *Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics*. PhD thesis, Deutsches Zentrum für Luft- und Raumfahrt, DLR, 1998. http://sumo.dlr.de/pdf/KraussDiss.pdf.

[12] LATOUR, BRUNO: *Aramis or the love of technology*. Harward University Press, Cambridge, Massachusetts, 1996.

[13] LEES-MILLER, J.D., J-S. HAMMERSLEY and R.E. WILSON: *Theoretical Maximum Capacity as a Benchmark for Empty Vehicle Redistribution in Personal Rapid Transit*. Journal of the Transportation Research Board 2010, 2146:76–83, 2003.

[14] LI, P., L. ALVAREZ and R. HOROWITZ: *AHS safe control laws for platoon leaders*. IEEE Trans. Contr. Syst. Technol., 5(6):615–628, November 1997.

[15] SCHWEIZER, J.: *Automated Guided Vehicle systems need a predictive, probabilistic network control*. Technical Report, Convegno annuale del Laboratorio di Analisi dei Sistemi di Trasporto, 12 2004.

[16] SCHWEIZER, J.: *Non-linear feedback control for short time headways based on constant-safety vehicle-spacing*. In *IEEE Intelligent Vehicles Symposium*, pages 167 – 172, S.N. – ITA, Giugno 2004. s.n. [RELAZIONE].

[17] SCHWEIZER, J.: *SUMOPy wiki*, 2016. http://www.sumo.dlr.de/wiki/Contributed/SUMOPy.

[18] Schweizer, J., A. Danesi, C. Lepori, M. Lupi and F. Rupi: *Sistemi innovativi per la mobilità urbana: il Personal Automated Transport a servizio misto*. T&T. trasporti e territorio, 3-4:165 – 172, 2008.

[19] Schweizer, J., A. Danesi, M. Lupi and F. Rupi: *Transport management and land-use effects in presence of unusual demand*, chapter Personal Rapid Transit : a self financing solution for tourist transport at the city of Rimini, pages 63–79. FrancoAngeli, Milano – Italia, 2011.

[20] Schweizer, J., A. Danesi, F. Rupi and E. Traversi: *Comparison of static vehicle flow assignment methods and microsimulations for a personal rapid transit network*. J. Adv. Transp., 46:340–350, 2012.

[21] Schweizer, J., L. Mantecchini and J. Greenwood: *Automated People Movers and Transit Systems 2011"From People Movers to Fully Automated Urban Mass Transit"*, chapter Analytical capacity limits of Personal Rapid Transit stations, pages 326–338. American Society of Civil Engineers (ASCE), RESTON (VIRGINIA) – USA, 2011.

[22] Schweizer, J. and J. Meggs: *3rd International Conference on Urban Public Transportation Systems 2013*, chapter PRT mode share estimations using a direct demand stated preference method, pages 1–9. Steven L. Jones, Restin, Virginia – USA, 2013.

[23] Schweizer, J., T. Parriani, E. Traversi and F. Rupi: *Optimum Vehicle Flows in a Fully Automated Vehicle Network*. In *Proceedings of the International Conference on Vehicle Technology and Intelligent Transport Systems: VEHITS*, volume 1. http://www.scitepress.org/DigitalLibrary/PublicationsDetail.aspx?ID=W0rOwi3OENM=.

[24] Spooner, J.T. and K.M. Passino: *Fault tolerant control for automated highway systems*. IEEE Trans. Veh. Technol., 46(3):771–785, August 1997.

[25] Szillat, M.: *PRT system simulation*. In *7th Int. Conf. on Automated People Movers'99*, pages 31–33, IDA House, Kalvabod Brygge 31-33,DK-1780 Copenhagen V, May 1999. IDA.

10 GatcomSUMO: A Graphical Tool for VANET Simulations Using SUMO and OMNeT++

P. Pablo Garrido Abenza, Manuel P. Malumbres and Pablo Piñol Peral
Dpto. Física y Arquitectura de Computadores; Miguel Hernández University, Spain
{pgarrido, mels, pablop}@umh.es

10.1 Abstract

A graphical application named GatcomSUMO is proposed to facilitate the realization of VANET simulations using the SUMO traffic simulator and the OMNeT++ network simulator. The main aim of this tool is to make easier the steps involved in the setup of VANET simulations. So, the tasks related to the generation of abstract scenarios, the use of network scenarios based on real maps, and the vehicles mobility modelling are integrated in a user-friendly graphical interface. This tool performs the necessary actions by invoking the set of utilities included in SUMO, saving the user from typing complex command-line orders. GatcomSUMO visualizes the network and routes easily, and they are created in such a way to meet the requirements of the involved simulators, in particular OMNeT++, in order to avoid misunderstandings and runtime errors. In addition, the application allows the conversion of the spatial coordinates used by SUMO into those used by OMNeT++, which is essential to place fixed elements like RSUs (Road-Side Unit) in the scenarios. GatcomSUMO is an open-source tool which is developed in Java, so, it can be run on the most commonly used platforms.

Keywords: VANET, SUMO, OMNeT++, Veins.

10.2 Introduction

A Mobile Ad hoc Network (MANET) is a kind of wireless network composed of a number of mobile nodes that can communicate directly with each other, without the use of any infrastructure. In turn, a Vehicular Ad hoc Network (VANET) is a particular kind of MANET in which the communication network is composed of mobile vehicles and fixed infrastructure, like Road-Side Units (RSU), so Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communications may take place. Whereas in a MANET the nodes can move freely in any direction, in a VANET the vehicles can only move on the roads network. VANETs can be applied to many fields, such as improving road safety or implementing traffic information systems. The use of simulation techniques is essential when working with wireless networks, because the implementation of a real testbed is usually an expensive and difficult task, especially in the case of VANETs. Furthermore, a VANET simulation is a more complex problem because two components are needed: a wireless network simulator to allow the wireless communication between the vehicles, and a traffic simulator for controlling their mobility. Several well known network simulators can be used by the scientific community, such as Riverbed Modeler (15), ns-2 (10), ns-3 (14), OMNeT++ (22), and JiST/SWANS (1). OMNeT++ is an open-source discrete event simulator that can be used in a great variety of domains; it has frameworks available for wired and wireless networks, queueing networks, satellite simulators, or cloud computing systems, to name a few. Frameworks like MiXiM (24), INET (21), or Castalia (12) are based on the OMNeT++ simulator and allows to simulate wireless networks.

On the other hand, there are a number of traffic simulators capable of generating mobility traces, which can be later processed by the network simulator. VanetMobiSim (4), VISSIM (8), and SUMO (6) are examples of well known traffic simulators. In real VANETs, the data received by one vehicle may determine its mobility behaviour. For example, when a vehicle receives a message with a warning about an accident located in its route, it can look for an alternative route to bypass the location of the accident. This behavior can be modeled by linking the traffic simulator with the network simulator in some way. SUMO (Simulation of Urban MObility) is a microscopic traffic simulator which models streets and roads, and supports many types of vehicles (multi-modal). SUMO offers an API (Application Programming Interface) named TraCI (Traffic Control Interface) (23) which allows an external application to control the simulation and to get information about the network and the simulated objects, that is, about the vehicles moving in the SUMO environment. In other words, SUMO allows to couple both traffic and network simulations through the TraCI interface. There exist several implementations of TraCI using different programming languages including C++, Java, Python or Matlab. Moreover, the above mentioned network simulators can also be linked to SUMO by means of some middleware like TraNS (13), iCS (5), Veins (19), or VSimRTI (17). TraNS (Traffic and Network Simulation) was intended for linking SUMO with ns-2, however, according to its website, its development has been suspended. On the other hand, iCS (iTETRIS Control System) connects SUMO with the ns-3 simulator. Veins (Vehicles in Network Simulation) is an OMNeT++ package that implements TraCI in order to connect SUMO with some network simulator framework like MiXiM (24) or INET (21). Finally, VSimRTI (V2X Simulation Runtime Infrastructure) is a more generic framework, as it allows to connect both VISSIM and SUMO traffic simulators with some different network simulators like ns-3, JiST/SWANS (Java in Simulation Time / Scalable Wireless Ad hoc Network Simulator), and OMNeT++.

For the purposes of our research we have selected OMNeT++ as the network simulator, SUMO as the traffic simulator and the Veins package to incorporate the VANET-specific protocols and to link both simulators. All of them are open-source software, which is a key factor to enhance reproducibility of research (20). In order to create a simulation, SUMO includes a set of tools (binaries and scripts) that receive files in XML (Extensible Markup Language) format, and generate output files with the same language. Although these tools and the XML files syntax are well documented, those files are difficult to edit by a human. Building a VANET simulation can be a time consuming and prone to errors task for any user, being a truly overwhelming process for a beginner. The aim of the GatcomSUMO tool is to automate every step needed in this task, although it can be used for other purposes too. Specifically, GatcomSUMO allows to generate abstract networks, to download and convert real maps from OpenStreetMap to be used with SUMO, and to generate routes for the mobile nodes. The necessary configuration files for the OMNeT++ simulator can also be generated. In addition, GatcomSUMO also incorporates a set of useful tools when working with wireless networks, such as the communication range and signal propagation loss calculators, and several unit converters (velocity, signal power, temperature) which provide the corresponding configuration parameters to be imported into the OMNeT++ simulator. GatcomSUMO reduces the learning curve for designing and implementing VANET simulations, and helps the researcher to really focus on research.

The remainder of the paper is organized as follows. Firstly, an overview of the SUMO workflow is presented in section 10.3, briefly describing the commands used for generating both the networks and the traffic demands. Section 10.4 shows the additional files needed for building VANET simulations with the OMNeT++ network simulator; if the reader has some experience with these simulators the previous sections can be skipped. In section 10.5, we enumerate some related works and summarize their main achievements. Then, in section 10.6, GatcomSUMO is described in detail. Finally, section 10.7 concludes the paper and proposes some future work.

10.3 SUMO Workflow

This section describes the usual steps needed in order to conduct a simulation that involves to the SUMO simulator. The usual workflow when working with SUMO is depicted in Fig. 10.1. As can be seen, SUMO works with two types of data: the network and the traffic demand, which are explained in the following sections.

10.3.1 Network

The network represents the roads or streets where the vehicles will move along. It is stored in ASCII text files with a plain-XML format (.net.xml). In brief, these files are compounds of a list of streets (*edges*), a list of lanes in each street (*lanes*), and a list of nodes that represents intersections between several streets (*junctions*). Optionally, they can include other information like connections or traffic lights. Fig. 10.2 shows a fragment of the 'net' section, which is the main part of a network file. Although the SUMO network files (.net.xml) can be edited manually, they are usually generated with some utility; if a network file needs to be modified manually, it is possible to use a visual network editor like the netedit tool included in SUMO. There are other command-line tools in SUMO for building either abstract networks (netgenerate), or import them from existing networks (netconvert). With netgenerate the user can generate from scratch three kind of abstract or synthetic networks with geometrical structure: grid, spider, and random networks (Fig. 10.3). GatcomSUMO allows to input the necessary arguments in a user friendly way in order to invoke to netgenerate with the desired command line. As a result, a network file (.net.xml) is generated. The following example generates a grid network named 'grid500.net.xml':

```
netgenerate --grid true --grid.number 5 --grid.length 500
            --output-file grid500.net.xml
```

An advantage of the grid networks is that the length of the streets are well known, which allows to validate the desired communication range in our simulations. Moreover, the names of nodes and edges are easy to remember, which is very useful when defining by hand our own routes for the vehicles (traffic demand). For example, an edge defined from node '0/0' to node '0/1' is assigned the identifier '0/0to0/1', and so on.

On the other hand, when working with VANETs scenarios, it is very common to use real maps. With the netconvert utility the user can import road networks in many formats, as OpenStreetMap (OSM) (3) and PTV VISUM/VISSIM (8). In this work only the OpenStreetMap format is considered, as it has enough data and quality for our experiments. All the maps downloaded from OpenStreetMap are based on the WGS84 (World Geodetic System 1984) coordinate system, which is the same ellipsoid used by GPS (Global Positioning System), and a coordinate system which is very close to the one used by GLONASS (Global Navigation Satellite System). Nowadays, both GPS and GLONASS are widely used to get the positioning information in vehicles and smartphones, so, it is feasible to use real tracks saved with a GPS receiver device on maps downloaded from OSM. All the maps that we have used in this work have been validated with tools like Osmose (OpenStreetMap Oversight Search Engine) (2), and, visually, by overlaying real GPS tracks (.gpx files) with them, by means of JOSM (Java OpenStreetMap) (16) or JGPSTrackEdit (9).

There are several ways for downloading maps from OSM. Firstly, by selecting a rectangular area from a web browser (http://www.openstreetmap.org/export). Secondly, by typing an URI (Uniform Resource Identifier) in a web browser indicating directly the coordinates in longitude/latitude of the south-west and north-east corners of the rectangular area (bounding box) to download. There are many utilities on the Internet to determine those coordinates, such as the one included in OpenStreetMap itself. For example, the following URI retrieves part of the city of Elche by using the OSM API, generating a file named 'map.osm.xml':

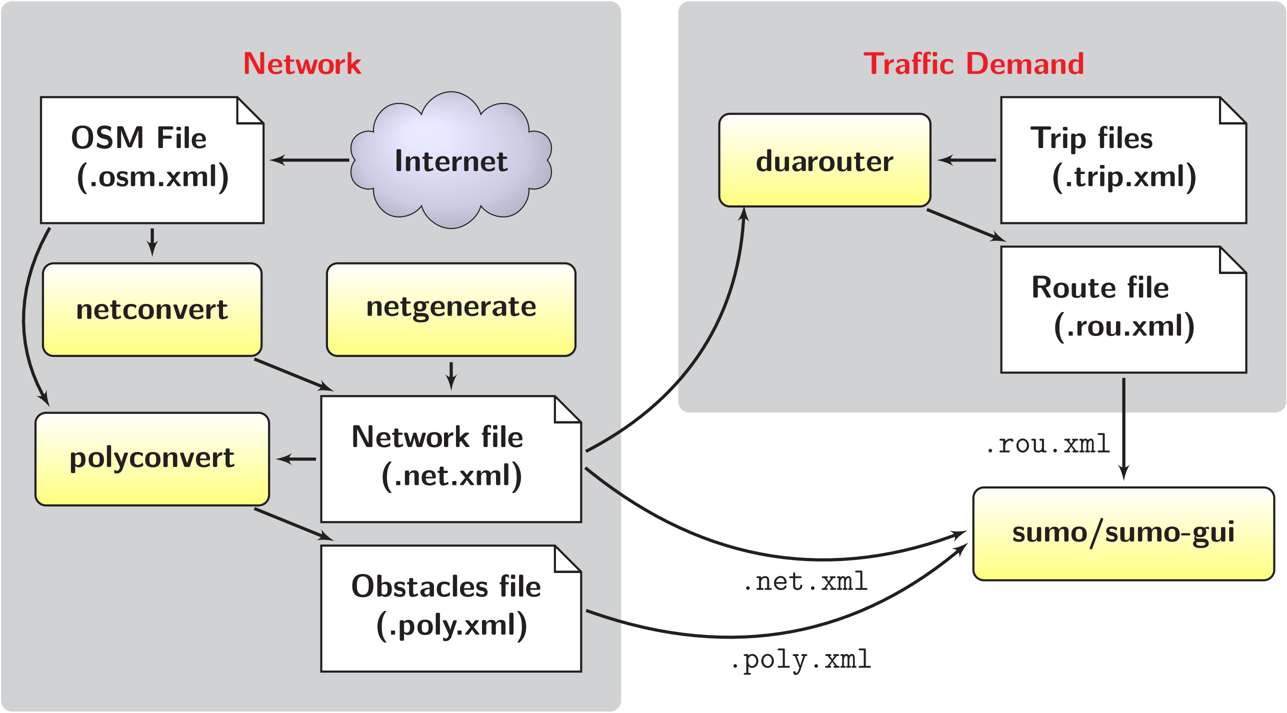


Figure 10.1: SUMO workflow

```
http://api.openstreetmap.org/api/0.6/map?bbox
   =-0.697,38.266,-0.677,38.283
```

The OSM API limits the maximum number of downloaded objects (e.g., the number of nodes is limited to 50000), and the maximum size of the bounding box (0.25 degree in either dimension). Nevertheless, it is possible to overcome these restrictions by using the OSM Extended API (XAPI), which allows much larger requests. As it will be explained later, GatcomSUMO builds URIs as shown above for retrieving the OSM maps in XML format, as well as several image formats (PNG, SVG, etc.).
A third option consists in the use of the `wget` command, which receives the previous URI as an argument and exports the map to the OSM file specified:

```
wget "<URI>" -O elche.osm.xml
```

The downloaded OSM file needs to be processed with the `netconvert` utility, which transforms geo-coordinates to metric coordinates of the network map in such a way that can be used by SUMO. The following is the minimal command line for this task:

```
netconvert --osm-files elche.osm.xml -o elche.net.xml
```

In addition, when working with realistic VANETs scenarios, it would be of interest to define the obstacles found in the scenario, like buildings, since they modify the radio coverage of vehicles circulating in their surroundings. At this point, the OSM files have the advantage of providing side information, apart from streets, lanes and junctions, to define extra polygons in the areas where vehicles cannot cross (buildings, parks, etc.). The obstacles can be generated automatically with the `polyconvert` utility, which takes as input a network file (.net.xml) and an OSM file (.osm.xml), and generates a new file (.poly.xml) with shapes that Veins identifies as obstacles. For example:

```
polyconvert --osm-files  elche.osm.xml  --net-file elche.net.xml
```

```
<net version="0.25" ...>
  <location netOffset="-312300.75,-5585447.93"
     convBoundary="10110.00,0.00, 15311.67,6350.00"
     origBoundary="30.355195,50.395771, 30.574502,50.527149"
     projParameter="+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84
        +units=m +no_defs" />
  <edge id="33632393" from="384747802" to="1502836202" priority="5"
     type="highway.service" shape="11371.46,5855.44 11349.99,5778.89
        ...">
     <lane id="33632393_0" index="0" speed="13.89" length="181.09"
        shape="11369.87,5855.88 11347.97,5777.79 11354.61,5775.75
           11329.48,5686.17" />
  </edge> ...
</net>
```

Figure 10.2: Network file (.net.xml) - 'net' section

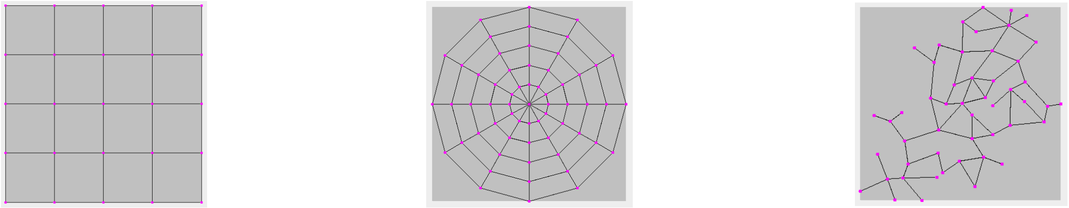


Figure 10.3: Abstract networks: Grid, spider and random

```
            --output-file elche.poly.xml
```

Although polyconvert accepts several input formats, it cannot process the abstract network files, so that kind of networks could not be used in a SUMO simulation with obstacles. However, GatcomSUMO has been extended with the functionality of generating the obstacles file only for grid networks, as it is very easy to compute the polygons as rectangles, without the use of the polyconvert utility.

10.3.2 Traffic Demand

As SUMO is a microscopic traffic simulator, each vehicle follows its own route, which is expressed as a list of edges. The mobility of all vehicles (traffic demand) is written into a routes file (.rou.xml). SUMO allows multi-modal simulations, that is, the use of distinct types of vehicles. The routes file (Fig. 10.4) starts with the definition of the vehicle types according with several physical parameters (e.g. length, maximum speed, acceleration, etc.), followed by the list of edge identifiers (ID) of every single route defined. Finally, the file contains the definition of individual vehicles or sets of vehicles (flows), including its type, the starting time, and the assigned route to follow.

There are several ways to generate these files, such as writing by hand explicit routes for each individual vehicle or flow, specifying the starting and the ending edge and searching the shortest path between both points with some tool (duarouter), using turning percentages at junctions (jtrrouter), using O/D (Origin/Destination) matrices (od2trips) with the PTV VISUM/VISSIM format, or generating them randomly with some scripts like randomTrips.py or dua-iterate.py.

Several things should be taken into account especially when writing a route file by hand. Firstly, the routes have to be connected, that is, each edge in a route should be followed by an adjacent edge. Secondly, a route needs at least two edges, even if a vehicle is going to be stopped (speed

```
<?xml version="1.0"?>
<routes>
  <!-- Vehicle types -->
  <vType id="rsu" length="1.0" maxSpeed="0.0"  accel="0.0" decel="0.0"
     />
  <vType id="car" length="5.0" maxSpeed="70.0" accel="2.6" decel="4.5"
         sigma="0.5"/>

  <!-- Routes -->
  <route id="route-rsu0" color="Orange"
         edges="0/4to0/3 0/3to1/3"/>
  <route id="route-car0" color="Blue"
         edges="1/3to2/3 2/3to3/3 3/3to3/2"/>

  <!-- RSU/Vehicles -->
  <vehicle id="rsu0" type="rsu" color="Yellow"
           depart="0.0" route="route-rsu0"/>
  <vehicle id="car0" type="car" color="Blue"
           depart="1.0" route="route-car0"/>
</routes>
```

Figure 10.4: Routes file (.rou.xml) - A fixed RSU and a car

```
<trips>
   <trip id="route-1" depart="0.0" color="Magenta"
    from="29350093#0" to="34691079#8" via="136456446#0" />
</trips>
```

Figure 10.5: Trip file (e.g. mytrip.xml)

equals to 0). Thirdly, the route assigned to each vehicle has been previously defined, and finally, the vehicles are sorted by the departure time, otherwise a warning message will be shown indicating that some vehicles will be ignored. Although there exists a script that checks some of the previous conditions (`routecheck.py`), there are other situations in which OMNeT++ runtime errors occurs, for example, when a vehicle leaves the playground area or moves to a negative coordinate. These conditions are difficult to check without the existence of some utility like GatcomSUMO.

The `duarouter` tool generates routes by specifying only the source and destination edges, and optionally, some intermediate edges too. This is known as a *trip*, and it is specified in an XML file as shown in Fig 10.5. `duarouter` receives the road network file (.net.xml) and the trip file and looks for the shortest path between the source and destination edges by means of a particular routing algorithm (`'dijkstra'` by default); as a result, a route file in XML format is generated:

```
duarouter --net-file elche.net.xml --trip-files mytrip.xml
          --output-file myroute.xml
```

SUMO includes some other tools and many more options, but in this section, only those that are supported by GatcomSUMO have been described.

10.4 OMNeT++ Workflow

As explained above, apart from the traffic simulator, the other necessary component in a VANET simulation is the network simulator. This section briefly explains how to use all the already generated SUMO files with the OMNeT++ network simulator, as well as other required files. The flow of data (⟶) and references between files (- - ->) are illustrated in Fig. 10.6, using the city of Elche as an example. The three files generated with SUMO (.net.xml, .rou.xml, and .poly.xml) need to be specified in a file 'elche.sumo.cfg', which is referenced from the following file 'elche.launch.xml'; this file references the same three previous files too. In turn, this file is referenced from the OMNeT++ configuration file (usually named omnetpp.ini). In addition, another XML file is needed with the analogue models parameters and the attenuation of the wireless signal depending on the different types of obstacles in the network (e.g. config.xml). This file should consider each type of obstacle defined in the obstacles file (.poly.xml), such as `building`, `parking`, `industrial`, `...`; on the contrary, they will not be shown in the OMNeT++ simulation window. The content of all of these files are omitted due to the space that would be needed. However, a fragment of them can be found in Fig. 10.18; as it will be shown later, GatcomSUMO generates all of them.

Some objects are referenced from the omnetpp.ini file, which need to be declared in an OMNeT++ scenario (scenario.ned); the design view of such scenario is shown in Fig. 10.7, but the source code is omitted again. Most of them belong to the Veins framework; the most relevant one is `manager`, which is an instance of the `TraCIScenarioManagerLaunchd` Veins class, and it is used by Veins for connecting with SUMO via the `sumo-launchd.py` script, and for creating the objects whose mobility will be controlled by SUMO.

10.5 Related Work

As can be seen, many software components and configuration files are needed in order to build a minimal VANET simulation, and each step requires some experience to do it properly. Several works have been published describing tools that provide additional functionality to the SUMO framework in order to to improve its usability.

eNetEditor (7) has a Graphical User Interface (GUI) that allows the creation of valid road networks for SUMO. The syntax of the network files (.net.xml) has been extended in order to support custom parameters about charging stations for electric vehicles, in a similar way than the definition of bus stops. However, the output network file is still compatible with SUMO. The main work of *eNetEditor* is done by building commands for invoking the SUMO tools. Specifically *eNetEditor* uses `netconvert`, `jtrrouter`, and `dfrouter` for generating routes. In addition, the routes can be calibrated with `duarouter`, in order to take into account traffic conditions, not only static conditions; depending on traffic, the shortest path may not be the fastest one.

TrafficModeler (11) is a tool developed in Java that allows to build the necessary configuration files for SUMO in a user-friendly way. It generates vehicle routes compatible with SUMO using several realistic traffic models. For example, it allows to generate activity-based traffic demand, that is, routes based on the daily activities of the population. Although the main purpose of *TrafficModeler* is the generation of vehicular traffic for SUMO, it also allows basic modifications to the road networks.

SUMOPy (18) is a tool written in Python which can be used from a GUI as well as from the command line interface with several provided scripts. The main aim of *SUMOPy* is to make SUMO more accessible to users with no programming experience. It can read and write road networks in the native SUMO format, download a real OpenStreetMap, convert it to a SUMO network file (.net.xml) and extract a polygons file (.poly.xml). Similarly to *TrafficModeler*, it allows to define activity-based traffic demand and additional vehicle types. Other traffic generation models can be used, such as the Origin/Destination (O/D) matrices, the junction turning definitions, or the random trips generators. For this task, *SUMOPy* uses some Python functions or the utilities provided by

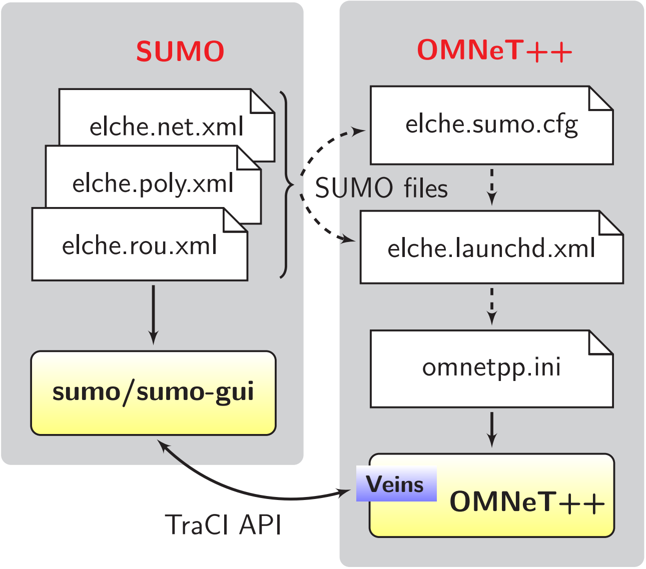


Figure 10.6: OMNeT++ workflow

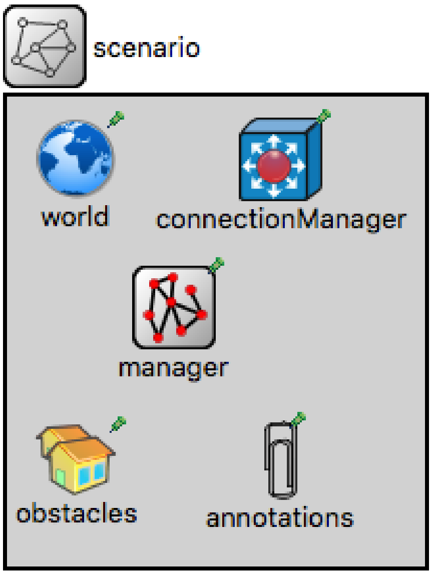


Figure 10.7: OMNeT++ scenario

SUMO, like `od2trips`, `duarouter`, and `jtrrouter`. This tool allows to examine the output files generated as a result of the simulation, which can be done interactively from the GUI.
Despite these tools overcome some handicaps, the GatcomSUMO application presented in this work fits better for the VANET field, especially when using the OMNeT++ network simulator.

10.6 GatcomSUMO

GatcomSUMO was born with the same basic idea as *TrafficModeler* and *SUMOPy*, that is, to simplify the use of the SUMO traffic simulator, specifically with network and traffic demand generation. Similarly to those tools, GatcomSUMO builds commands and delegates its work to the SUMO utilities (e.g. `netconvert`, `duarouter`, etc.). However, SUMO is not the only tool used when working with VANETs environments. The researcher needs a network simulator like OMNeT++ together with some frameworks extensions (e.g. Veins, INET, etc.) for the simulation of wireless networks.
The development of GatcomSUMO represents an effort to simplify each task involved with this complex software tool chain. The learning curve of SUMO is hard, and accomplishing a basic network simulation could take a long time and be frustrating. Even for an expert, doing a new experiment in which either the network or the routes of the vehicles change can be exasperating. GatcomSUMO is an application developed in Java that allows working in a user-friendly way; it does not contain any external script. The only requirements to execute it are to have SUMO and a JRE (Java Runtime Environment) installed, in addition to the executable or JAR (Java ARchive) file of the application. The graphical interface is structured in several main tabs according to the steps to follow during the setup of a simulation with SUMO and OMNeT++ (Fig. 10.8). Each main tab may contain several secondary tabs to cover all the provided functionality. Firstly, it is necessary to build or import the network, which contains nodes and edges ('Network' tab). Secondly, it is necessary to define the mobility of the vehicles ('Traffic demand' tab). Both the network and routes can be visualized in order to validate them and have a whole view of the network scenario ('Map' tab). Then, the set of files needed by OMNeT++ to run the simulation can be generated automatically ('OMNeT++' tab). Finally, the application includes several tools that provide different functionalities which may be useful for the researcher ('Tools' tab), like the conversion between geographical coordinate systems, some calculators for determining the transmission range and the values of corresponding configuration parameters, and the equivalences between different magnitude units (velocity, temperature, signal power, etc.). All the main tabs will be explained in detail in the following sections.

10.6.1 GatcomSUMO - Network

The 'Network' tab shows the options for either generating abstract networks, or downloading OSM real networks. Concerning to the abstract networks, the application can generate grid, spider and random networks (Fig. 10.3). After writing the values for the required parameters of the selected network, it will be created by pressing the [Create Network] button (Fig. 10.8), which invokes the `netgenerate` tool in the background (see the previous section for an example). If an offset is not used, the first street will be located at the origin `(0,0)`, and some of its lanes may be assigned to negative coordinates; this is not a problem for SUMO, but an error will be launched when running an OMNeT++ simulation, because vehicles can not move onto negative coordinates. So, it is recommended to add an offset value, and add it to the parameters `*.playgroundSizeX` and `*.playgroundSizeY` in the OMNeT++ configuration file (omnetpp.ini).

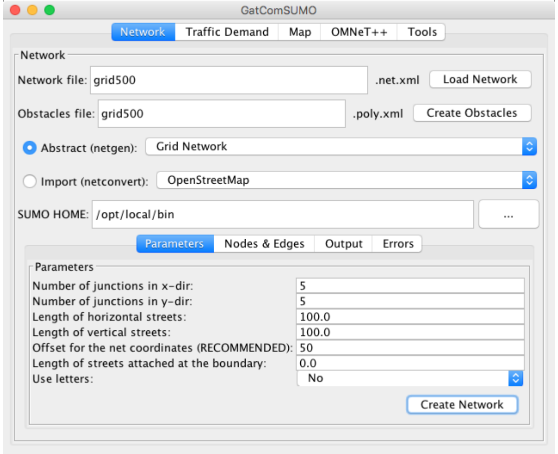
Another kind of networks supported by GatcomSUMO are those downloaded from OpenStreetMap. The application allows to download, import and generate the obstacles file in an intuitive way. At the present time it is necessary to know the real coordinates of the rectangle area to download, either geographic or UTM coordinates. Once those coordinates are determined, by clicking the [Download] button, GatcomSUMO builds the proper URI request as a text string taking into account the coordinates specified, creating an instance of the `URLConnection` Java class for retrieving the map. In the end, the geo-coordinates will be used, as shown in the previous section, but this is transparent for the user. Apart from downloading the map in the XML format (.osm.xml), the user can download the same area in a raster or vector graph format as PNG, JPEG, SVG, or PDF. In addition, the download dialog computes the playground size, which can be useful if the user needs to work with an area of an specific size. Once the OSM file (.osm.xml) has been downloaded, it is necessary to convert it to the SUMO network file format (.net.xml). GatcomSUMO allows to do this task by clicking the [Convert] button, which builds the command as a text string for invoking the `netconvert` tool and executes it. A set of predefined options is shown, and they can be changed before the converting process, if needed.

Once the network has been created, either abstract or OSM network, just clicking in [Create Obstacles] will generate the obstacles file (.poly.xml). The `polyconvert` utility cannot deal with abstract networks, but for the specific case of grid networks, GatcomSUMO computes them as rectangles a little smaller than the rectangles formed by the intersections of the streets (Fig. 10.14). In case of OSM networks, the `polyconvert` is used as shown above. From the user's point of view, there is no difference in the way in which the obstacles files are built for each type of network.

Finally, before defining the traffic demand, it is necessary to load the network file into the application. This is done by clicking the [Load Network] button. Once the network has been loaded, the user can visualize the map, identify a node or edge from the map by clicking over it, or locate an object in the map by selecting the edge or node from the list (Fig. 10.9); this will be necessary when specifying edges to build the routes. If the user needs the data created in a previous session, the network and the traffic demand can be loaded by clicking in the corresponding button.

10.6.2 GatcomSUMO - Traffic Demand

GatcomSUMO can easily generate robust routes and write them to a route file (.rou.xml). At first place, the user has to define vehicle types, although the application already incorporates several types, one for each vehicle class defined in SUMO by default, like `passenger`, `emergency`, `taxi`, or `bus`, to name a few. After that, a route can be defined by hand by selecting edges from a list of edges. When the user selects an edge from the list, only the adjacent edges to the last added one are shown, making the process faster and, at the same time, avoiding the possibility of building an unconnected route. The user can visualize any route in the map, with the associated color to validate it before running the simulation (Fig. 10.9). If some vehicles are already defined, the created routes



Network

Network file: grid500 .net.xml

Obstacles file: grid500 .poly.xml

● Abstract (netgen): Grid Network

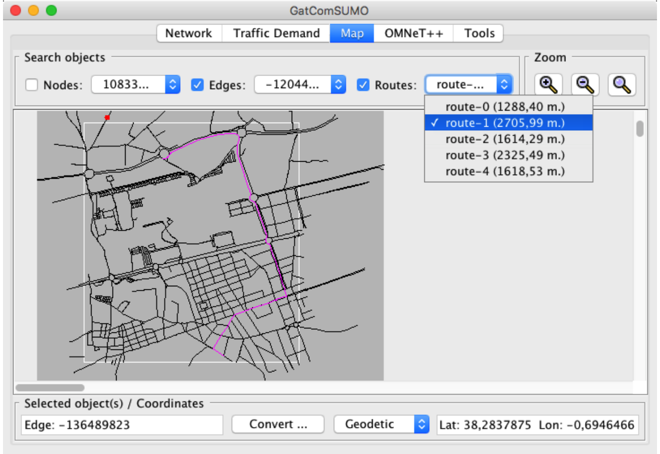
○ Import (netconvert): OpenStreetMap

SUMO HOME: /opt/local/bin

Parameters

Number of junctions in x-dir:	5
Number of junctions in y-dir:	5
Length of horizontal streets:	100.0
Length of vertical streets:	100.0
Offset for the net coordinates (RECOMMENDED):	50
Length of streets attached at the boundary:	0.0
Use letters:	No

Figure 10.8: Network tab



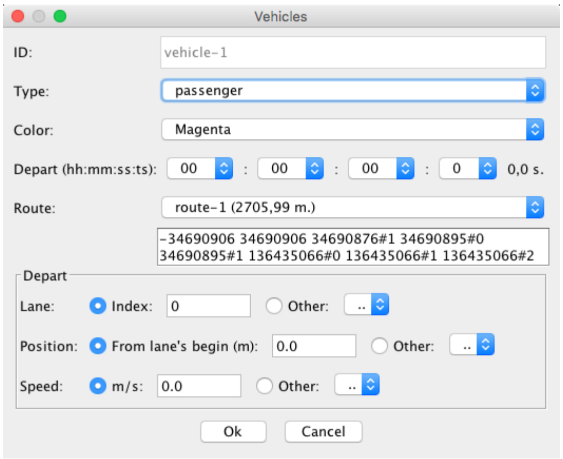
□ Nodes: 10833... ☑ Edges: -12044... ☑ Routes: route-...

route-0 (1288,40 m.)
✓ route-1 (2705,99 m.)
route-2 (1614,29 m.)
route-3 (2325,49 m.)
route-4 (1618,53 m.)

Selected object(s) / Coordinates

Edge: -136489823 Geodetic Lat: 38,2837875 Lon: -0,6946466

Figure 10.9: Map of city of Elche



ID: vehicle-1

Type: passenger

Color: Magenta

Depart (hh:mm:ss:ts): 00 : 00 : 00 : 0 0,0 s.

Route: route-1 (2705,99 m.)

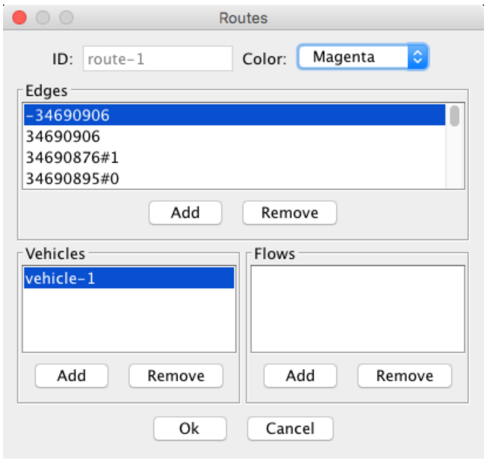
-34690906 34690906 34690876#1 34690895#0
34690895#1 136435066#0 136435066#1 136435066#2

Lane: ● Index: 0 ○ Other: ..

Position: ● From lane's begin (m): 0.0 ○ Other: ..

Speed: ● m/s: 0.0 ○ Other: ..

Figure 10.10: Vehicle dialog



ID: route-1 Color: Magenta

Edges:
-34690906
34690906
34690876#1
34690895#0

Vehicles:
vehicle-1

Figure 10.11: Route dialog

can be assigned to one or more of them, either by selecting its route from the vehicle definition (Fig. 10.10), or by adding a vehicle to the current route in the route dialog (Fig. 10.11).

The above procedure is only feasible when routes are known and there are few routes to create. Another way to create new routes is the use of *trips*; a *trip* is a route computed knowing only the origin and destination edges, which may be selected from the list of available edges in the map or randomly. Optionally, the user can add one or more intermediate edges (Fig. 10.12), in order to force the route to cross a particular edge in the path to the destination. Once the required data has been defined, the program generates an intermediate file for the trip (.trip.xml) which is used as input to the SUMO `duarouter` tool, together with the network file (.net.xml), in order to look for a path between source and destination with some routing algorithm; currently, GatcomSUMO uses the shortest-path Dijkstra's algorithm only (see the previous section for an example). The output file of `duarouter` is a route file (.rou.xml) containing the list of edges of the route found, creating a route object in memory. In case of building a greater number of routes, GatcomSUMO allows to specify a number of iterations, and the previous process is repeated as many times as the user requests. This works similarly to the `randomTrips.py` script, but with the following improvements. Firstly, the program detects, at each iteration, if a valid route between a given source and destination cannot be found; for example, when an isolated edge is selected as the origin or destination of the route. In that case, the application ignores the current iteration and retries again with a different source or destination (in case of being randomly choosen). Secondly, the program allows to apply some filters before or after the route creation process: (1) the length of the generated routes should be less or greater than the specified value, (2) to filter routes that have some edge outside the bounding box, because OMNeT++ generates a runtime error when a vehicle leaves the simulation area, and (3) it is also possible to specify the vehicle type permitted by the edges within each route; otherwise, a runtime error occurs when a vehicle tries to move through an edge that does not allow that kind of vehicle. At the end of the process, the user can preview the generated temporary routes, their distance, and the list of edges that compose them. When the user selects one or more routes, the minimum playground size to enclose all the selected routes is computed and shown at bottom. Once the desired routes are selected, they can be finally added to the routes table by clicking the [Add route(s)] button (Fig. 10.13). In order to visually check the coherence of each route, the user can select them from the 'Map' tab. If the bounding box filter is selected, no route should leave that area, which is shown as a white rectangle on the map (Fig. 10.9); this helps in keeping the average vehicle density constant in a specific area. An additional utility allows to compute distances between two points, which is useful for validating the length of any street, the bounding box of the network, or the communication range in a wireless network. GatcomSUMO offers the possibility of creating either vehicles or vehicle flows automatically and assigning routes to them. The identifiers, type of vehicle, and color can be specified, as well as the initial departure time of a vehicle, and the increment value for each one.

Finally, all the data described in this section will be saved into a single route file (.rou.xml), that can be previewed in sumo-gui or used to run an OMNeT++ simulation. It is worthy noticeable that the vehicles and vehicle flows will be sorted by their departure time, avoiding the above mentioned warning. The process of writing a route file has been hugely simplified with GatcomSUMO, as the user does not need to write routes by hand, neither to invoke `duarouter` repeatedly.

10.6.3 Placing RSUs

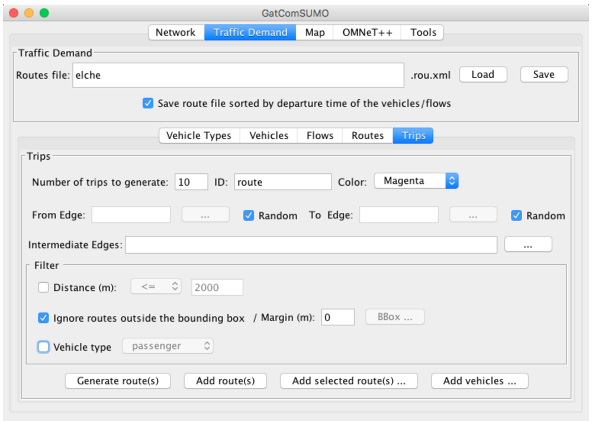
In VANETs scenarios is very common to work with both mobile vehicles and fixed antennas or RSUs (Road-Side Unit). When it is necessary to set the position of a fixed object there are two choices: (a) by defining a 'mobile' car without mobility in SUMO, or (b) by specifying its position in the OMNeT++ configuration file (omnetpp.ini).

The first case consists in defining a car based on a vehicle type defined in the SUMO route file (.rou.xml) whose maximum speed is set to 0; in addition, as it has been said before, a route needs two edges at least (see Fig. 10.4). However, this option has some disadvantages, such as setting the exact RSU position; the object will be located at the beginning of the first edge specified in the assigned route, and oriented to the second one. Furthermore, the vehicle should be kept 'hidden' because it will be considered a vehicle just like all the rest, so, its presence could affect to other mobile vehicles and could cause a traffic jam (Fig. 10.14a).

A better alternative could be to specify the exact coordinates for the RSU in the OMNeT++ configuration file (omnetpp.ini). The following example sets the position `(1500,500)` for the OMNeT++ object `rsu[1]`:

```
*.rsu[1].mobility.x = 1500
*.rsu[1].mobility.y =  500
```

However, a problem appears at this point since the coordinate systems used in SUMO and OMNeT++ are different. Although both of them use Cartesian coordinates, SUMO considers the origin `(0,0)` in the bottom-left corner and the Y-axis grows upwards, whereas in OMNeT++ the origin is located at the top-left corner, and the Y-axis grows downwards (Fig. 10.15). In addition, SUMO applies an offset to all the coordinates of the network files (.net.xml). As said before, the maps downloaded from OSM use geodetic coordinates (i.e. Longitude/Latitude), which are converted into metric coordinates by `netconvert` SUMO utility by applying a projection to an specific UTM zone in order to minimize the distortion. Next, the UTM coordinates are translated into the origin `(0,0)`, that is, an offset is applied. In this way, the values in the whole file are much lower than the real ones, and


GatComSUMO
Network | Traffic Demand | Map | OMNeT++ | Tools
Load
Save
Vehicle Types | Vehicles | Flows | Routes | Trips
...
BBox ...
Generate route(s)
Add route(s)
Add selected route(s) ...
Add vehicles ...

Traffic Demand

Routes file: elche .rou.xml

☑ Save route file sorted by departure time of the vehicles/flows

Trips

Number of trips to generate: 10 ID: route Color: Magenta

From Edge: ☑ Random To Edge: ☑ Random

Intermediate Edges:

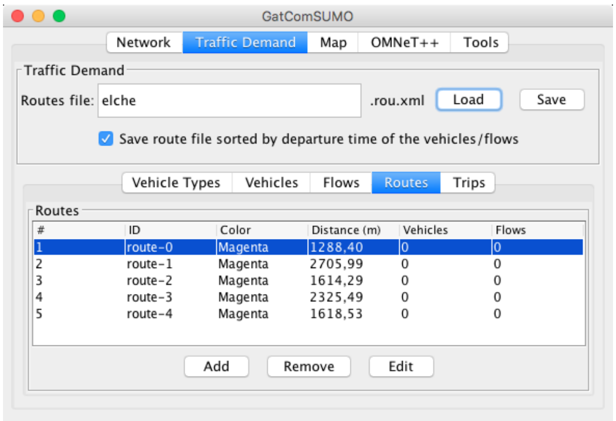
Filter

☐ Distance (m): <= 2000

☑ Ignore routes outside the bounding box / Margin (m): 0

☐ Vehicle type passenger

Figure 10.12: Trips definition


GatComSUMO
Network | Traffic Demand | Map | OMNeT++ | Tools
Load
Save
Vehicle Types | Vehicles | Flows | Routes | Trips
Add
Remove
Edit

Traffic Demand

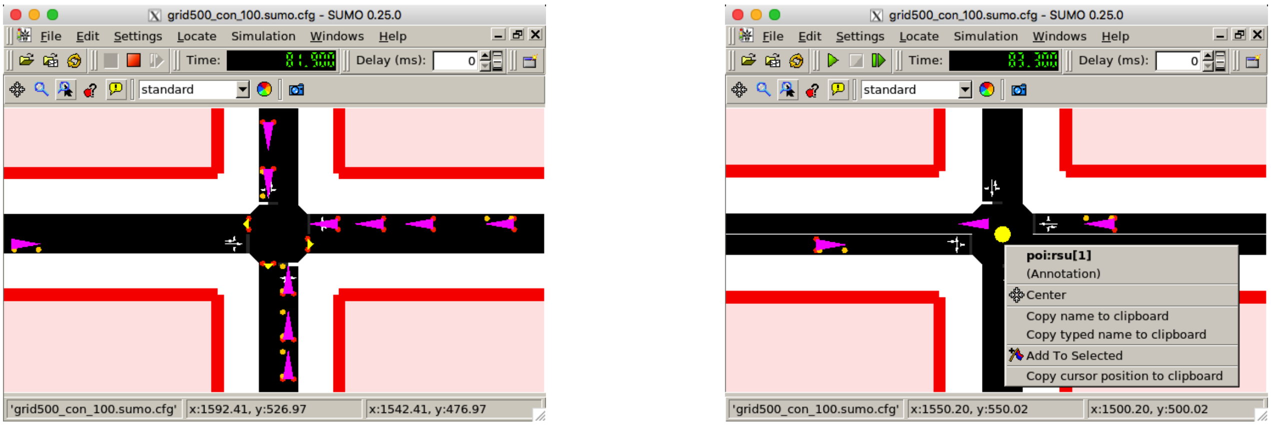
Routes file: elche .rou.xml

☑ Save route file sorted by departure time of the vehicles/flows

Routes

#	ID	Color	Distance (m)	Vehicles	Flows
1	route-0	Magenta	1288,40	0	0
2	route-1	Magenta	2705,99	0	0
3	route-2	Magenta	1614,29	0	0
4	route-3	Magenta	2325,49	0	0
5	route-4	Magenta	1618,53	0	0

Figure 10.13: Table of defined routes


grid500_con_100.sumo.cfg - SUMO 0.25.0
File Edit Settings Locate Simulation Windows Help
Time:
81.900
Delay (ms):
0
standard
'grid500_con_100.sumo.cfg'
x:1592.41, y:526.97
x:1542.41, y:476.97
grid500_con_100.sumo.cfg - SUMO 0.25.0
File Edit Settings Locate Simulation Windows Help
Time:
83.300
Delay (ms):
0
standard
poi:rsu[1]
(Annotation)
Center
Copy name to clipboard
Copy typed name to clipboard
Add To Selected
Copy cursor position to clipboard
'grid500_con_100.sumo.cfg'
x:1550.20, y:550.02
x:1500.20, y:500.02

(a) Three RSUs stopped causing a traffic jam (b) An RSU placed in OMNeT++

Figure 10.14: Placing RSUs (yellow) and cars (magenta)

these are the values stored in the SUMO network file (.net.xml). At the beginning of the network file, below the `net` root element (Fig. 10.2), there is a child element named `location`. This element contains the `projParameter` parameter, which shows the data of the applied projection: UTM zone and ellipsoid, usually WGS84 (the process of converting geo-coordinates into UTM coordinates is outside of the scope of this work). On the other hand, the value of the `netOffset` parameter contains the offset applied to each UTM pair `(x,y)` in order to do the translation into the final SUMO coordinates.

In the context of this work, the most relevant conversions are those between SUMO and the OMNeT++ network simulator. As it has been said before, this is necessary for placing fixed objects in the network like RSUs by specifying coordinates in the OMNeT++ configuration file (omnetpp.ini). As such coordinates cannot be determined from the SUMO graphical view (sumo-gui), it is necessary to convert from one coordinate system to the other. There exist two functions in the Veins framework named `traci2omnet()` and `omnet2traci()` within the `TraCIConnection` class, which do the conversions from SUMO to OMNeT++ and vice versa, respectively. However, both functions are declared as `private` and they cannot be invoked by a user module outside that class. Moreover, the problem is that the user needs to know which coordinates to write in the omnetpp.ini file before running any OMNeT++ code.

GatcomSUMO is able to do all the necessary conversions as explained next. The `convBoundary` parameter of the `location` element is necessary for this task. This parameter is the bounding box or rectangle that encloses the network in SUMO coordinates: `xmin`, `ymin`, `xmax`, and `ymax`. In addition, it is necessary to take into account that the direction of the Y-axis is the opposite, for which it is necessary to compute previously the height of the network. In order to convert SUMO coordinates (xs,ys) to OMNeT++ (xo,yo) and vice versa, the implemented functions are like the following ones (in pseudo-code):

```
function sumo2omnet(xs,ys) {
   height = convBoundary.ymax - convBoundary.ymin
   xo = xs - convBoundary.xmin
   yo = height - (ys - convBoundary.ymin)
}

function omnet2sumo(xo,yo) {
   height = convBoundary.ymax - convBoundary.ymin
   xs = xo + convBoundary.xmin
   ys = height - (yo + convBoundary.ymin)
}
```

In addition, the Veins class TraCIScenarioManager uses a margin parameter, which is a value added to the vehicle positions received from SUMO (25 by default), and it should be also considered in the previous functions. Currently, that value is not considered in GatcomSUMO, but this is solved by adding it to the coordinates of each fixed object in the omnetpp.ini file. For example, the following fragment specifies a margin of 25 and defines an iteration variable for adding that value to the position of a fixed RSU named rsu[1]:

```
*.manager.margin = ${margin=25}
*.rsu[1].mobility.x = 1500 + ${margin}
*.rsu[1].mobility.y =  500 + ${margin}
```

Sometimes it would be interesting to convert between real geo-coordinates or UTM, and OMNeT++ or SUMO coordinates. For example, the user could be interested in placing an RSU knowing its Longitude/Latitude or UTM coordinates, or in a selected point in the sumo-gui window; after doing the proper conversion to OMNeT++ coordinates, the user could specify them in the omnetpp.ini file. Although the conversions described before are not complicated, doing them manually is a tedious task and prone to errors. The application allows to convert coordinates in two different ways: (1) by clicking with the mouse on the map (Fig. 10.9), and (2) by typing the exact values to convert by means of a converter (Fig. 10.16). GatcomSUMO works with the SUMO coordinate system, and

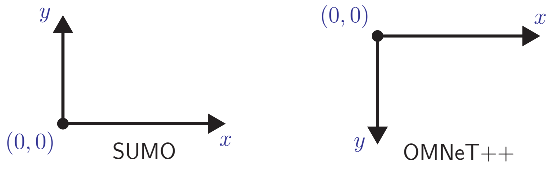
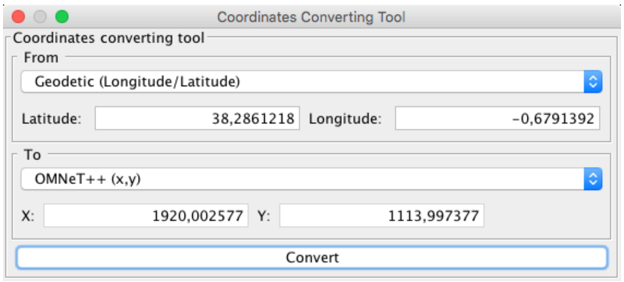


Figure 10.15: Coordinate systems



Coordinates converting tool

From

Geodetic (Longitude/Latitude)

Latitude: 38,2861218 Longitude: -0,6791392

To

OMNeT++ (x,y)

X: 1920,002577 Y: 1113,997377

Figure 10.16: Coordinate systems conversion

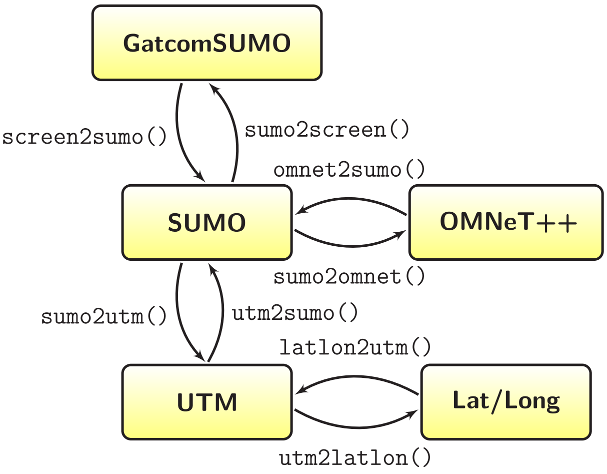


Figure 10.17: Conversion functions

includes functions to do conversions between all the above mentioned coordinate systems, directly or indirectly (Fig. 10.17). The names of the functions are self-explanatory.
An advantage of placing the fixed objects in OMNeT++ is the possibility of using sequences of values and parallel iteration within the omnetpp.ini file. With this OMNeT++ features it is possible to repeat some experiment varying the position of the fixed objects. In the next example, three positions for the `rsu[0]` object are defined: `(500,500)`, `(600,600)` and `(700,700)`, being `x` and `y` two iteration variables that change in parallel:

```
*.rsu[0].mobility.x = ${x=500..700 step 100}
*.rsu[0].mobility.y = ${y=500,600,700 ! x}
```

An additional advantage has to do with the type of the OMNeT++ module that Veins uses to instantiate the vehicle objects in SUMO, as well as their name. Both parameters are specified by setting the parameters `moduleType` and `moduleName` of the `TraCIScenarioManager` object defined in the scenario, respectively. For example:

```
*.manager.moduleType = "es.umh.nodes.car"
*.manager.moduleName = "car"
```

If the specified module code will be in charge of modelling the behaviour of both mobile (i.e. cars) and non-mobile (i.e. RSUs) objects, this could be large and confusing. Moreover, with the previous declaration, all the objects will be named `car[0]`, `car[1]`, `...`, regardless of the type of object. In order to use different module codes and names for each case, the alternative is to use fixed coordinates in the omnetpp.ini file for the fixed objects, as explained before. In this way, the fixed objects could be named as `rsu[0]`, `rsu[1]`, `...`, whereas the mobile vehicles can be defined as usual and they will be controlled by the road traffic simulation.
To sum up, the possibility of setting the position for the fixed objects in the omnetpp.ini instead of setting it in SUMO has the following three advantages:

- To be able to place the non-mobile objects in the exact position in the scenario, and without interfering with the rest of the traffic.
- To be able to define a sequence of values for the position of fixed objects in order to repeat some experiment varying their position.
- To be able to use a module code for fixed objects different from the one used for mobile nodes whose mobility is controlled by SUMO, improving the readability.

As a minor drawback, the objects placed as explained in this section will not be shown in sumo-gui. This could be a problem if the user desires to validate their position or take a screenshot. However, this can be easily solved by using POIs (Point of Interest) for each fixed object. For example, the following snippet creates a small yellow circle as a POI in sumo-gui when that window is open and the object receives the first self-message:

```
/*************  RSUApp.cc  *************/

using namespace Veins;

cModule*                 host;
BaseMobility*            mobility;
TraCIScenarioManager*    traciMng;
TraCICommandInterface*   traciCmd;
TraCIColor               color_poi;
```

```
bool                    firstTime;

void RSUApp::initialize(int stage) {
  host = findHost();
  mobility = dynamic_cast<BaseMobility*>(host->getSubmodule("mobility"
     ));
  traciMng = TraCIScenarioManagerAccess().get();
  traciCmd = traciMng->getCommandInterface();
  color_poi = TraCIColor::fromTkColor("yellow");
  firstTime = true;
}

void RSUApp::handleSelfMsg(cMessage *msg) {
  if (firstTime && ev.isGUI()) {
    firstTime = false;
    Coord pos = mobility->getCurrentPosition();
    traciCmd->addPoi(host->getFullName(),"Annotation",color_poi,6,pos)
       ;
  }
}
```

Once the POI has been created, it will be shown in the sumo-gui window (Fig. 10.14b), and it could be searched by its name as any other POI by using the option POI within the Locate menu.

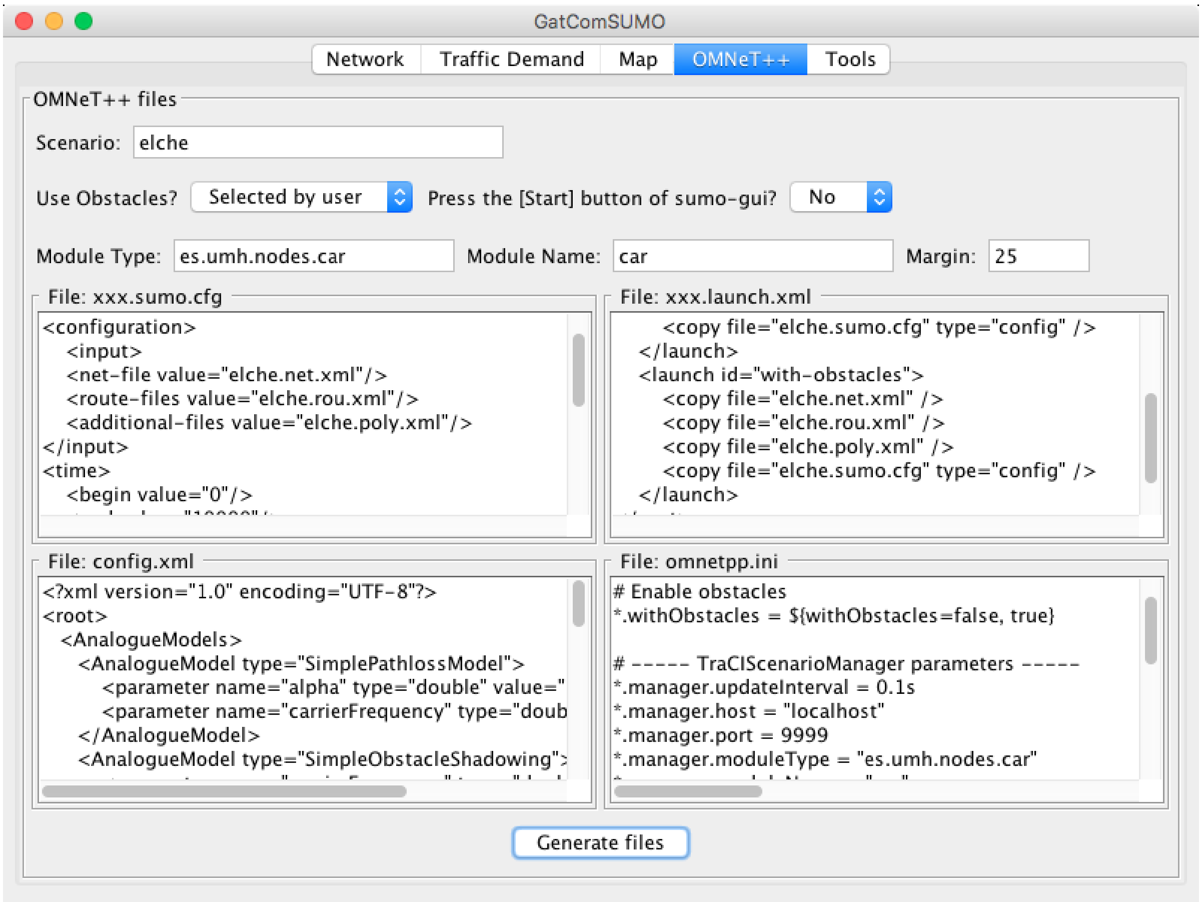
10.6.4 GatcomSUMO - Simulation Files

As explained in section 10.4, a VANET simulation requires the files used by the SUMO traffic simulator (.net.xml, .poly.xml, .rou.xml), and a set of additional files needed by the OMNeT++ network simulator. With the 'OMNeT++' tab, the user can generate automatically all such files, instead of copying them from a previous project and changing the name and some other parameters. In the example shown in Fig. 10.18, by clicking the [Generate files] button, the following files will be generated and shown, according with the selected options: elche.sumo.cfg, elche.launchd.xml, config.xml, and a fragment of the omnetpp.ini file (only the part related with the `TraCIScenarioManager` object and a recommended minimum size for the playground are included). GatomSUMO takes into account the different kinds of polygons defined in the network (.poly.xml file) and appends an obstacle type for each one in the config.xml file. Some other files are also generated, like a shell script `run.sh`, useful for running the simulations sequentially or in parallel if a multi-core or a multi-processor system is available.

10.6.5 GatcomSUMO - Tools

The application includes the 'Tools' main tab, which contains several useful converters and calculators (Fig. 10.19). When working with wireless networks in the OMNeT++ network simulator, it is necessary to specify some values in some specific units, and the user has to take his/her pocket calculator, to use a spreadsheet, or to look for some utility in the Internet. The aim of this tab is to have all the necessary utilities in a centralized place.

The main tool deals with the Free-Space Path Loss (FSPL), which needs to be know when dealing with radio communications, for example, for computing the communication range. FSPL is the loss in the signal strength that occurs when a wave is sent through a medium without obstacles. The



OMNeT++ files

Scenario: elche

Use Obstacles? Selected by user Press the [Start] button of sumo-gui? No

Module Type: es.umh.nodes.car Module Name: car Margin: 25

File: xxx.sumo.cfg

```
<configuration>
  <input>
  <net-file value="elche.net.xml"/>
  <route-files value="elche.rou.xml"/>
  <additional-files value="elche.poly.xml"/>
</input>
<time>
  <begin value="0"/>
```

File: xxx.launch.xml

```
    <copy file="elche.sumo.cfg" type="config" />
  </launch>
  <launch id="with-obstacles">
    <copy file="elche.net.xml" />
    <copy file="elche.rou.xml" />
    <copy file="elche.poly.xml" />
    <copy file="elche.sumo.cfg" type="config" />
  </launch>
```

File: config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <AnalogueModels>
    <AnalogueModel type="SimplePathlossModel">
      <parameter name="alpha" type="double" value="
      <parameter name="carrierFrequency" type="doub
    </AnalogueModel>
    <AnalogueModel type="SimpleObstacleShadowing">
```

File: omnetpp.ini

```
# Enable obstacles
*.withObstacles = ${withObstacles=false, true}

# ----- TraCIScenarioManager parameters -----
*.manager.updateInterval = 0.1s
*.manager.host = "localhost"
*.manager.port = 9999
*.manager.moduleType = "es.umh.nodes.car"
```

Figure 10.18: OMNeT++ simulation files

following formula computes the FSPL in mW:

$$FSPL(mW) = \left(\frac{4 \cdot \pi \cdot d \cdot f}{c}\right)^2 = \left(\frac{4 \cdot \pi \cdot d}{\lambda}\right)^2 \tag{10.1}$$

where:

$FSPL$ = Free-Space Path Loss (mW)
d = distance of the receiver from the transmitter (metres)
f = signal frequency (Hertz)
λ = signal wavelength ($\lambda = c/f$ meters)
c = speed of light in a vacuum ($\sim 3 \cdot 10^8$ m/s)

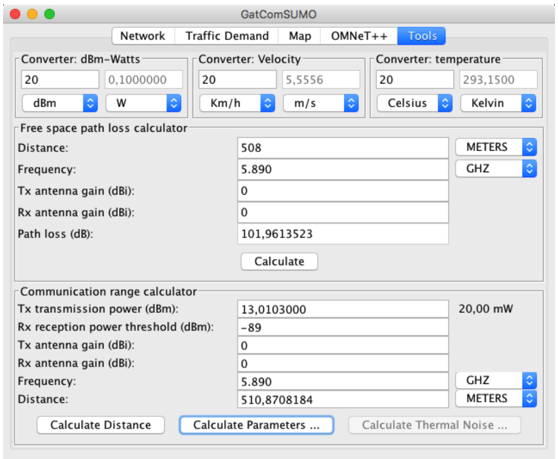
Since the majority of measurements in this field are made in decibels (dB), it could be necessary to convert between mW and dBm; these conversions use the following formulas:

$$P_{dBm} = 10 \cdot log_{10}(P_{mW}) \tag{10.2}$$

$$P_{mW} = 10^{(P_{dBm}/10)} \tag{10.3}$$

By applying (10.2) to (10.1) we can obtain another formula that calculates the FSPL in terms of dB:

$$FSPL(dB) = 20 \cdot log_{10}(d) + 20 \cdot log_{10}(f) - 147.56 \tag{10.4}$$



Converter: dBm–Watts: 20 | 0,1000000 | dBm | W

Converter: Velocity: 20 | 5,5556 | Km/h | m/s

Converter: temperature: 20 | 293,1500 | Celsius | Kelvin

Free space path loss calculator

Distance: 508 METERS

Frequency: 5.890 GHZ

Tx antenna gain (dBi): 0

Rx antenna gain (dBi): 0

Path loss (dB): 101,9613523

Communication range calculator

Tx transmission power (dBm): 13,0103000 20,00 mW

Rx reception power threshold (dBm): -89

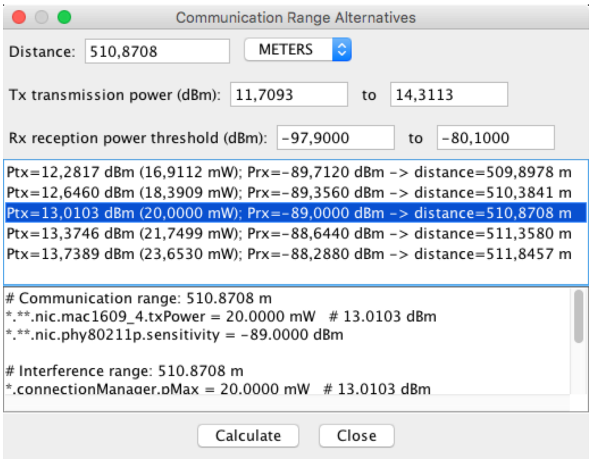
Tx antenna gain (dBi): 0

Rx antenna gain (dBi): 0

Frequency: 5.890 GHZ

Distance: 510,8708184 METERS

Figure 10.19: Tools



Distance: 510,8708 METERS

Tx transmission power (dBm): 11,7093 to 14,3113

Rx reception power threshold (dBm): -97,9000 to -80,1000

```
Ptx=12,2817 dBm (16,9112 mW); Prx=-89,7120 dBm -> distance=509,8978 m
Ptx=12,6460 dBm (18,3909 mW); Prx=-89,3560 dBm -> distance=510,3841 m
Ptx=13,0103 dBm (20,0000 mW); Prx=-89,0000 dBm -> distance=510,8708 m
Ptx=13,3746 dBm (21,7499 mW); Prx=-88,6440 dBm -> distance=511,3580 m
Ptx=13,7389 dBm (23,6530 mW); Prx=-88,2880 dBm -> distance=511,8457 m
```

```
# Communication range: 510.8708 m
*.**.nic.mac1609_4.txPower = 20.0000 mW   # 13.0103 dBm
*.**.nic.phy80211p.sensitivity = -89.0000 dBm

# Interference range: 510.8708 m
*.connectionManager.pMax = 20.0000 mW   # 13.0103 dBm
```

Figure 10.20: Communication range tool

where:

$FSPL$ = Free-Space Path Loss (dB)
d = distance of the receiver from the transmitter (metres)
f = signal frequency (Hertz)

The final constant value depends on the units used for *d* and *f*; in this case the value -147.56 is needed for meters and Hz. If the distance or the frequency are expressed in other units, the constant value will be different. For example, for meters and GHz, such value is 32.44. GatcomSUMO allows to use any units for *d* (meters, kilometers) and *f* (Hz, KHz, MHz, GHz) by simply selecting them from the dropdown list, and the program computes the proper value accordingly.
In order to calculate the communication range it is necessary to know the power of reception. When a signal reaches the recipient with an intensity above a certain threshold, then it will be able to interpret the packets sent; otherwise, such packages are considered as noise and will be discarded. Signals can also can cause interference and errors in packets if they collide.
The Friis equation for computing the power of reception in linear units (not dB) is given by:

$$P_{rx} = P_{tx} \cdot G_{tx} \cdot G_{rx} \cdot \left(\frac{\lambda}{4 \cdot \pi \cdot d} \right)^2 \tag{10.5}$$

where:

P_{rx} = received power or sensitivity (W)
P_{tx} = transmitter output power (W)
G_{tx} = antenna gain of the transmitter (absolute)
G_{rx} = antenna gain of the receiver (absolute)
λ = signal wavelength ($\lambda = c/f$ meters)
c = speed of light in a vacuum ($\sim 3 \cdot 10^8$ m/s)
f = signal frequency (Hertz)

By solving the previous equation (10.5) for the distance (d), we get the equation for computing the communication range, as used in the `calcInterfDist()` function of the Veins `ConnectionManager` class:

$$d = \left(\frac{P_{tx} \cdot G_{tx} \cdot G_{rx} \cdot \lambda^2}{(4 \cdot \pi)^2 \cdot P_{rx}} \right)^{1/2} \tag{10.6}$$

As an example, the default parameter values in Veins are the following: P$_{tx}$ = 13.0103 dBm ($\sim$ 0.02 W = 20 mW), P$_{rx}$ = -89 dBm ($\sim$ 1.2589254E-12 W), G$_{tx}$ = G$_{rx}$ = 0 dBi ($\rightarrow 1$), and f = 5.890 GHz. With these values, the achieved communication range is: $d = 510.8708184\ m$ ($\sim 0.51\ Km$).

More interesting than the previous calculations is to answer the question of which values are needed for such parameters P_{tx} and P_{rx} in order to achieve a desired communication range. As there is not a single solution, it is necessary to vary each parameter with values ranging between a lower and an upper boundary. The distance is computed for each combination of values as shown before, and if the obtained distance is very close to the desired communication range, that case is added to the list of possible solutions. This is what GatcomSUMO does by just clicking on the [Calculate Parameters] button, which opens the dialog shown in Fig. 10.20. The user only has to select one of the results of the list, and the corresponding Veins parameters will appear with the calculated values in the bottom part, which can be directly copied and pasted to the OMNeT++ configuration file (omnetpp.ini). As can be seen, some parameters are expressed in mW and others in dBm; this is why the converter between W, mW, and dBm was added using the (10.2) and (10.3) equations. There is another converter for velocity units (km/h, m/s), useful for defining vehicle types. Finally, there is a converter between temperature units like Celsius degrees (°C), Fahrenheit degrees (°F), and Kelvin (K), which can be useful for computing the thermal noise, as the parameters need to be expressed in Kelvin.

10.7 Conclusions and Future Works

In this work, an application named GatcomSUMO has been presented, which allows to build the necessary simulation files when planning a simulation of VANETs using SUMO (network and traffic demand), OMNeT++, and Veins. The application generates all the necessary XML files in a user-friendly way in a matter of minutes, and without previous advanced knowledge about SUMO, OMNeT++ or OpenStreetMap. The application acts as a front-end, builds the proper commands according to the selected options in the graphical interface, and invokes the SUMO tools as a back-end. As a result, network files (.net.xml) and routes files (.rou.xml) are generated, among others. This ensures that the syntax and semantic of the files are correct, and error-free when running the simulations with the OMNeT++ network simulator and SUMO. The application can be used as an educational tool too, as the executed commands are shown, as well as the output and error messages. GatcomSUMO makes reproducibility of VANET simulations easier, because networks can be generated or downloaded from a brief description (e.g. coordinates of the bounding box), and the generated routes can be used as mobility traces for other experiments.

Future work includes to improve the map visualization, specifically to draw the different edges of a street, and lanes of each edge. As edges are unidirectional, it is very common that every street is defined with two edges, one for each direction, but currently, both are drawn one over the other. This is a problem when selecting the initial or final edge for a route by clicking on the map. In addition, an edge is drawn only as a poly-line, regardless the number of lanes contained in the streets. Regarding to the traffic demand, more generation models could be supported; currently, only defining routes by hand and randomly generated are supported. This work could be done by coding it into the application, or by invoking some SUMO utilities like `marouter`, `od2trips`, or `jtrrouter`. It might also be interesting to be able to randomly select the type of vehicles according to some probability value when routes and trips are generated; for the moment, it is possible to choose the vehicle type manually, and that type will be used for each vehicle. Nevertheless, the process can be repeated as many times as you want, with a different type each time.

Some of the tools included have been specially developed to be used with Veins, like the parameters obtained with the communication range utility, which can be simply copied and pasted from the application to the omnetpp.ini file. This utility could be generalized to show the parameters for other Wireless Networks frameworks like INET, MiXiM, Castalia, etc. Nevertheless, the values generated are still useful as they are shown currently, but the user has to write them in the proper parameters of the omnetpp.ini file.

GatcomSUMO can be downloaded from the Gatcom web site (`http://atc.umh.es/gatcom/soft/`

gatcomsumo), both the source code and a multi-platform JAR file, although they are still under active development. In addition, the application has been compiled to Windows, Linux, and Mac OS X with the native compiler Excelsior JET, mainly for improving the performance during the visualization of the map.

Acknowledgments

This work was partially supported by the Spanish Government under Grant TIN2015-66972-C5-4-R, co-financed by FEDER funds (MINECO/FEDER/UE). The authors would like to thank to the developers of the open-source software used for our research, namely, SUMO (dlr.de/ts/sumo), OMNeT++ (https://omnetpp.org), and Veins (http://veins.car2x.org). We would also like to express our gratitude to the developers of the Java programming language and the Excelsior JET compiler (https://www.excelsiorjet.com).

10.8 Bibliography

[1] Barr, R., Z. J. Hass and R. van Renesse: *JiST/SWANS: Java in Simulation Time / Scalable Wireless Ad hoc Network Simulator*. http://jist.ece.cornell.edu. [Accessed April 20, 2017].

[2] Chové, Etienne, Jocelyn Jaubert, Frédéric Rodrigo et al.: *Osmose*. http://osmose.openstreetmap.fr/es/map/. [Accessed April 20, 2017].

[3] Haklay, Mordechai (Muki) and Patrick Weber: *OpenStreetMap: User-Generated Street Maps*. IEEE Pervasive Computing, 7(4):12–18, October 2008.

[4] Härri, Jérôme, Marco Fiore, Fethi Filali and Christian Bonnet: *Vehicular Mobility Simulation with VanetMobiSim*. Simulation, 87(4):275–300, April 2011.

[5] Krajzewicz, Daniel, Laura Bieker, Jérôme Härri and Robbin Blokpoel: *Simulation of V2X applications with the iTETRIS system*. In *TRA 2012, Transport Research Arena, April 23-26, 2012, Athens, Greece / Also published in Procedia - Social and Behavorial Sciences, Volume 48, 2012*, Athens, GRÈCE, 04 2012.

[6] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3&4):128–138, December 2012.

[7] Kurczveil, Tamás and Pablo Alvarez López: *eNetEditor: Rapid prototyping urban traffic scenarios for SUMO and evaluating their energy consumption*. In *SUMO 2015 - Intermodal Simulation for Intermodal Transport*, pages 137–160. Deutsches Zentrum für Luft und Raumfahrt e.V., May 2015.

[8] Lownes, Nicholas E. and Randy B. Machemehl: *VISSIM: A Multi-parameter Sensitivity Analysis*. In *Proceedings of the 38th Conference on Winter Simulation*, WSC '06, pages 1406–1413. Winter Simulation Conference, 2006.

[9] Lutnik, Hubert: *JGPSTrackEdit*. https://sourceforge.net/projects/jgpstrackedit/. [Accessed April 20, 2017].

[10] ns-2: *The Network Simulator*. http://www.isi.edu/nsnam/ns/. [Accessed April 20, 2017].

[11] Papaleondiou, L. G. and M. D. Dikaiakos: *TrafficModeler: A Graphical Tool for Programming Microscopic Traffic Simulators through High-Level Abstractions*. In *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, pages 1–5, April 2009.

[12] Pediaditakis, Dimosthenis, Yuri Tselishchev and Athanassios Boulis: *Performance and scalability evaluation of the Castalia wireless sensor network simulator*. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, page 53. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2010. [Accessed April 20, 2017].

[13] Piórkowski, M., M. Raya, A. Lezama Lugo, P. Papadimitratos, M. Grossglauser and J.-P. Hubaux: *TraNS: Realistic Joint Traffic and Network Simulator for VANETs*. SIGMOBILE Mob. Comput. Commun. Rev., 12(1):31–33, January 2008.

[14] Riley, George F. and Thomas R. Henderson: *The ns-3 Network Simulator*, pages 15–34. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[15] Riverbed Technology: *Riverbed Modeler*. http://www.riverbed.com/products/performance-management-control/network-performance-management/network-simulation.html. [Accessed April 20, 2017].

[16] Scholz, Immanuel, Dirk Stöcker et al.: *JOSM*. https://josm.openstreetmap.de/. [Accessed April 20, 2017].

[17] Schünemann, Björn: *V2X simulation runtime infrastructure VSimRTI: An assessment tool to design smart traffic management systems*. Computer Networks, 55(14):3189 – 3198, 2011. Deploying vehicle-2-x communication.

[18] Schweizer, Joerg: *SUMOPy: An Advanced Simulation Suite for SUMO*. Lecture Notes in Computer Science (LNCS). Springer-Verlag Berlin Heidelberg, November 2014. [Accessed April 20, 2017].

[19] Sommer, Christoph, Reinhard German and Falko Dressler: *Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis*. IEEE Transactions on Mobile Computing, 10(1):3–15, January 2011.

[20] Uhrmacher, Adelinde M., Sally Brailsford, Jason Liu, Markus Rabe and Andreas Tolk: *Reproducible Research in Discrete Event Simulation: A Must or Rather a Maybe?* In *Proceedings of the 2016 Winter Simulation Conference*, WSC '16, pages 1301–1315, Piscataway, NJ, USA, 2016. IEEE Press.

[21] Varga, András et al.: *INET Framework*. https://inet.omnetpp.org. [Accessed April 20, 2017].

[22] Varga, András and Rudolf Hornig: *An Overview of the OMNeT++ Simulation Environment*. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pages 60:1–60:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[23] Wegener, Axel, Michał Piórkowski, Maxim Raya, Horst Hellbrück, Stefan Fischer and Jean-Pierre Hubaux: *TraCI: An Interface for Coupling Road Traffic and Network Simulators*. In *Proceedings of the 11th Communications and Networking Simulation Symposium*, CNS '08, pages 155–163, New York, NY, USA, 2008. ACM.

[24] Wessel, Karl, Michael Swigulski, Andreas Köpke and Daniel Willkomm: *MiXiM: the physical layer an architecture overview*. In *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, Simutools '09, ICST, Brussels, Belgium, Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

11 Cooperative Traffic Management for Emergency Vehicles in the City of Bologna

Laura Bieker-Walz; German Aerospace Center
Laura.Bieker@DLR.de

11.1 Abstract

In case of an emergency the incident location must be reached by emergency vehicles as fast as possible. But driving an emergency vehicle is a dangerous task. The risk being involved in accidents, being insured or killed is much higher with an emergency vehicle than with a private vehicle. Cooperative traffic management strategies can support emergency vehicle drivers by switching the traffic lights on the way of the emergency vehicles accordingly to reach the incident location fast and safe. The approach was simulated in SUMO using the simulation scenario of the city of Bologna.

Keywords: Emergency vehicles, V2X Communication, Traffic Management

11.2 Introduction

One essential part of a working health system is that people who are seriously injured or sick receive professional assistance immediately. Therefore, emergency vehicles are allowed to use special rights e.g. having priority at intersections and violating red traffic lights etc. Driving an emergency vehicle is still a stressful situation. The driver is in a conflict situation not only to reach the incident location as fast as possible but also to avoid accidents and critical situations (5). Furthermore, the drivers might mentally prepare for the following operation while they need to concentrate on the surrounding traffic. The aim of this study is to improve the operation drives of emergency vehicles considering traffic safety and travel time.

11.3 Traffic Safety of Emergency vehicles

Statistically, the risk for emergency vehicles having car accidents is much higher than that for other traffic participants (9). A traffic safety study (16) indicates that the probability to have an accident for emergency vehicles is 8 times higher than that for other vehicles. Additionally, the study indicates that at least one person is killed per 272,000 emergency drives. This fact is dramatic, especially considering that emergency vehicles should help other people and prevent death instead of causing injuries and deaths. Other studies confirm also the higher risks for emergency vehicle drivers (e.g. (16), (15), (9), (7)).
To find out which supports emergency vehicles need to prevent accidents an internet research was performed and the results can be found in (2). The description of 189 accidents with emergency vehicles in Germany in the years 2009 and 2015 were collected and evaluated. All accidents happened

while the vehicle were requesting special rights with blue flashing lights and siren. It can be assumed that emergency vehicles which were driving without special rights should have a similar traffic behavior and risk for accidents as normal vehicle and therefore are not considered in this study.
The reasons for the analyzed accidents are shown in figure 11.1. As a result, most of the recorded accidents happened at controlled and uncontrolled intersections (51%). 32% of the reported accidents were due to red light violation. It is not evaluated whether the accidents were the fault of the emergency vehicle driver. Legally, a emergency vehicle driver is obliged to make sure that other drivers have noticed the emergency vehicle and react accordingly and therefore will mostly be blamed for the respective accident at least partly.
Also other reasons are strongly related to the special rights of emergency vehicles e.g. overtaking (12%), abrupt breaking (4%), one way road (0.5%) and stop sign (0.5%).But there are also other reasons which could lead to an accident for every vehicle type for example the influence of alcohol (3%) and icy roads (5%).

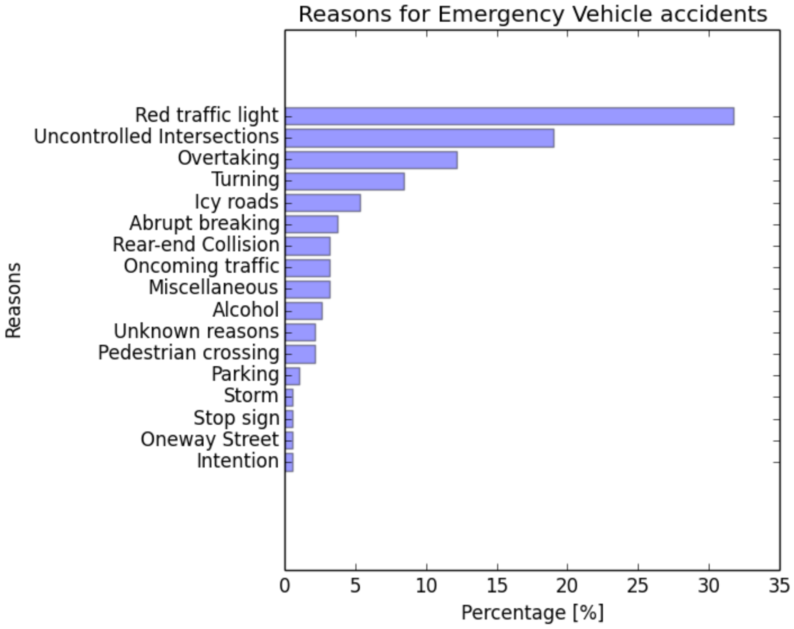


Figure 11.1: Reasons for Emergency Vehicle Accidents

Based on this emergency safety evaluation this study focuses on supporting emergency vehicles at controlled intersections. To reduce the accident risks emergency vehicles should get priority at intersections without violating red traffic lights.

11.4 Prioritization strategy via Vehicle-to-Infrastructure communication (V2I)

This research aims at adapting the traffic lights on the way of emergency vehicles and give the drivers priority at every intersection without violating red traffic lights. Other studies also investigated the impact of priority systems for emergency vehicles (11). In the work of Nelson (11), the priority for emergency vehicles is given manually by a traffic controller. The main idea was to extend the green time durations for emergency vehicles.
Another way to support emergency vehicles is special routing adjusted to the needs on the characteristics of emergency vehicles and has been investigated in the work of Haghani et. al. (8). Haghani et. al. inspect how the routing and dispatching of emergency vehicles could be optimized and therefore

lead to faster routes for emergency services. Furthermore, a special routing for emergency vehicles was introduced in (18) as well.

Vehicle-to-Vehicle and Infrastructure communication (V2X) has the potential to improve traffic conditions in several ways. On the one hand side V2X communication can be used to inform traffic participants about the current traffic state or dangerous road conditions (warnings about icy roads, construction sides or accidents). In the case of supporting emergency vehicles, V2X communication can provide warning messages to inform drivers about an approaching emergency vehicle (14) (6). On the other hand V2X communication can support traffic management solutions with the current traffic state e.g. route recommendations according to the current traffic state or dynamic traffic light schedules. In (13), a navigation model using V2V communication for emergency vehicles was investigated.

The scope of this research is to get the information about an approaching emergency vehicle via V2X communication and adapt the traffic light state automatically on the route of the corresponding emergency vehicle accordingly. The basic approach was already investigated in (1) and is extended and evaluated in this paper. The traffic lights will be switched to green for the direction of the emergency vehicles while the red traffic lights are on to all other traffic participants. The procedure is described as follows: An emergency vehicle on its way to an incident location or the way back sends unicast messages to a road side unit (RSU), which include its expected route and current position. If an RSU receives such a message it will inform the traffic management center which sets the traffic lights on the way of the oncoming emergency vehicle to green while all other vehicles have to stop at a red light. The traffic light controller will continue its normal operation after the emergency vehicle has passed the intersection.

This basic approach has two problems. First, if drivers have to wait at a traffic light for a long time and they do not know why. They might assume that the traffic light is not working properly. Furthermore a long red phase might cause large traffic jams. And the second problem is how more than one emergency vehicle are handled. To overcome these problems the basic approach was extended.

To minimize the time the traffic light is set to red, the application calculates the best time when the traffic light phase should be adapted. The basic approach adapts the traffic light if the emergency vehicle is in communication range (300 meters) of the RSU of the traffic light and has reached the street in front of the traffic light. If there are more traffic information available e.g. loop detector data or V2X communication, the information can be used to calculate how long the traffic light should be set to green for the street of the emergency vehicle to free its way. For each vehicle which is waiting at the controlled intersection it was estimated that it needs 2 seconds for passing the traffic light plus 5 seconds for the emergency vehicle and reaction time for safety reasons:

$$T_{free} = \#waiting\_vehicles * 2sec + safety\_time$$

The distance from the emergency vehicle to the traffic light when the traffic light program should be adapted can be derived from T_{free} and the speed of the emergency vehicle:

$$distance = T_{free} * v_{emergency\_vehicle}$$

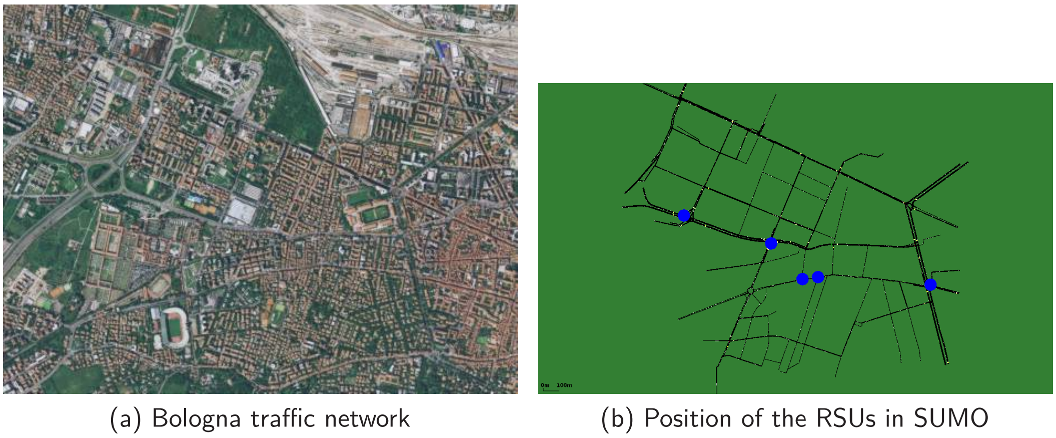
where: $v_{emergency\_vehicle}$ is velocity of the emergency vehicle (m/sec)

This allows the application to change the traffic light phase as long as necessary and as short as possible. The surrounding traffic have enough time to react to the emergency vehicle before they normally would notice the siren and the blue flashing light. Normally, the siren of an emergency vehicle can be perceived by the drivers only up to a distance of 25 meters according to (12). The application extends this distance up to the communication range of 300 meters. This approach has

only one drawback the traffic participants have to be also equipped with V2X communication or at least a induction loop has to count how many vehicles are waiting in front of the traffic light.
Another important issue is how to handle multiple emergency vehicles at the same time. If two or more emergency vehicles are driving at the same time the algorithm checks first whether the routes of them are having an intersection in common. If not the traffic management center can set the traffic lights two green for both routes without having any conflict. Otherwise, the application calculates the estimated time slot of the emergency vehicles at the conflicting intersections. The emergency vehicle which will probably reach the intersection first will get the priority at this intersection and after the emergency vehicle has left the intersection the second emergency vehicle will get priority. If the estimated arrival of both emergency vehicles are in the same interval, the emergency vehicle gets priority with the highest priority class. Ambulances have the highest priority class, second fire brigades and third police cars. If both vehicles have also the same priority class then the vehicle which sent the request first will receive priority and both vehicles will receive a warning message that another emergency vehicle is approaching at the same intersection.

11.5 Simulation

For this research the traffic of the city of Bologna was simulated with SUMO version 0.28 (10) and the used scenario was described in detail in (3). The communication model which was used for the message transmission is described in (4). Bologna has many traffic problems because of crowded and narrow streets. The scenario includes the area of a hospital and the football stadium of Bologna. When football games are taking place the area is extremely crowded. Therefore some lanes are restricted for buses which emergency vehicles are also allowed to use. The road network and the simulated area in SUMO are illustrated in Figure 11.2. In the simulation every traffic light was equipped with an RSU to receive the messages from the emergency vehicles. Every emergency vehicle sends Cooperative Awareness Messages (CAMs) with the current route and the vehicle type information. CAMs are periodically updated and sent and include the current information of the sender. For the message transmission of the V2X communication the standard of the IEEE 802.11p standard is used.
The vehicle type indicates that the emergency vehicle requests priority at the downstream intersections. The RSU triggers the traffic light to switch to green for the direction of the emergency vehicle and to red for all other directions. After the emergency vehicle has passed the intersection, determined again by using the position information from a CAM, the traffic lights continue their normal operations.
Currently there is no model for simulating an emergency vehicle available. This aspect should therefore be the focus of further research. Some aspects like building an emergency lane have been



(a) Bologna traffic network

(b) Position of the RSUs in SUMO

Figure 11.2: Region of Bologna which was used for the simulation

modeled and evaluated in (17). While the application modifies the traffic light program in a way that there is no need for the emergency vehicle to violate red lights, the simulation needs only two small modifications to simulate the behavior of an emergency vehicle for the purpose of this application: having right to use bus lanes and getting priority at roundabouts, see Figure 11.3b.

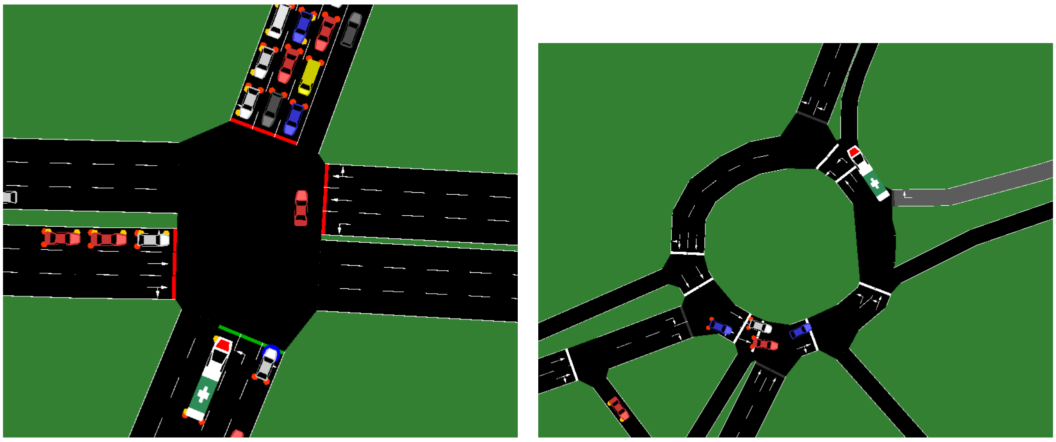
11.6 Results

The simulation was performed 40 times with the same simulation scenario. For comparison 3 different scenarios have been simulated and analyzed. The baseline scenario is an emergency vehicle without surrounding traffic: the emergency vehicle was simulated in an empty traffic net. The second scenario is the simulation of the scenario with the emergency vehicle as a normal vehicle. And the last scenario is the simulation with the application for the emergency vehicle. The result of the average travel time, the waiting time and time loss of the emergency vehicle in the 3 different scenarios can be seen in Table 11.1.

Table 11.1: Simulation Results

Scenario	Travel time (seconds)	Waiting time (seconds)	Time loss (seconds)
EV without traffic	216.25	12.325	32.548
EV with traffic	229.567	19.15	46.0655
EV with Application	210.6	1.7	27.086

As expected the travel time of the emergency vehicles with the application is lower than without the application. It also can be seen that the emergency vehicle is even faster with the application than without any traffic, because of the normal program of the traffic lights. In real life of cause the emergency vehicle would not wait at the red traffic light. But still the results show that the application could support the travel time of emergency vehicles and could reduce the number of stops on its route.



(a) Emergency vehicle receives priority at intersection

(b) Emergency vehicle gets priority at an round about

Figure 11.3: Emergency vehicle on its route in Bologna

11.7 Conclusions

The simulation results indicate that V2I can support the routes of emergency vehicle regarding the aspect of travel time. It was not possible to evaluate the safeness of the routes limited to the fact that it is not possible to simulate accidents in SUMO or other safety criteria. According to the literature (16) and the traffic safety evaluation (2) it is expected that emergency vehicle driving will be much safer if there is no need to violate red traffic lights.

Many V2X applications have the problem that they need a specific amount of equipped vehicles to work properly. The advantage of this applications is that only the emergency vehicles and the traffic infrastructure have to be equipped with the communication technology.

But it also has to be considered that for more reliable investigations of emergency vehicle routes a better model of emergency vehicles and their special driving behavior is needed. In reality the emergency vehicle is expected to reach its destination faster than in the simulation because of its special rights. Also another interesting aspect will be how and when other traffic participants will react to the emergency vehicle.

11.8 Bibliography

[1] Bieker, Laura: *Emergency Vehicle Prioritization using Vehicle-To-Infrastructure Communication*. In *Young Researchers Seminar*, Juni 2011.

[2] Bieker, Laura: *Traffic safety evaluations for Emgergency Vehicles*. In *Young Researchers Seminar*, Juni 2015.

[3] Bieker, Laura, Daniel Krajzewicz, Antonio Pio Morra, Carlo Michelacci and Fabio Cartolano: *Traffic simulation for all: a real world traffic scenario from the city of Bologna*. In *SUMO 2014*, Mai 2014.

[4] Bieker, Laura, Daniel Krajzewicz, Matthias Röckl and Hans Capelle: *Derivation of a fast, approximating 802.11p simulation model*. In *Intelligent Transport Systems Telecommunications (ITST2010)*, November 2010.

[5] Bockting, S.: *Verkehrsunfallanalyse bei der Nutzung von Sonder-und Wegerechten gemäß StVO. Konzeptionelle Vorschläge zur Verbesserung der Aus-und Fortbildung*, 2007.

[6] Buchenscheit, Andreas, Florian Schaub, Frank Kargl and Michael Weber: *A VANET-based emergency vehicle warning system*. In *Vehicular Networking Conference (VNC), 2009 IEEE*, pages 1–8. IEEE, 2009.

[7] Burke, A F: *Batteries and ultracapacitors for electric, hybrid, and fuel cell vehicles*. Proc. IEEE, vol:95no4pp806–820.

[8] Haghani, Ali, Huijun Hu and Qiang Tian: *An optimization model for real-time emergency vehicle dispatching and routing*. In *82nd annual meeting of the Transportation Research Board, Washington, DC*. Citeseer, 2003.

[9] Hempel, O.: *Straßenverkehrsrecht und Rettungsdienst: Grundlagen im Überblick*. Notfall Rettungsmed, 10:367–371, 2007.

[10] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3&4):128–138, December 2012.

[11] Nelson, Eric and Darcy Bullock: *Impact of emergency vehicle preemption on signalized corridor operation: An evaluation*. Transportation Research Record: Journal of the Transportation Research Board, 1727:1–11, 2000.

[12] Optik, Systemtechnik und Bildauswertung IOSB Fraunhofer-Institut für: *Martinshorn-Erkennungs- und Warnsystem*, 2014.

[13] Ravish, V, M M Manohara Pai, M Boussededjra and J Mouzna: *Mobility Model with Inter Vehicular Communication Based Navigation for Emergency Vehicles*. In *Intelligent Transport Systems Telecommunications (ITST2010)*, November 2010.

[14] Strang, Thomas: *Intelligent Transportation Systems: European C2C Activities*, 2010.

[15] Strassenwesen, BAST Bundesanstalt für: *Ursachenuntersuchung innerörtlicher Unfallstellen. Wissenschaftliche Informationen der Bundesanstalt für Straßenwesen.*, 1994.

[16] Strassenwesen, BAST Bundesanstalt für: *Verbesserung der Sicherheit bei Sondersignaleinsätzen. Bergisch-Gladbach, Info 34/95.*, 1995.

[17] Weinert, Florian and Michael Düring: *Development and Assessment of Cooperative V2X Applications for Emergency Vehicles in an Urban Environment Enabled by Behavioral Models*. In *Modeling Mobility with Open Data*, pages 125–153. Springer, 2015.

[18] Woelki, Marko, Ting Lu and Sten Ruppe: *Ranking of alternatives for emergency routing on urban road networks*. In Brebbia, C. A. (editor): *Urban Transport 2015*, volume 146 of *WIT Transactions on the Built Environment*, pages 591–598. WIT Press, 2015.

12 External signal control: Integrating LISA+ into SUMO

Mirko Barthauer and Bernhard Friedrich
Technische Universität Braunschweig, Institute of Transportation and Urban Engineering
{m.barthauer, friedrich}@tu-braunschweig.de

12.1 Abstract

Both traffic engineers and researchers use microscopic traffic simulation to optimize and evaluate signal control with respect to traffic capacity and environmental concerns. While researchers are free to imagine complex actuated control strategies, traffic engineers have to deal with standards from signal control producers. To overcome this obstacle, several simulations offer a model in the loop approach: Signal control is sourced out to an external signal controller model. While TraCI enables interaction with SUMO traffic lights during the simulation, this Python-based interface does not offer any assistance to build actuated control strategies out of standardized logic blocks. On the other side, signal planning programs like LISA+ enable road administrations to develop signal programs ready to deploy. LISA+ also offers a virtual signal controller which communicates with other applications via network messages. In this document, the missing link between both worlds is presented: A Python script controls SUMO and collects signal plan as well as detector information to send it to LISA+. Afterwards, it implements the signal states received from the virtual signal controller. Apart from a technical insight, a case study highlights the advantages of this approach for rule-based actuated traffic control.

Keywords: external signal control, network communication, TraCI

12.2 Introduction

Different categories of software to support traffic engineers in their tasks have emerged in the last decades. Besides traffic simulation of varying detail, dedicated signal control design and traffic planning programs are available. Even if both types of software can be used to design and to evaluate signal plans and actuated signal control logic, each has its own strengths and weaknesses.
Firstly, analytic formulas like those in HBS (2) had been developed out of field studies for evaluating traffic quality of fixed-time signal control. Nowadays, growing complexity of actuated signal control leads to microscopic traffic simulation as the tool of choice.Theoretically, it can reproduce any signal control logic and the traffic flow based on single vehicle-driver units. However, neither is signal control the only argument for microscopic traffic simulation nor do common software of this field focus intensively on the aspect of signal control. Apart from fixed-time signal programs, the supported types of actuated signal control vary between different simulation tools. Especially when it comes to completely customized control logic, the user is often confronted with a big effort to (re)implement it from scratch. From a different point of view, signal control design is done with specific software

which already offers many helpful editors and standardized control logic elements. This software may also provide a model of the signal controller with input and output interfaces.

Coupling traffic simulation software with other programs has been done for several reasons, including modeling of emissions (6), communication (13) or fine-scale vehicle dynamics (9). Hirschmann et al. (7) have introduced the adaptive traffic network control called MOTION to VISSIM traffic simulation by means of an interface for external signal control. They also used the system to calculate emissions with a dedicated model. Some traffic simulation companies have added specific interfaces to allow for external signal control. However, as they cater each only a small subset of the available interfaces, many users cannot use this function.

Since Webster's formula for minimal delays with fixed-time signal programs (see German signal control guidelines RiLSA (3, p. 20)), several concepts for signal control optimisation have been proposed. Both macroscopic approaches like the Cell Transmission Model (see (10)) as well as microscopic simulations (e.g. see (12)) have been used for actuated and coordinated signal control. Within this work, we focus on implementing rule-based actuated signal control. At first, signal program prototypes are presented. They are compared to the modeling capabilities in the microscopic traffic simulation SUMO 0.28.0 (8) and the signal program design software LISA+ 5.2 (11), respectively. To remedy the disadvantages of each, a coupling of SUMO and an external signal control developed in the signal program design software LISA+ is presented. Given the open structure of the signal control part, it could also be easily adapted to other applications like other traffic or driving simulations. Finally, the coupling is applied to a large actuated intersection.

12.3 Signal program structure

RiLSA guidelines (3, p. 38) contain the general concepts and constraints for their domain. Together with the traffic quality indicators defined in the HBS (2), they guide the traffic engineer during the design and the evaluation of signal programs for a specific intersection. Table 12.1 categorizes signal programs at the macroscopic (network) and the microscopic (local intersection) level. Both levels can be relevant as coordination of traffic-actuated intersections has become common practice.

Especially the microscopic actuated signal programs can become complex because they usually consist of a combination of at least two of the prototype signal programs. The changing elements of a phase-oriented signal program are:

- Cycle time
- Phase sequence
- Number of phases
- Green periods
- Offset time

Non-conflicting signal groups with simultaneous green periods are grouped in a phase. Their green

	control strategy	number	activation		traffic-actuated changing elements of the signal program				
			time-dependent	traffic-dependent	cycle time	phase sequence	number of phases	green periods	offset time
A: macroscopic control	program selection	A1	X						
		A2		X					
	traffic-dependent permissive period plan	A3		X					
B: microscopic control	fixed-time	B1							
	green time adjustment	B2						X	
	signal phase change	B3				X			
	phase on request	B4					X		

Table 12.1: Signal program classification (adapted from (3, p. 30))

period can be adapted based on detector measures like time gaps. Some phases may be only shown on request, e.g. for public transport. Another way of public transport priority can be reached by changing the phase sequence. The resulting control logic is documented in flow diagrams or structure charts (see (3, p. 39)) together with a domain-specific programming language.

12.4 Signal control in SUMO

SUMO offers multiple ways to control the signalized intersections (5). On one hand, there is integrated support for fixed-time signal programs, time gap actuated programs and time-dependent activation of signal programs. On the other hand, other prototype signal programs from table 12.1 have to be realized as external control with the TraCI interface which allows a direct access to the signal states. Changing the signal phase sequence and optional phases can be implemented this way.

12.5 LISA+ actuated control

LISA+ is designed with RiLSA (3) and HBS (2) requirements in mind: It helps to create a signal control definition with (phase-based) signal programs like the RiLSA examples (1). There are specific editors for the intersection geometry, signal groups, signal heads and detectors. The signal program can be generated out of signal phase sequences with dedicated transitions taking into account the intergreen times due to conflicts between traffic participants. Permission plans allow to restrain phase durations to a part of the cycle time. Additionally, special programs for switching the signal control on and off need to be set. This way, all operation modes of the signal control are included. By definition, LISA+ builds rule-based actuated signal logic upon existing fixed-time signal programs. It offers the so-called OML library which consists of many functions to query the signal control state and to trigger actions. By means of flow diagrams, these functions can be compiled to a custom logic for actuated signal control. An example is shown in the case study (see figure 12.5). The signal design program also includes a limited testing section for single intersections. There detectors can be either activated manually or through a simple traffic simulation to watch the resulting signal states.

12.6 Model in the loop approach

External signal control for microscopic traffic simulations means implementing a model in the loop (MiL) approach. There is at least one model for signal control and there is SUMO simulation responsible for the traffic flow. The signal control is considered as an embedded system: it reacts to the traffic flow and influences it with the signal states, too. All in all, the simulation consists of SUMO, its Python-based TraCI interface and the simulated LISA+ controller (a Java executable called OmlFgServer.jar). In terms of signal control, the LISA+ part acts as a server and TraCI as a client. Both communicate via network comunication using the SOAP protocol: messages include a header part and a XML body. On the TraCI side, a Python-written library deals with sending requests and processing the answers. Figure 12.2 explains the relation between the Python library and the SUMO network: The library accesses the intersection objects to gather the settings for the LISA+ controller. This includes the chosen signal program, whether it should be actuated, coordinated and further options.

Consequently, object information from LISA+ has to be linked to SUMO objects and vice versa. Common SUMO and LISA+ objects are linked by their numeric identifier. By default, SUMO detector objects are not bound to a specific signal control but have to for LISA+. Further on, the communication protocol of the LISA+ controller is described and the linkage of LISA+ and SUMO objects is explained.

12.6.1 LISA+ controller communication protocol

The LISA+ controller needs to know which signal program to show and to which kind actuated control shall be available. It is defined by the following fields:

1. controlMode: whether to control the intersection indepently of others
2. coordinated: whether to use coordination information
3. sp: number of the signal program
4. va: whether to use actuated logic
5. oev: marker to enable/disable public transit related actuation parts
6. iv: marker to enable/disable private transport related actuation parts

Originally, LISA+ support for microscopic traffic simulation has been limited to VISSIM networks (11). As all the necessary information for LISA+ controlled intersections can be entered in a graphical interface in VISSIM, no developer documentation of the communication with the LISA+ controller has been supplied. Thanks to Wireshark (4), a listener program for network communication, messages between VISSIM and LISA+ have been recorded and interpreted. The VISSIM simulation network to gather the messages consisted of a small intersection with two signal groups and three detectors. Additionally, the LISA+ controller comes with some description files in WSDL (Web Services Description Language) format about the available message types.

Fortunately, each message type appears meaningful and can be linked to specific objects and tasks from the simulation or the LISA+ controller. The message types listed below deal with the essential questions to control traffic signals. Except for PutMessage, these message types are sent only at the simulation start. A concise description of the message contents follows:

- GetTaskListRequest: Ask the LISA+ controller for running tasks. A task is defined as providing the signal states for one intersection.
- SetDataDirRequest: Inform the LISA+ controller of the directory for the LISA+ signal control export files. These contain the actuated control logic.
- RemoveTaskRequest: Ask the LISA+ controller to end a given task. May be useful when the LISA+ controller runs longer than one simulation run or if a signal control has been switched off.
- SetTaskRequest: Ask the LISA+ controller to provide signal states for a given signal control.
- GetObjectListRequest: Return information about the signal control components from the LISA+ controller. This includes signal groups, detectors, signal programs, phases...
- PutMessage: Either switch the signal control on/off or ask for the next second's signal states. Here, the detector states from SUMO are transmitted to the LISA+ controller.

A typical sequence of exchanged messages between the LISA+ controller and the Python/TraCI part is shown in figure 12.1. As messages of the PutMessage type contain the essential information exchanged between the LISA+ controller and SUMO, they are described more in detail as they have been understood here.

The message in figure 12.2a requests the next signal state for task 9 at simulation second 192 given the options in the second curly brackets, where e.g. actuated control can be activated. The third pair of curly brackets adds information about detector occupation. Positive values indicate a car entering

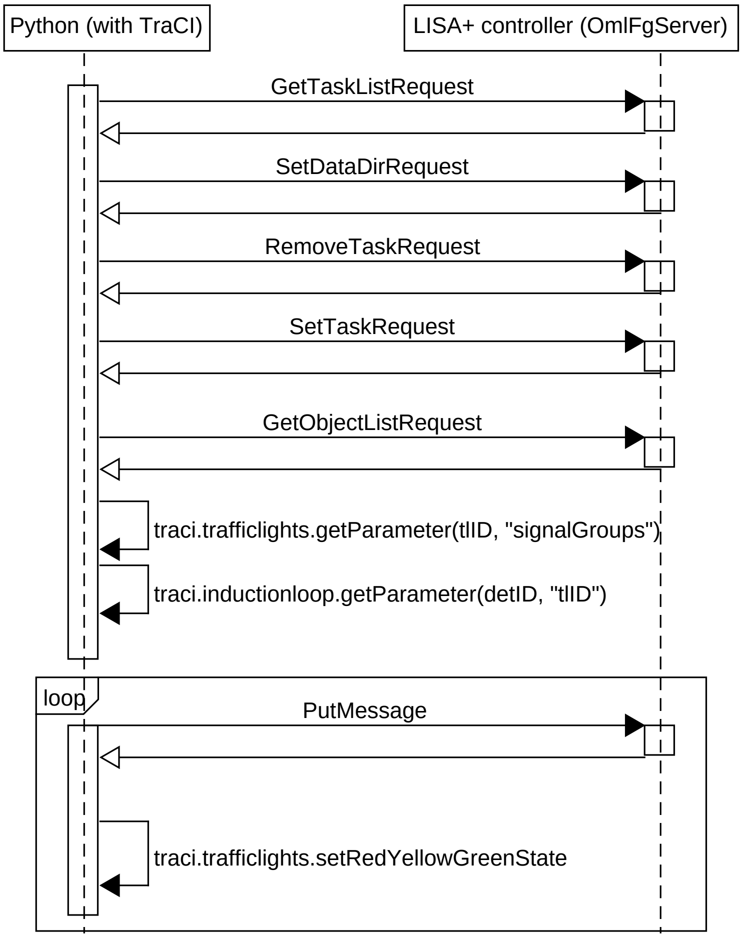

Python (with TraCI)
LISA+ controller (OmlFgServer)
GetTaskListRequest
SetDataDirRequest
RemoveTaskRequest
SetTaskRequest
GetObjectListRequest
traci.trafficlights.getParameter(tlID, "signalGroups")
traci.inductionloop.getParameter(detID, "tlID")
loop
PutMessage
traci.trafficlights.setRedYellowGreenState

Figure 12.1: Sequence diagram of exchanged messages between the LISA+ controller and Python (with TraCI)

the detector and negative ones mean the detector has been freed. The prefix numbers indicate which detectors have been skipped because of inactivity: Here, only detectors 2 and 6 in the order defined by LISA+ have registered vehicle movements. Each message sent to the server is followed by an answer. For each simulated second, the signal states are applied to the SUMO intersections using the TraCI traffic light functions. Figure 12.2b shows the corresponding answer to the preceding request:

It contains the signal states for cycle second 35 at simulation second 192. The third pair of curly brackets lists the signal state per signal group in the order defined by LISA+. Here, 3 and 48 mean red and green, respectively. The fifth pair of curly brackets indicates which phase or transition the signal controller is executing: In this case, the third phase is active.

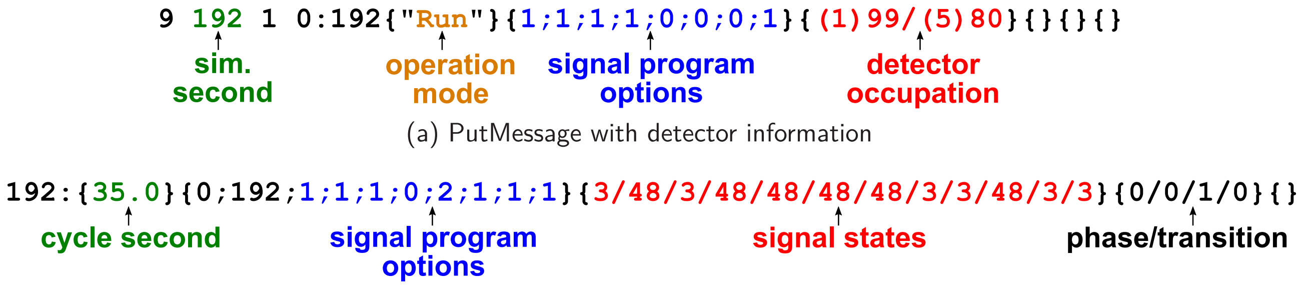
12.6.2 SUMO/TraCI extension

Due to different needs, SUMO and LISA+ differ in some points with respect to signal control. While the RiLSA guidelines (3) and LISA+ use signal groups to address traffic flows released simultaneously, SUMO uses a one signal to one flow relation by default. Transitions between signal phases are seen as a clearly distinguished set of signal changes in LISA+ which is not the case for SUMO. As detectors play an important part in actuated signal control, LISA+ defines them independently for each intersection. SUMO however uses them for several purposes and does not link them to traffic lights.

The idea is to add all necessary information to SUMO with additional object attributes. Currently there is no preferred way to add them, but key-value pairs can be added to some of them. At this moment, they are mainly used to set commands for actuated signal control logic. Support for reading these attributes from the network XML file is limited, but at least TraCI library has been prepared for a wider use together with lanes and detectors.

As more LISA+ controlled intersections may share detectors with the same identifiers, a custom detector attribute defines the unique relationship with the intersection. Global settings like the file directory of the LISA+ control files and the path to the LISA+ controller have to be given as parameters to Python at this stage of development. Figure 12.2 shows the attributes needed for the coupling of SUMO with LISA+-based external signal control.

The TraCI Python package already allows to get and set key-value pairs for simulation elements like detectors, junctions, traffic lights and more. But only traffic light related key-value pairs are read from the network file into SUMO. This can be corrected by either using another configuration file for the Python script or make SUMO understand more complex network files. As most of the new attributes belong to elements from the network file, building a separate configuration file seemed unnecessary. Instead, extended support for key-value pairs in network files was implemented in SUMO.



```
9 192 1 0:192{"Run"}{1;1;1;1;0;0;0;1}{(1)99/(5)80}{}{}{}
```

(a) PutMessage with detector information

```
192:{35.0}{0;192;1;1;1;0;2;1;1;1}{3/48/3/48/48/48/48/3/3/48/3/3}{0/0/1/0}{}
```

(b) PutMessage answer

12.7 Case study

12.7.1 Intersection

RiLSA (1) offers many examples modeled after real-world sites but still representing categories from table 12.1. The chosen example intersection shown in figure 12.3 has all approach lanes detected and includes protected and permissive left turns signals. In the following paragraphs, the signal program is described as well as the test results from SUMO. Every variant of the signal program is based on the same phase sequence (see figure 12.4) as well as the same control logic. However it may use a different cycle time, permission plan or different maximum green times for the signal groups. The general phase sequence is as follows:

- Phase 1
- Continue to phase 2 if the headway from left exceeds a threshold and the left turn lane from the right is congested, skip otherwise
- Phase 3
- Continue to phase 4 if the headway from top exceeds a threshold and the left turn lane from the bottom is congested, skip otherwise

To make the signal program work in SUMO, some adjustments have to be done with respect to the integrated signal control. Traffic lights and detectors to be used with external signal control need to have their additional attributes to be filled by hand.

12.7.2 Runtime investigation

The MiL approach adds some complexity to the simulation by outsourcing the signal control to an external software. Additionally, the communication between SUMO and the LISA+ controller takes some time. This can become an issue if the external control is used for research questions asking for many simulation runs like optimisation. Anyway, to get a rough impression of how much the external control has an impact on the simulation runtime, it has to be compared with the internal control using simulations with varying intersection numbers.

Four experiments are executed and measured: one or two identical intersections with either fixed-time signal program or external control. All last 1800s with 0.2s time step and lack traffic to concentrate on the mere signal control aspect. The results are shown in table 12.2. Even if measuring the runtime

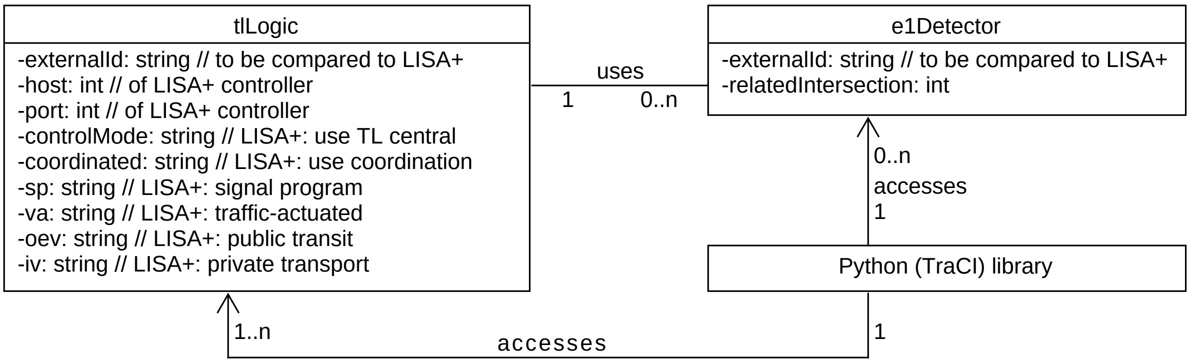

tlLogic
-externalId: string // to be compared to LISA+
-host: int // of LISA+ controller
-port: int // of LISA+ controller
-controlMode: string // LISA+: use TL central
-coordinated: string // LISA+: use coordination
-sp: string // LISA+: signal program
-va: string // LISA+: traffic-actuated
-oev: string // LISA+: public transit
-iv: string // LISA+: private transport
uses
1
0..n
e1Detector
-externalId: string // to be compared to LISA+
-relatedIntersection: int
0..n
accesses
1
Python (TraCI) library
1..n
accesses
1

Figure 12.2: Extension of SUMO network elements' attributes

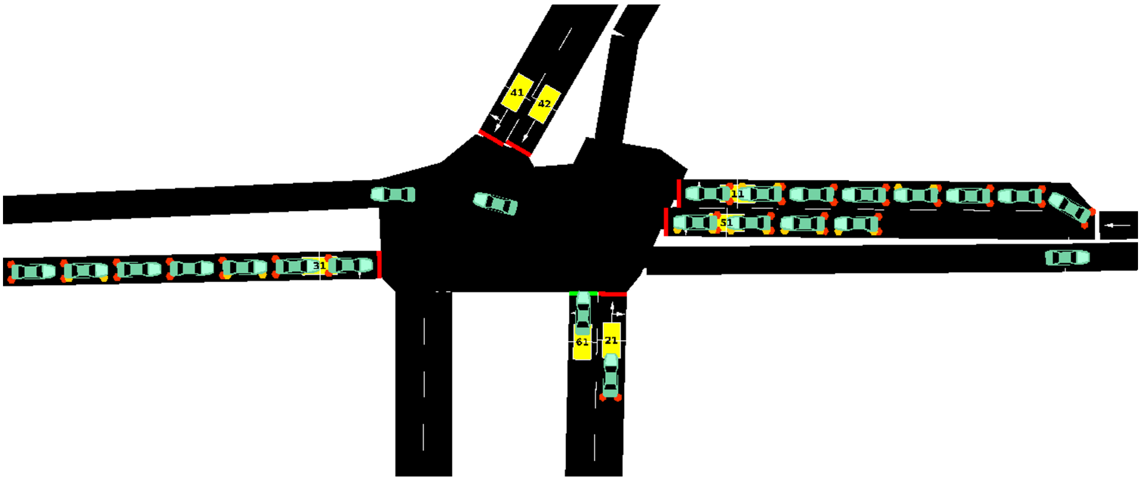


Figure 12.3: intersection from (1, p.30) (chapter 3.6) in SUMO

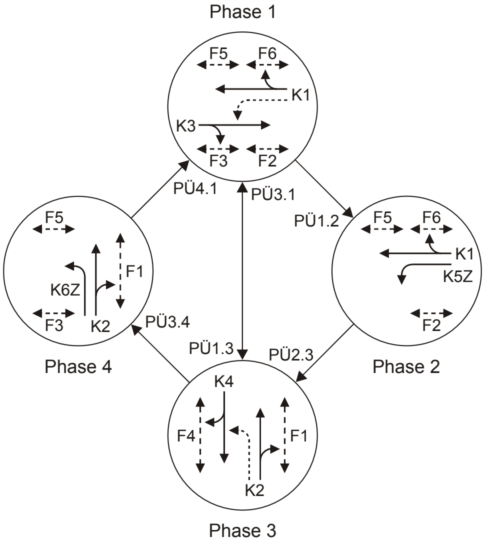


Figure 12.4: signal phase sequence diagram from (1, p.30) (chapter 3.6)

# intersections	avg. runtime fixed-time control [s]	avg. runtime external control [s]
1	29	66
2	30	61

Table 12.2: SUMO runtime for simulation of 1800s with 0.2s time step

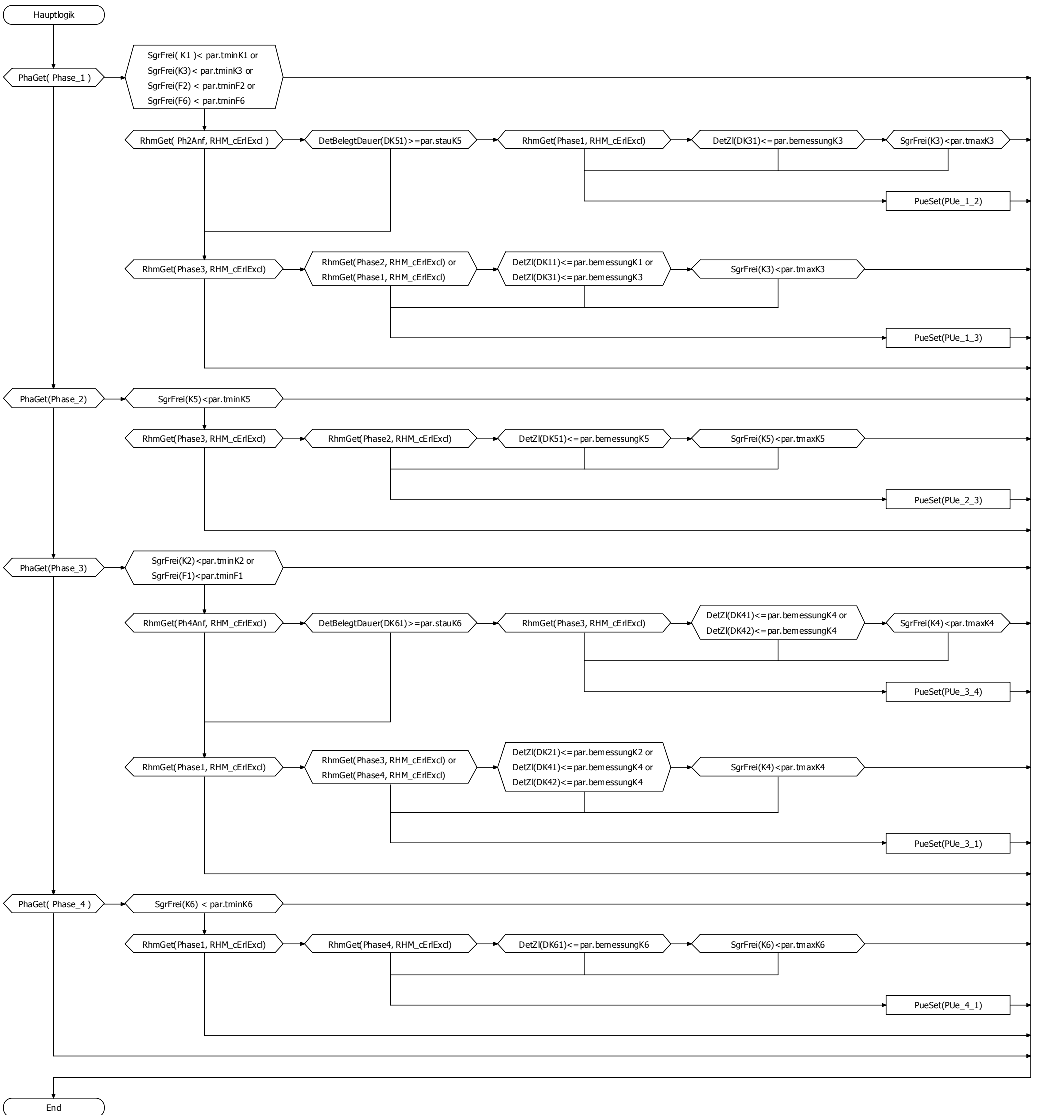

Hauptlogik
PhaGet(Phase_1)
SgrFrei(K1)< par.tminK1 or
SgrFrei(K3)< par.tminK3 or
SgrFrei(F2) < par.tminF2 or
SgrFrei(F6) < par.tminF6
RhmGet(Ph2Anf, RHM_cErlExcl)
DetBelegtDauer(DK51)>=par.stauK5
RhmGet(Phase1, RHM_cErlExcl)
DetZl(DK31)<=par.bemessungK3
SgrFrei(K3)<par.tmaxK3
PueSet(PUe_1_2)
RhmGet(Phase3, RHM_cErlExcl)
RhmGet(Phase2, RHM_cErlExcl) or
RhmGet(Phase1, RHM_cErlExcl)
DetZl(DK11)<=par.bemessungK1 or
DetZl(DK31)<=par.bemessungK3
SgrFrei(K3)<par.tmaxK3
PueSet(PUe_1_3)
PhaGet(Phase_2)
SgrFrei(K5)<par.tminK5
RhmGet(Phase3, RHM_cErlExcl)
RhmGet(Phase2, RHM_cErlExcl)
DetZl(DK51)<=par.bemessungK5
SgrFrei(K5)<par.tmaxK5
PueSet(PUe_2_3)
PhaGet(Phase_3)
SgrFrei(K2)<par.tminK2 or
SgrFrei(F1)<par.tminF1
RhmGet(Ph4Anf, RHM_cErlExcl)
DetBelegtDauer(DK61)>=par.stauK6
RhmGet(Phase3, RHM_cErlExcl)
DetZl(DK41)<=par.bemessungK4 or
DetZl(DK42)<=par.bemessungK4
SgrFrei(K4)<par.tmaxK4
PueSet(PUe_3_4)
RhmGet(Phase1, RHM_cErlExcl)
RhmGet(Phase3, RHM_cErlExcl) or
RhmGet(Phase4, RHM_cErlExcl)
DetZl(DK21)<=par.bemessungK2 or
DetZl(DK41)<=par.bemessungK4 or
DetZl(DK42)<=par.bemessungK4
SgrFrei(K4)<par.tmaxK4
PueSet(PUe_3_1)
PhaGet(Phase_4)
SgrFrei(K6) < par.tminK6
RhmGet(Phase1, RHM_cErlExcl)
RhmGet(Phase4, RHM_cErlExcl)
DetZl(DK61)<=par.bemessungK6
SgrFrei(K6)<par.tmaxK6
PueSet(PUe_4_1)
End

Figure 12.5: LISA+ flow diagram of the rule-based actuated logic from (1, p.34) (chapter 3.6)

can be influenced by other computer processes, the big picture is clear: The external control slows the simulation down by around 100%. This holds true for both tested network sizes.

12.8 Conclusion

Up to now, some actuated signal control logic with optional phases or changes in the phase sequence can only be brought to the microscopic traffic simulation SUMO by means of external signal control. The signal design program LISA+ offers extended support for actuated signal control logic but could not be used together with SUMO. Thanks to an adaptation of the coupling to VISSIM, the LISA+ virtual controller can be used as model in the loop for external signal control in SUMO. This opens new perspectives as users of both software can make use of the control logic design in LISA+ and test it in SUMO. In our Python-based implementation, unfortunately this comes along with a massive slowing down of the simulation performance. Further investigation has to be done to optimise this intermediate part between SUMO and LISA+ and on the scalability of the approach for bigger simulation networks.

12.9 Bibliography

[1] *Beispielsammlung zu den Richtlinien für Lichtsignalanlagen RiLSA*. FGSV ; 321. FGSV Verlag, Köln, 2010 edition, 2010.

[2] *Handbuch für die Bemessung von Strassenverkehrsanlagen : HBS*. FGSV ; 299. FGSV-Verlag, Köln, 2015 edition, 2015.

[3] *Richtlinien für Lichtsignalanlagen : RiLSA : Lichtzeichenanlagen für den Straßenverkehr*. FGSV ; 321. FGSV Verlag, Köln, 2015 edition, 2015.

[4] *Wireshark website*, 2015. Accessed 01.11.2016 at `https://www.wireshark.org/`.

[5] *Simulation of Urban MObility - Wiki*. http://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-_Wiki, 2017.

[6] Hausberger, Stefan and Daniel Krajzewicz: *COLOMBO: Deliverable 4.2. Extended Simulation Tool PHEM coupled to SUMO with User Guide*. Technical Report, TU Graz / DLR, 2014. Accessed 03.08.2016 at `http://elib.dlr.de/98047/1/COLOMBO_D4.2_ExtendedPHEMSUMO_v1.7.pdf`.

[7] Hirschmann, Karin, Michael Zallinger, Martin Fellendorf and Stefan Hausberger: *A New Method to Calculate Emissions with Simulated Traffic Conditions*. In *13th International IEEE Annual Conference on Intelligent Transportation Systems*, Madeira, 2010.

[8] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Bieker Laura: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3-4):128–138, 2012.

[9] Langenberg, Jens: *Auslegung, Simulation und Bewertung von Verkehrsassistenzsystemen durch die Kopplung mit Verkehrsflussmodellen*. PhD thesis, Technische Universität Braunschweig, 2015. Accessed 03.08.2016 at `http://digisrv-1.biblio.etc.tu-bs.de:8080/docportal/servlets/MCRFileNodeServlet/DocPortal_derivate_00041888/Diss_Langenberg_Jens.pdf`.

[10] Pohlmann, Tobias: *New approaches for online control of urban traffic signal systems*. PhD thesis, Technische Universität Braunschweig, 2010. Accessed 18.04.2017 at http://www.digibib.tu-bs.de/?docid=00035257.

[11] Schlothauer & Wauer Ingenieurgesellschaft für Strassenverkehr mbH: *LISA+ website*, 2017. Accessed 13.02.2017 at http://www.schlothauer.de/softwaresysteme/lisa.

[12] Stevanovic, Aleksandar: *Assessing deterioration of pretimed, actuacted-coordinated, and SCOOT control regimes in simulation environment*. PhD thesis, University of Utah, 2006. Accessed 18.01.2017 at http://faculty.eng.fau.edu/stevanovic/files/2014/03/Dissertation_Final.pdf.

[13] Tchouankem, Hugues, Daniel Schmidt and Henrik Schumacher: *Impact of Vehicular Communication Performance on Travel Time Estimation in Urban Areas*. In *Networks for Mobility 2012 - 6th International Symposium*, Stuttgart, 2012.

13 SUMO-Cadyts calibration with limited data quality

Yun-Pang Flötteröd; German Aerospace Center
{Yun-Pang.Floetteroed}@DLR.de

13.1 Abstract

Traffic simulation has been often used to support decision making in traffic management. Such simulation requires a large amount of fine data which cannot be easily obtained in practice especially when a large network is considered. Under this circumstance, the respective calibration work is challenged. In this study, the route choice calibration of SUMO and Cadyts is applied with a real case study in Hefei, China, to examine (1) to which extent the calibration can achieve and (2) which issues need to be especially taken into account. With the resource limitation the network data is based on the OSM with insufficient quality. Major adjustments are made on the number of lanes, turning lane allocations, speed limitations and traffic signal plans. The last two parts still suffer from the limited data quality. The collected flow data for calibration is also limited for only 15 minutes at five major intersections during different moments in the evening peak hour. The relative absolute error is used as the performance indicator. The results show that the simulation quality is improved with the given limited data and quality, but only to a certain degree, namely the percentage of the investigated links with a flow deviation less than 30% has increased from 38% to 72%. The spectrum of the relative errors has been greatly narrowed after the calibration. Gained experiences related to the application of SUMO-Cadyts calibration are also pointed out as references for further applications.

Keywords: SUMO, Cadyts, Route Choice Calibration, OptimUM

13.2 Introduction

Microscopic traffic simulation has been successfully and extensively applied both for investigating different traffic manoeuvres, phenomenon and problems and for quantitatively evaluating various design alternatives and management strategies. Such kind of traffic simulation has also become an essential tool for supporting decision making in traffic management. In order to obtain accurate results traffic network needs to be calibrated firstly. More and more traffic related data has become available with use of innovative technologies, such as Bluetooth and GPS, and can be used for calibration purposes. However, traffic flows are still the most often used data for network calibration not only due to their easy availability and accessibility, but also due to that lots of well-developed calibration models use traffic flows as reference data. Cadyts (8; 4; 2) is one of these calibration models and has been coupled with different simulation software, such as SUMO (11; 9), MATSim (10) and DRACULA (1). These couplings have been successfully tested with some academic synthetic networks respectively. Furthermore, the large traffic network of the City of Zurich has been well calibrated with use of MATSim and Cadyts.

MATSim is a complex agent-based simulation and focuses on traffic flow simulation, re-planning and so on. In comparison to that SUMO also focuses on traffic simulation, but more on the simulation of driving behaviours in road sections, between lanes and at/within intersections as well as the simulation of vehicular communication. Each trip has only one origin and one destination, i.e. no trip chains are considered, in SUMO. Besides, SUMO needs much finer network data than MATSim, such as signal timing plans or priority rules at intersections and the allocation of turning lanes at intersections. In practice, such information cannot be easily obtained especially when dealing with a large city network and is often partially collected and partially estimated with best knowledge. Under this circumstance, the respective data quality is quite limited. Thus, this paper focuses on the performance of the SUMO-Cadyts calibration with limited data quality and examines to which extent the calibration can achieve and which issues need to be especially taken into account.

13.3 Approach

Cadyts is a Bayesian-based route choice calibration model. As mentioned before, the coupling of SUMO and Cadyts has been established and been tested with some academic synthetic networks (2; 5; 6). Cadyts can also deal with congested conditions (3).The basic concept of the SUMO-Cadyts calibration is that SUMO's DUAROUTER generates draws from a prior distribution of route choices so that the equilibrium of route alternatives is reached. The calibration model Cadyts makes adjustments such that a prior distribution of route choices turns into the posterior distribution given all available traffic flow measurements.

Generally, Cadyts adjusts trips' route choices together with the respective departure times after each traffic simulation until the convergence or the predefined number of iterations is reached. The route choice calibration and the traffic simulation (together with its own route choice model) interplay with each other during iterations. However, such a way does not work with SUMO. SUMO's route choice model, i.e. the Gawron model (7), does not take travellers' perception about travel time into account, i.e. drivers use only the respective fastest route, while the logit-based models do consider travellers' perception as an error term in the respective equations. The Gawron model modifies Cadyts' results after each calibration and it leads to an inconsistent correction of the route choice distribution and a poor calibration performance. Thus, the interaction between SUMO and Cadyts is reduced as follows: DUAROUTER is used to generate route alternatives for each driver before applying the SUMO-Cadyts calibration. After that, Cadyts and SUMO (only traffic simulation) start to interplay with each other until the predefined number of iterations or the convergence is reached. The script cadytsIterate.py serves for the interplay of Cadyts and SUMO traffic simulation. It is possible to scale the traffic demand, i.e. the number of trips, with Cadyts just in case that the original demand is underestimated. The whole calibration process is illustrated in Figure 13.1.

13.4 Real Case Study

The applied network is based on the network built in the project OptimUM which is financially supported by the Helmholtz Association and Chinese Academy of Sciences (CAS). This project aims at investigating traffic-related environmental issues with use of traffic simulation and camera monitoring in the science and technology park of the Hefei City, China. This science and technology park locates at the western side of Hefei downtown (see Figure 13.2). Due to the resource limitation, the simulation network is based on the OpenStreetMap (OSM). However, the respective information in the OSM is very limited. The network data and the traffic signal data are therefore partially collected and partially estimated with best knowledge. The built network is shown in Figure 13.3.

Moreover, the applied traffic demand is derived from the existing VISUM daily traffic model and only trips with cars and trucks are available. This daily traffic demand is then converted to hourly traffic demand according to the respective time series of traffic flow data. There are totally 115 traffic analysis zones (TAZs) where 68 zones are within the study cordon. The other 47 TAZs are virtual

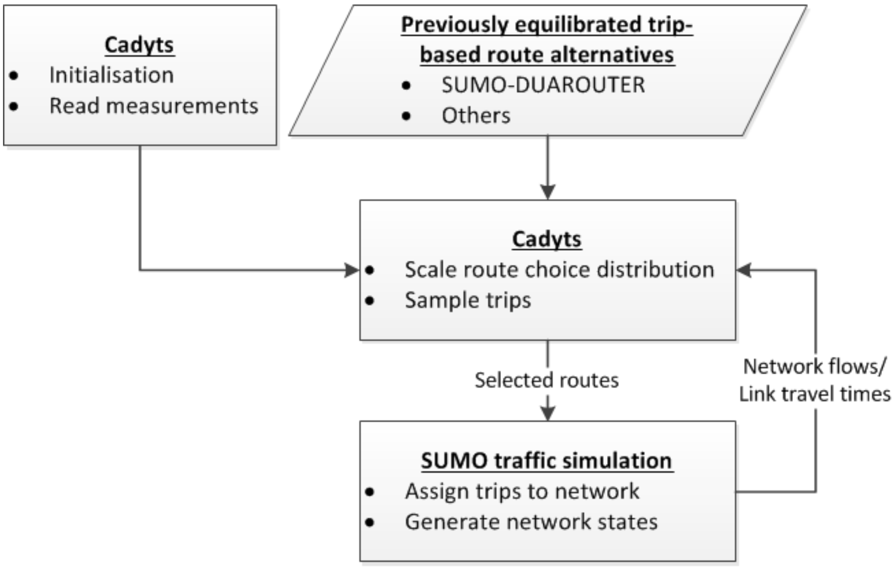


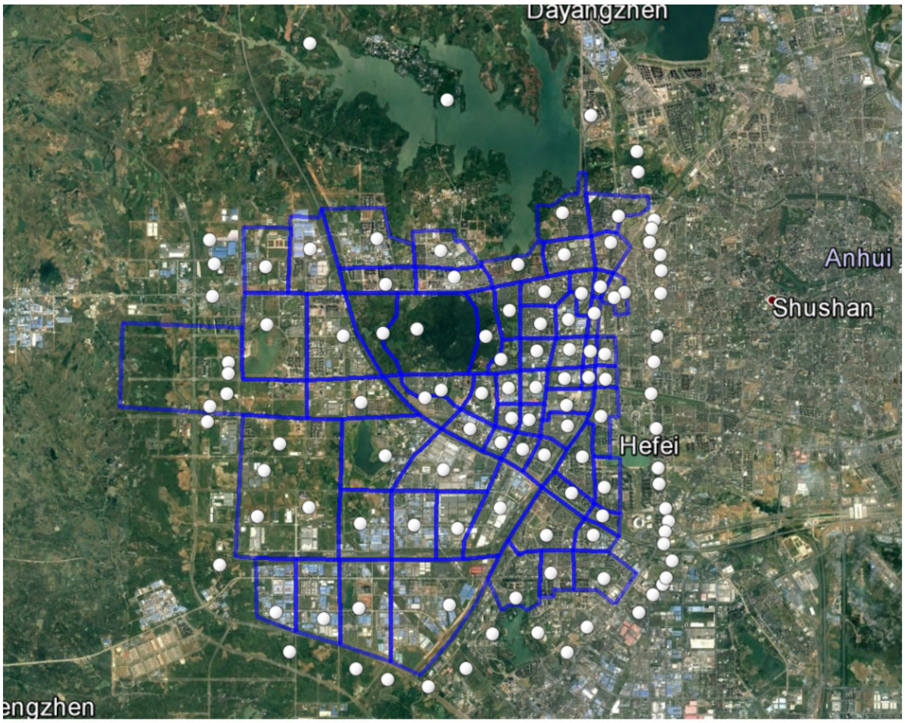
Figure 13.1: SUMO-Cadyts Calibration process

zones for addressing internal and through trips. The respective centroids are shown as white circles in Figure 13.2. The total number of trips is around 40,000 during the evening peak period between 17 and 19 o'clock. Furthermore, traffic counts at 21 edges closed to the major intersections, shown in Figure 13.3, are manually collected for 15 minutes during the evening peak period and are used as reference data in the calibration. The collected flows are between 100 and 650 vehicles.
According to the observation, the major intersections in the study cordon often suffer from traffic congestion during the peak period. Many flexible driving manoeuvres, such as frequent lane changing, using road shoulders, turning lanes used as through lanes or the other way around and headway shortening, have been executed in order to find gaps and to drive through the congestion areas as quickly as possible. Some of these driving manoeuvres, which increase the corresponding road and intersection capacities during congestion, cannot be simulated with SUMO yet. To deal with such manoeuvres in the calibration, teleporting time is set as 150 seconds and the gap acceptance is reduced in SUMO. For generating route alternatives with DUAROUTER, the teleporting time is however set as 300 seconds so that there are sufficient route alternatives for each trip. The gap acceptance parameter in SUMO's car following model is correspondingly reduced as well. Moreover, the simulation begin time is adjusted, i.e. to set the time earlier than the trips' planned begin times, since the trips' begin times may be adjusted by Cadyts. In addition, the traffic demand is scaled with 1.5 times in the calibration to ensure a sufficient traffic demand. This additional demand will be used only when more trips with certain routes are needed, given the flow measurements.

13.5 Results

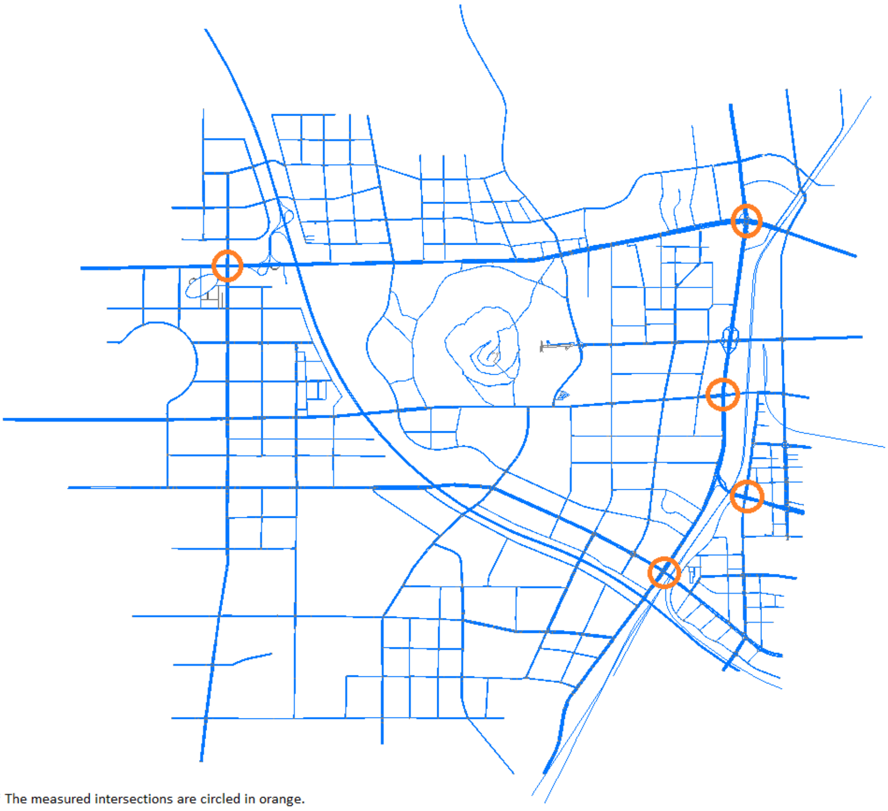
13.5.1 Deviation between measured and simulated flows

The absolute relative error is used for evaluate the calibration results although the empirical GEH formula (12) is often used for comparing two sets of traffic flows. The main reason is that the collected traffic counts are for 15 minutes while the GEH formula requires hourly traffic volumes. Due to the shortage of man power all link flows are counted for 15 minutes in the peak period, but not at the same time. The flow conversion from 15 minutes to 1 hour is thus discarded in order to avoid the possible error propagation (error enlargement).
Figure 13.4 indicates that the difference between the simulated (the orange squares) and measured link (the blue diamonds) flows is quite large before calibration. Most of the flows on the measured



* The TAZs within the study cordon are the areas bounded in blue.
* The centroids of the TAZs are indicated as white circles.

Figure 13.2: The investigated network area in Hefei, China



* The measured intersections are circled in orange.

Figure 13.3: The layout of the investigated simulation network

links are underestimated with the simulation. There are only 38% of links with an absolute relative error less than 30%. After the calibration, the respective percentage increases to 72%. The number of the links with the underestimated (more than 20%) flows has then reduced from 71% to 33%. When observing the simulation, it is also noticed that one of the main reasons for the flow underestimation is traffic congestion that impedes vehicles reaching the measured location in the measurement period (15 minutes). In addition, a few of the link flows are overestimated. It might also be due to traffic congestion which makes vehicles choose other 'cheaper' routes.

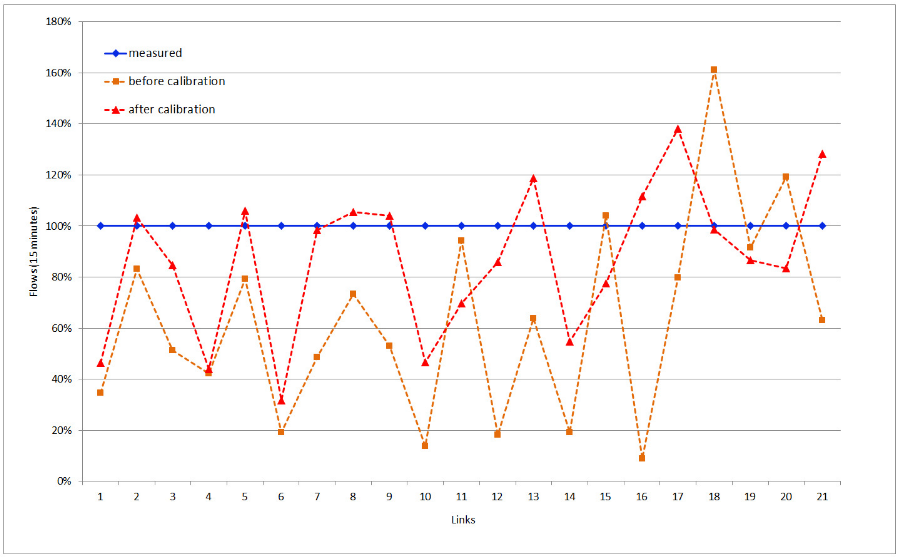


Figure 13.4: Link flow comparison before and after the calibration

As reference comparison the GEH-values with 15-minutes flows are calculated for all measured links. Table 13.1 shows that 67% of the measured links have a GEH-value less than 5 and the corresponding calibration results are an acceptable fit. Such finding corresponds to the results based on the relative absolute error. 14% of the links have possible model errors or bad data and 19% of the links are with high probability of modelling error or bad data.
Furthermore, the range of the error distribution becomes significantly narrower after the calibration (see Table 13.2). The link absolute relative errors spread quite regularly in the spectrum between 0.1 and 1 before the calibration. After the calibration, the corresponding errors are between 0.1 and 0.7 where around 60% of the links have an absolute relative error less than 20%.

13.5.2 Gained Experience

Instead of the link-based calibration, Cadyts calibrates route choice distribution in a given network according to the given route alternatives and link flow measurements. If the given route alternatives do not cover the measured links, Cadyts cannot make any route choice adjustment and the results will be miserable. The quality of the calibrated demand, i.e. the routes, is highly dependent on the locations of the collected data. The more the precisely collected data could cover the mainly

Table 13.1: GEH-evaluation based on the 15-minutes flows

GEH-threshold	Meanings	Percentage of the links
<5	Acceptable fit, probably OK	67%
$5-10$	Possible model error or bad data	14%
>10	High probability of modelling error or bad data	19%

Table 13.2: Distribution of the link absolute relative errors

Relative absolute error	Before calibration (%)	After calibration (%)
<0.1	14	29
0.1−0.2	10	29
0.2−0.3	14	14
0.3−0.4	10	5
0.4−0.5	10	5
0.5−0.6	10	14
0.6−0.7	10	4
0.7−0.8	9	0
0.8−0.9	19	0
0.9−1.0	4	0
>1.0	0	0
Total	100	100

used routes in a study network, the more representative the calibrated demand would be. Therefore, attentions need to be paid to the following issues when the SUMO-Cadyts calibration is adopted.

1. Route alternative generation

   DUAROUTER is responsible for generating route alternatives with the given demand and network characters. It searches a new fast route for every trip at each iteration. If a new route is found, it will be added in the respective route set. Therefore the number of iterations needs to be appropriate and the network data needs to be 'reasonable' in order to collect sufficient route alternatives. Furthermore, the option 'keep all routes' should be activated in order to avoid some routes being thrown away.

2. Teleporting time setting

   Generally speaking, an adequate teleporting time setting is necessary in SUMO in order to prevent possible deadlocks during simulation. Teleporting time adjustment is also a way to partially reflect the different driving manoeuvres in different areas and situation in the simulation. However, a short teleporting time may result in insufficient route alternatives, since the edge travel times are underestimated and only a certain amount of routes will be found.

3. TAZ connectors' definition

   In this study, a TAZs-based OD-matrix is used for generating link-based trips. It is therefore necessary to define the links for connecting each TAZ to the given network. The defined connecting links greatly influence route searching results as well as route alternatives. Thus, attention needs to be paid to this issue. For example, the overlapping of the connectors between TAZ should be avoided.

4. Priority rules and intersection configuration setting

   As mentioned before, intersection capacity plays a crucial role in the temporal distribution of trips in the network and the respective route alternatives/choices. Besides traffic signals, priority rules and intersection configuration, such as the number and the length of left-turn or right-turn lanes, also decide intersection capacity and need to be taken extra care of in the simulation network.

5. Lower traffic demand

Given a low demand the calibrated link flows can still be lower than the measured ones after calibration with the demand adjustment. It is suggested to execute the calibration with the results of the previous calibration in order to adjust the overall demand again. Such a way needs to be used with caution, since underestimated link flows may be possibly due to inaccurate signal timings, intersection configuration, the allowed travel speeds and driving behaviours in the network. In this situation, increasing demand will only result in more traffic jams.

6. Flow data collection

   Flow data is used as restraints for adjusting the respective amount of routes (trips) in the calibration. The more the data collection cover the major used routes, the better the calibration quality and the more representative the calibrated results will be. Thus, traffic flows at major corridors and intersections should be collected according to the given analysis period.

13.6 Conclusions

It is known that microscopic traffic simulation needs much finer and precise data than macroscopic traffic simulation. The data quality has a great influence on the calibration results. Not like a synthetic network, the data for a real network is often with limited quality and some data is not easily to obtain, e.g. signal timing plans. Moreover, different parameters with regard to car-following, driving and route choice behaviours also need to be considered and, if necessary, adjusted to insure the simulation quality in SUMO. All these issues make the calibration with microscopic simulation much more challenging especially when dealing with a large network.

According to the results, the SUMO-Cadyts calibration with a limited data quality can still improve the simulation quality, especially that the spectrum of the errors is greatly reduced to a narrow range. These results are however not promising yet, especially when the GEH-values are considered. It is mainly since the input data quality greatly influences the road and intersection capacities which affect not only the number of the vehicles passing the measured location at each time interval, but also the travellers' route choices and their route choice sets. More accurate signal timing plans and speed limitations can help to improve the results. Moreover, the simulation of driving behaviours in road sections, between lanes and at/within intersections can affect road and intersection capacities as well. Accordingly, two current adjustable parameters in SUMO, namely the gap acceptance and the teleporting time, are adjusted to reflect local people's driving behaviours during peak hour periods. Further investigation and modelling on local people's driving behaviours are still needed for better simulating driver behaviours in China and help to deliver representative simulation results.

Besides the insufficient data and data quality, the short time interval for traffic flow data collection may also be the reason for the unsatisfied calibration results. Due to traffic congestion some of the simulated vehicles cannot reach the measured locations in time within a short time interval. Traffic flow measurements with a short time interval further challenge the accuracy of the calibration with microscopic traffic simulation. Calibration with limited data quality should perform better when the time interval for data collection covers both the peak and off-peak periods for mitigating the congestion effect.

In this case study, the used flow data are only on 21 links which are mostly at the intersections on the eastern side of the network for examining the influence of the insufficient data quality on the calibration and the representation of the actual traffic conditions. The measured flow data does not cover all of the main corridors currently. Therefore, the derived demand (routes) is only partially representative of the true demand. More flow data at corridors are required for obtaining a representative traffic demand.

13.7 Bibliography

[1] DRACULA: *DRACULA - Dynamic Route Assignment Combining User Learning and microsimulAtion web site*. Accessed 2017.

[2] Flötteröd, Gunnar: *Cadyts - a free calibration tool for dynamic traffic simulations*. In *Proceedings of the 2009 Swiss Transport Research Conference*, Monte Verita, Switzerland, September 2009.

[3] Flötteröd, Gunnar and Michel Bierlaire: *Improved estimation of travel demand from traffic counts by a new linearization of the network loading map*. In *Proceedings of the European Transport Conference*, Noordwijkerhout, The Netherlands, October 2009.

[4] Flötteröd, Gunnar, Michel Bierlaire and Kai Nagel: *Bayesian demand calibration for dynamic traffic simulations*. Transportation Science, 45(4):541–561, 2011.

[5] Flötteröd, Gunnar and Ronghui Liu: *Disaggregate path flow estimation in an iterated DTA microsimulation*, November 2010.

[6] Flötteröd, Yun-Pang and Gunnar Flötteröd: *Route choice calibration from multipoint vehicle stream measurements*. In *2nd International Conference on Models and Technologies for Intelligent Transportation Systems*, Leuven, Belgium, June 2011.

[7] Gawron, Christian: *An Iterative Algorithm to Determine the Dynamic User Equilibrium in a Traffic Simulation Model*. International Journal of Modern Physics C, 9(3):393–407, 1998.

[8] Gunnar Flötteröd: *Cadyts – Calibration of Dynamic Traffic Simulations web site*. Accessed 2017.

[9] Krajzewicz, Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, 5(3&4):128–138, December 2012.

[10] MATSim: *MATSim - Multi-Agent Transport Simulation web site*. Accessed 2017.

[11] SUMO: *SUMO: Simulation of Urban MObility - Wiki web site*. Accessed 2017.

[12] Transportation, Wisconsin Department of: *Microsimulation Guidelines: Model Calibration web site*. Accessed 2017.

14 SUMO and the German Handbook for the Dimensioning of Highways (HBS)

Jakob Erdmann, Johannes Rummel, and Peter Wagner; German Aerospace Center
{Jakob.Erdmann, Johannes.Rummel, Peter.Wagner}@DLR.de

14.1 Abstract

This work presents a comparison of the micro-simulation tool SUMO with one scenario taken from a the German version of the Highway Capacity Manual (HCM) (1), the Handbook for the Dimensioning of Highways (HBS) (2). The HBS defines fundamental diagrams for a number of freeway scenarios which are used in the planning process of Germany's roads. Although the fundamental diagrams in this manual are a kind of average description, it is interesting to know how well they can be approximated by SUMO.

Keywords: Traffic simulation; calibration; traffic planning handbooks; fundamental diagram

14.2 Introduction

Often, microscopic traffic simulation software is not calibrated carefully, with SUMO (3) being no exception. Regarding calibration (and validation) of microscopic traffic simulation in general, however, a lot of work has been done already (5), so on a conceptual basis little has to be added here – it simply has to be done.

In following (7), calibration is defined as the process of estimating a good set of parameters by comparing the simulation output with data to reproduce and minimizing the difference between model and data. The parameters are often determined manually, however, automated approaches can be used as well, see (4) for an example to this. The validation, then, fixes the parameters and checks for the difference between the simulation output and another data-set than the one used for calibration. This is an assessment of the model's ability to generalized to conditions that have not been used to estimate the parameters.

Clearly, the input data matter. However, by these data need not to be real ones. At least in Germany, there is the "Handbuch für die Bemessung von Straßenverkehrsanlagen" (Handbook for the Dimensioning of Highways) (2), which has for many different situations on roads an expected behavior that is either defined by certain functions, or even by look-up tables. For freeways this handbook, as well as its US counterpart, the HCM (1), defines fundamental diagrams in the form of speed v-flow q-curves $v(q)$ for different number of lanes, grades, truck rate, and some other influence factors. These fundamental diagrams can be interpreted as some kind of average over many similar situations, and they may be used as a test for a traffic simulation software.

This article will stick to the example of a three lane freeway, see Fig. 14.1 for the relevant presentation of the fundamental diagram in the HBS. The description in the HBS assumes the curves to represent 1 hour averages. In this case, it is specified for freeways near larger cities (conurbations), with a grade smaller than 2% and for four different truck rates from 0 to 30%. Only the free branch of the fundamental diagram is represented in the HBS.

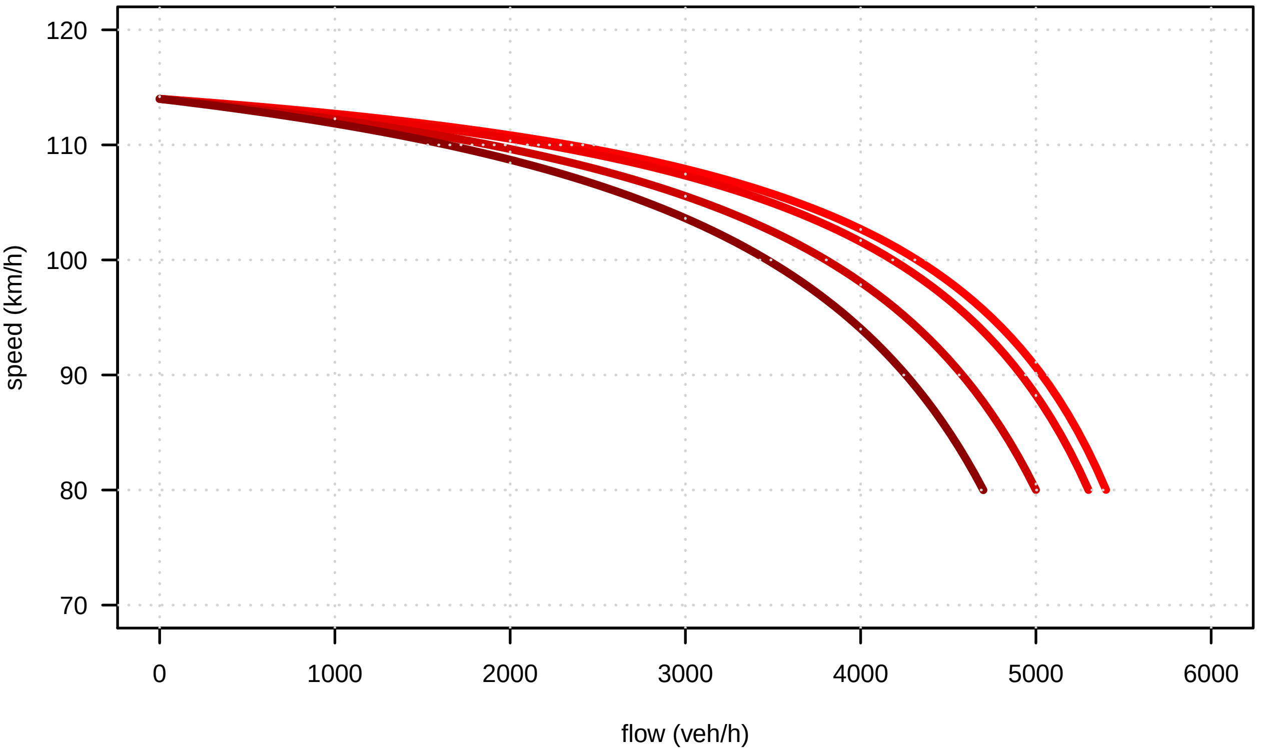


Figure 14.1: The HBS version of the fundamental diagram of a three lane German freeway for different truck rates and with speed limit 100 km/h. The curves follow a function $v(q) = V_0/(1 + V_0/(L_0(C_0 - q))$ with appropriate choices of the parameters, e.g. $(V_0, L_0, C_0) = (121.2\,km/h,\ 0.312\,km, 6155\,veh/h)$ for four different truck rates $p_T = (\leq 0.05, 0.1, 0.2, 0.3)$; the rightmost curve is for the smallest truck rate. Note, that the passenger car speed is presented, not the average over all vehicles.

This work presents results achieved with SUMO by trying to reproduce some of the curves from 2015 version of the HBS. It uses SUMO's default model, and except where stated otherwise, it also uses SUMO's default parameters. Note also, that despite the fact that most of the models implemented in SUMO allow the computation of a fundamental diagram for idealized conditions (one lane road, no noise, no heterogeneity) of the fundamental diagram, for realistic scenarios it is very difficult or even impossible to compute a fundamental diagram. Therefore, simulation has to be used to compute the curves of the fundamental diagrams to compare with the HBS.

14.3 Three lane freeway

SUMO is set-up to reproduce Fig. 14.1 as follows: the study area consist of a stretch (10 km) of a three-lane road which is divided into 1 km pieces to ease the simulation data analysis. It is bounded at the upstream side by a 2 km inflow edge, and on the downstream side by a 500 m long edge which can be equipped with a Variable Speed Sign (VSS). This VSS is not used in this study, it is mentioned here in case also the jammed branch of the fundamental diagram is explored. Then, it can be used to force the study area into a jammed state – otherwise, jams are rarely or never created on such a stretch of open road. The upstream edge is used to let the (artificial) inflow pattern dissolve; this results from the fact that the insertion mechanism of SUMO does not know about the actual headway distribution generated by the simulation and therefore impresses a certain pattern (in this case a binomial distribution) which may or may not have any similarities with the one realized in the simulation. So, it is assumed that 2 km are enough to let the simulation relax against its natural flow state and not the one imposed by the insertion. This may not cause any difference, but it can also produce difficult to find deviations.

The demand enters the study area in the form of a slowly and stepwise increasing function. Each step is hold in most scenarios for 1 h to match HBS's 1 h time frame. However, shorter times have also been used and show little difference except that with shorter aggregation times, the resulting simulation data become more noisy. After reaching maximum flow, the simulation is stopped and data are extracted for analysis.

The simulation data can be extracted from SUMO by many different mechanisms. The most straightforward way is to instrument the study area with loop detectors. Here, SUMO's `edgeData` mechanism is used, which gives a quite complete summary of the behavior that results:

```
<additional>
    <edgeData id="ed" freq="3600" file="edgeStats.xml" />
    <laneData id="ld" freq="3600" file="laneStats.xml" />
</additional>
```

This is included into the simulation as an additional file, and in this case it aggregates its output into time bins of 3600 seconds and writes all the data into the `laneStats` and `edgeStats` files.

In a first step, SUMO is run with the default values of version 0.29, and without trucks. This yields a flat fundamental diagram, both for SUMO's microscopic, as well as for the mesoscopic variant, see Figure 14.2. Note also, that the microscopic and the mesoscopic results are very close to each other.

It can be seen, that the default parameters produce a flow that is too large, and that there is no bending in the fundamental diagram. The maximum flow can be addressed mostly by the parameters `sigma` and `tau`, while the bending is a consequence of the heterogeneity of the speeds of the vehicles. The latter can be addressed either by introducing different vehicle types (trucks are needed anyway) or by using mechanisms built into SUMO. Most notably, these are the two parameters `speedFactor` and `speedDev` that will be used here. The `speedFactor` determines, how much faster (or slower) than the current speed limit a certain vehicle will go, the number indicating the vehicle conforms

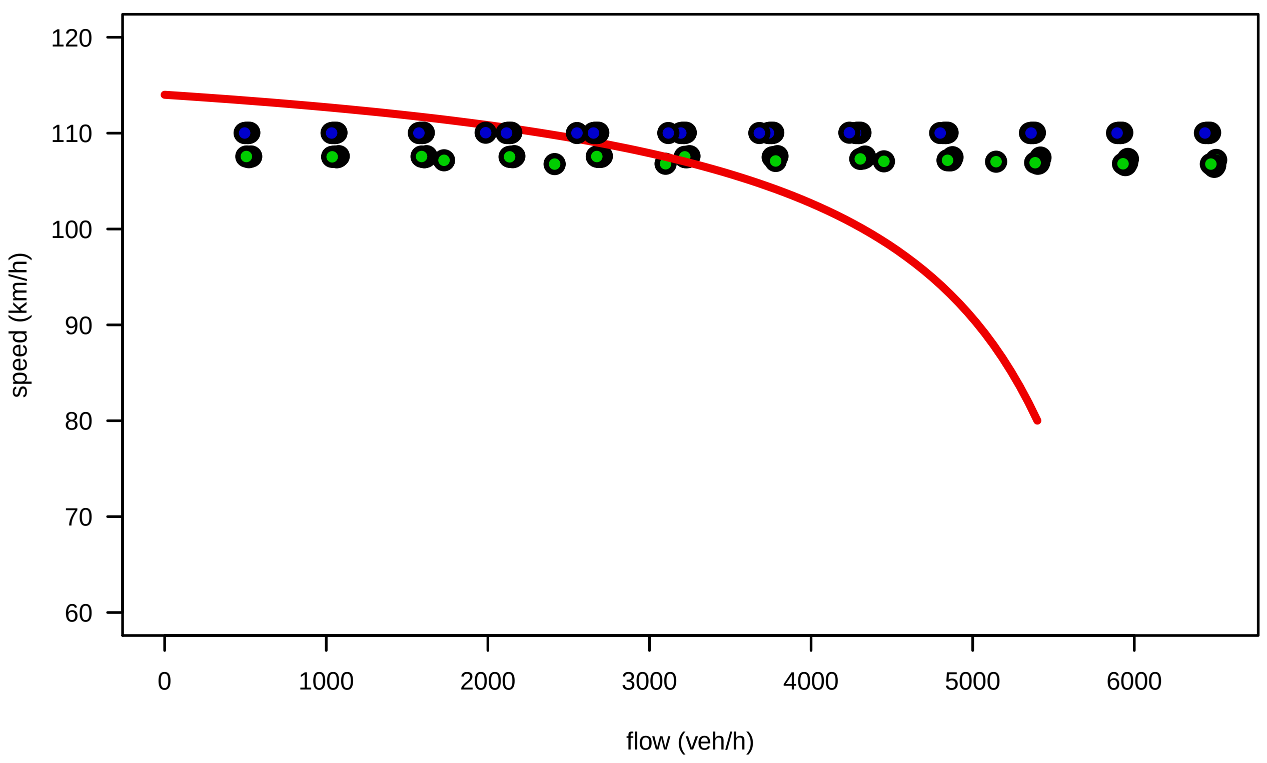


Figure 14.2: The fundamental diagram obtained with SUMO's default model with the default parameters for a three-lane freeway with speed-limit 100 km/h and zero truck rate. The blue points are from the mesoscopic model, while the green ones are from the microscopic model. The red curve is the HBS curve for zero truck rate.

exactly with the speed-limit. The parameter speedDev defines the width of the distribution of the speedFactor variable for the individual vehicles. I.e. the speedFactor of each vehicle is determined upon creation of the vehicle by drawing from a normal distribution with a mean value determined by speedFactor and a standard deviation determined by speedDev. Setting speedDev = 0.15, the bending becomes visible and the maximum flow reduces somewhat, see Figure 14.3.
Of course, it may be asked what are realistic numbers for these two parameters. An idea can be found from single vehicle data from German freeways. By analyzing the data from the German freeway A92 between Munich Airport and the freeway intersection Neufahrn (where a speed-limit of 100 km/h is in place), the two distributions displayed in Figure 14.4 can be found. From these distributions, the speedFactor, speedDev parameters can be estimated, they are just the mean and the standard deviations of the two distributions and yield speedFactor = (1.01,0.88), and speedDev = (0.18,0.09) (passenger cars and trucks), respectively. Furthermore, the length distribution can be used to estimate the lengths of the simulated vehicles by computing the mean of the data. This yields $\ell_P = 4.2$ m, $\ell_T = 11.6$ m for passenger cars and trucks, respectively. Note, however, that the truck-length distribution is clearly bimodal, so using just one mean-value for the truck-lengths may not be the best way to proceed, but by far the simplest one.

14.4 Fitting

Even this simple model has a large number of parameters. They are specified as follows:

- Two sets of vehicle-specific parameters (passenger cars / trucks – here, the SUMO names are used): these are maxSpeed, accel, decel, minGap, tau, sigma, length

- Two sets of speed-specific parameters speedFactor, speedDev

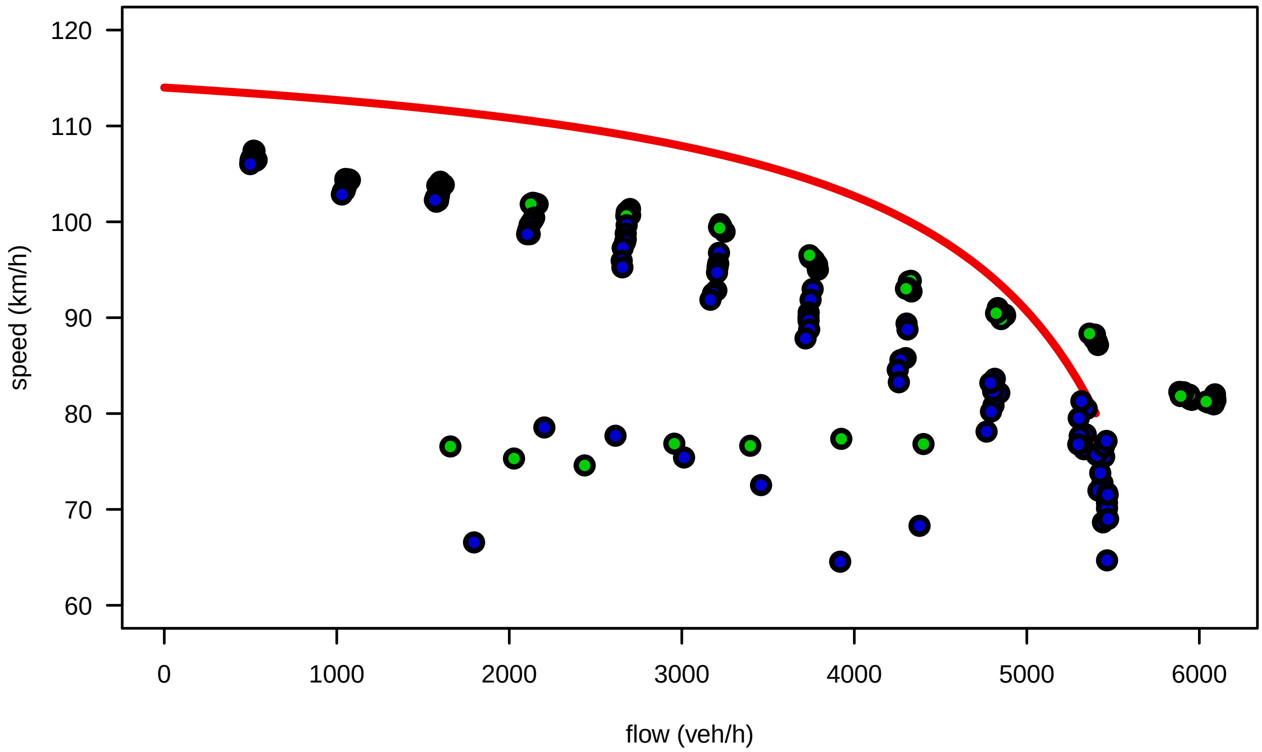


Figure 14.3: The fundamental diagram obtained with SUMO's default model for a three-lane freeway and zero truck rate, but now with `speedDev = 0.15`.

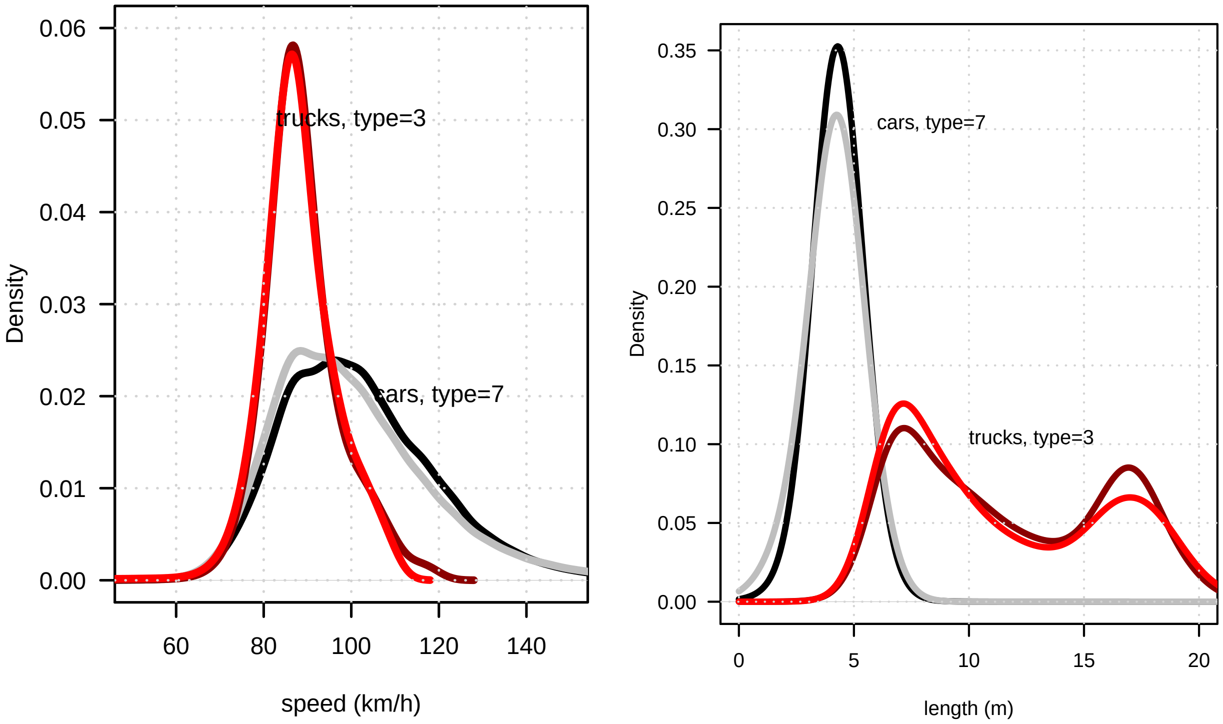


Figure 14.4: The speed and length distribution, for data taken from all lanes at all cross-sections. Only passenger cars and trucks have been picked from the data; the similar looking curves in each plot are for the two directions.

- Two sets of lane-changing parameters for the default model `lcCooperative`, `lcSpeedGain`, `lcKeepRight`.
- Another parameter is also the time-step size of the simulation, named Δt here, but in this study it is left at its default value of 1 second.

What is also not straightforward is how to define the objective function to be minimized. Intuitively, it should be something that measures how close the simulated curve is to the HBS curve. However, there is one step in between: trying to insert a certain flow q_i into SUMO does not necessarily result in observing this flow. This is due to a mismatch between actual demand and the maximum flow that can be reached by SUMO (named capacity). When this demand approaches SUMO's capacity, vehicles are either queued at the entrance to the study area, or they can be thrown away after a certain time without getting into the system. Here, the latter approach has been chosen, after waiting for 10 seconds in vain, a vehicle is eliminated from this input queue. This has the advantage, that there is no overhanging queue of vehicles that distort the demand, and it implements a simple type of self-limitation of the flow. So, the inflow versus demand curve is a line with slope one up to the point, where capacity is reached – then inflow stays constant.

However, in the definition of the objective function this may or may not be included. Assume an inflow of 5000 veh/h, which leads to a certain speed according to the HBS curve. Assume further, that the simulation can accommodate only 4500 veh/h and for this flow yields the same speed as the HBS-curve with 5000 veh/h. Then, by comparing the speeds, this accounts for no error, which is clearly wrong. It is not that easy to define the distance between the two curves, and here a certain compromise has been used that compares the speed at the traffic flow measured in the simulation.

$$e^2 = \frac{1}{N}\sum_{i=1}^{N}(v_i(q_i) - v_{\textbf{HBS}}(q_i))^2. \tag{14.1}$$

Fitting is done exemplary for one example only because it is a time-consuming process: fixing the truck rate at $p_T = 0.1$ and assuming the HBS parameters for a three-lane freeway close to conurbations with no grades, a nonlinear optimization has been set-up that in each step runs a full simulation to determine the simulated $v(q)$ curve which is then compared to the HBS curve: the simulation returns for each input demand 10 different flow and speed values (q_i, v_i) from the 10 different 1 km segments, for which the difference to the HBS curve can be computed as in eq. (14.1). This is fed into a nonlinear optimization algorithm named COBYLA (6), which is part of the scipy python library. Note, that COBYLA is an optimization tool that works best for a deterministic objective function, it works in this example since the fluctuations in the simulation output are not very large.

Such an optimization typically needs some 90 iterations to converge to a solution, as is demonstrated in Figure 14.5. Still, the simulation does not reproduce the HBS curve exactly, but the results are satisfactorily.

14.5 Conclusions

This paper compared the SUMO simulation to the HBS. Note, that more work has to be done here. First of all, it will be interesting to look at different models, and to look at a comparison between SUMO and real data that uses more quantitative measures than "looks-roughly-the-same". There are differences between the HBS and real data, too. And of course, a more microscopic comparison has to be done and is definitely on our future research program. While doing this work, a lot of small recommendations for SUMO results, which will be found in the next version of the program. So, this work will add to making SUMO more realistic, and to better understand which parameters of it are important, and which ones are not. Obviously, this depends on the objective function, too, and inn addition, on the scenario the analyst wants to perform.

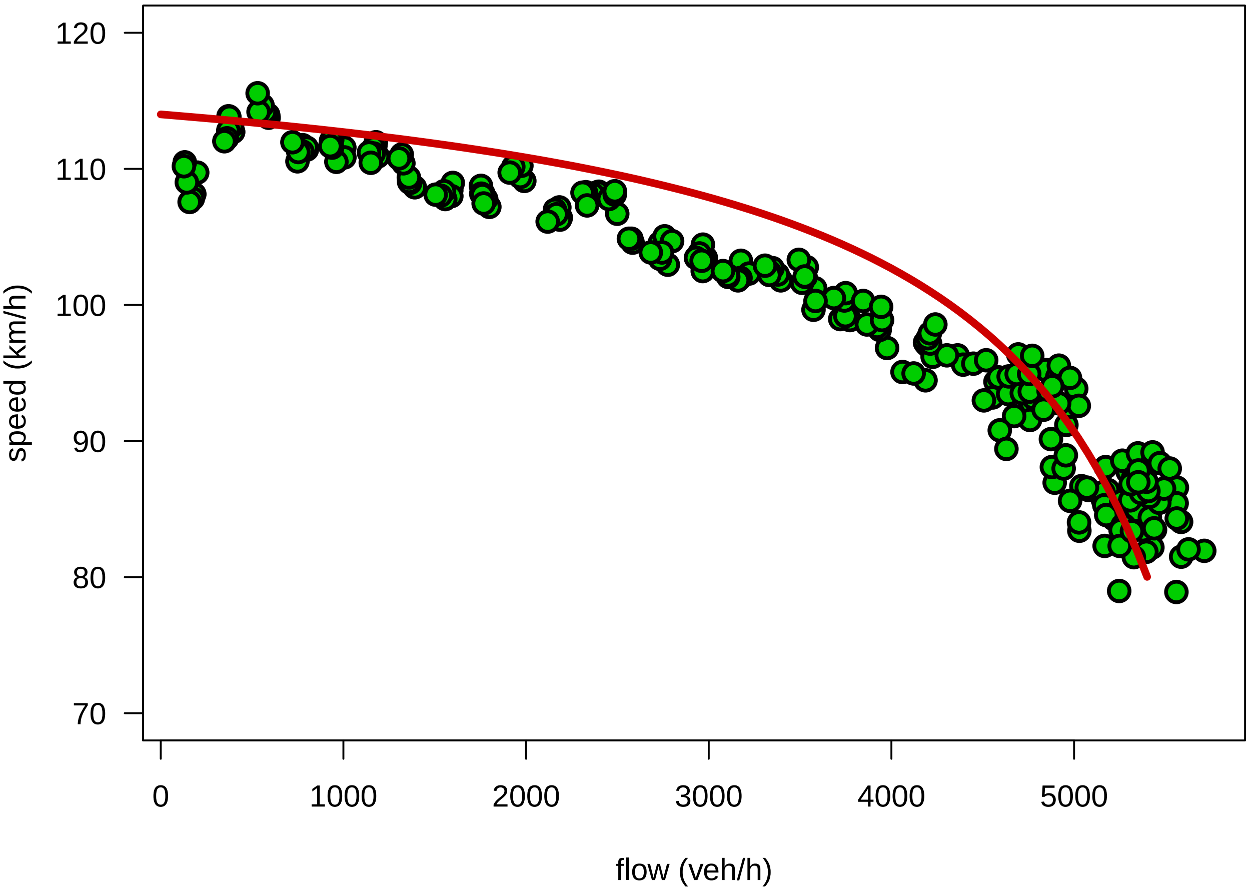


Figure 14.5: A comparison of SUMO with the HBS curve for the truck rate $p_T = 0.1$. The optimization was allowed to change the all parameters of SUMO (except the time step size).

14.6 Bibliography

[1] *Highway Capacity Manual 2010*, volume Volumes 1 - 4. Transportation Research Board., 2010.

[2] *HBS – Handbuch für die Bemessung von Straßenverkehrsanlagen*. Number ISBN: 978-3-86446-103-3. FGSV Verlag – Der Verlag der Forschungsgesellschaft für Straßen- und Verkehrswesen, 2015.

[3] Behrisch, Michael, Laura Bieker, Jakob Erdmann and Daniel Krajzewicz: *SUMO – Simulation of Urban MObility: An Overview*. In *Proceedings SIMUL 2011, The Third Conference on Advances in System Simulation*, pages 63 – 68, 2011.

[4] Brockfeld, E., R. D. Kühne, A. Skabardonis and P. Wagner: *Towards a benchmarking of Microscopic Traffic Flow Models*. Transportation Research Records, 1852:124–129, 2003.

[5] Daamen, Winnie, Christine Buisson and Serge P. Hoogendoorn (editors): *Traffic Simulation and Data – Validation Methods and Applications*. Taylor and Francis, CRC Press, 2014.

[6] Powell, M.J.D.: *Advances in Optimization and Numerical Analysis*, chapter A direct search optimization method that models the objective and constraint method by linear interpolation. Kluwer Academic (Dordrecht), 1994.

[7] Trucano, T.G., L.P. Swiler, T. Igusa, W.L. Oberkampf and M. Pilch: *Calibration, validation, and sensitivity analysis: What's what*. Reliability Engineering & System Safety, 91(10 – 11):1331 – 1357, 2006.

15 Replication of the HBS Autobahn with SUMO

Johannes Rummel; Technische Universität Berlin
Johannes.Rummel@tu-berlin.de

15.1 Abstract

This work seeks to prove the following hypothesis: SUMO can reproduce the fundamental diagrams put forward by the German handbook HBS to a certain degree. This is demonstrated by the particular example of what is known in Germany as the E1 ramp, which is a one-lane ramp connected to a two-lane freeway. A comparison of the simulation by SUMO with the HBS will show that SUMO can qualitatively reproduce some realistic scenarios, but not the full range of the fundamental diagram.

Keywords: HBS, on-ramp, capacity, autobahn, freeway, highway

15.2 Introduction

The German HBS (1) is a handbook and determined to plan inter alia highways. To this end it makes predictions about capacities of different highway elements. Traffic simulation is not well established in the planning process, as the skepticism towards the applicability of simulations is high. Selecting one highway element, the on-ramp of type E1, SUMO (2) should be adjusted to reproduce the predictions made by the HBS.

The on-ramp of type E1 consists of a main road with two lanes and a ramp with one lane. Obviously, the flow of the on-ramp is limited by the maximum flow of the two inflows. But also the capacity of the main road behind the ramp further limits the capacity of the entire on-ramp. As the ratio of the inflows can vary, the HBS describes the capacity by means of an elliptical function. This function is derived from four theoretical boundaries - the capacity of the main road before the ramp (C_{HO}), the capacity of the ramp (C_E) and the capacity of the main road behind the ramp (C_{HU}). The HBS adds a fourth boundary resulting from the merging of the arriving vehicles from the ramp to the main road (C_M). The flow of the merging area q_M is the sum of the ramp's flow q_E and the flow of the main road's right lane $q_{HO,r}$:

$$q_M = q_E + q_{HO,r}. \tag{15.1}$$

The HBS describes this boundary by the following function:

$$q_E = C_E \cdot \sqrt[a]{1 - \left(\frac{q_{HO}}{C_{HU}}\right)^a}, \tag{15.2}$$

with q_{HO} as the entire flow of the main road before the ramp and a, chosen such that the elliptical function touches the boundary C_M. This function is represented by the dashed line in Figure 15.1, which also shows the above-mentioned boundaries.

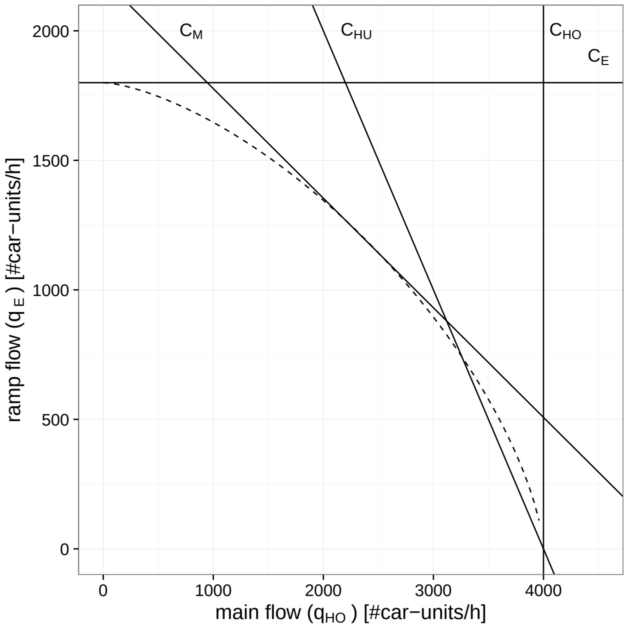


Figure 15.1: Construction of the capacity curve by the HBS for on-ramp E1

Assuming that the HBS describes the reality, a microscopic simulation such as SUMO should reproduce these characteristics. Figure 15.2 displays an enlarged detail of the simulation, showing the significant part of the on-ramp.

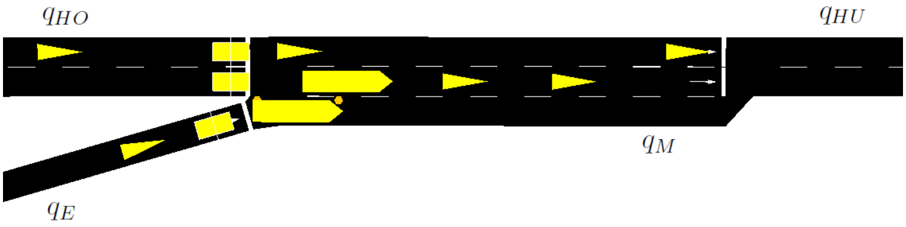


Figure 15.2: Screen shot of the simulation for on-ramp E1

15.3 Methods

To determine the capacity of the on-ramp with SUMO, the simulation runs with slowly increasing flows. The capacity is reached briefly before the main flow breaks down. As the HBS makes no statement about breakdowns, it is not necessary to discuss the definition of breakdowns here. Important is that the capacity is defined as the maximum flow - in this scenario depending on the two inflows. With the current inflows, one gets one point of the capacity function predicted by the HBS.

The simulation measures the flows and speeds with induction loops aggregated for 5 minutes each. Each setup is repeated 10 times with a different set of random numbers. The HBS measures the number of vehicles in car-units, which adds a factor of 2 to the number of heavy traffic vehicles.

Since the ratio of heavy traffic is chosen to be 10% in all simulations, the measured number of vehicles corresponds to the same number of car-units, but with a factor of 1.1 .

15.3.1 Implementation

Inflows

For the examination, it is necessary to choose some ratios of the two inflows leading to six realistic scenarios. These scenarios, named G1-G6, are defined with the following ratios q_E/q_{HO}:

G1 : 2/3 x 100%

G2 : 2/3 x 70%

G3 : 2/3 x 45%

G4 : 2/3 x 30%

G5 : 2/3 x 20%

G6 : 2/3 x 10%

The measured ratios of the two inflows fluctuate, as there are an initial 4 km of main road and 500 m of ramp road to give the vehicles time to show a natural behavior. The scenarios always have an inflow ratio for heavy traffic of 10%.

To introduce as many vehicles as possible to the main road, some depart properties have to be observed. Property "departPos" is always set to "last." This introduces the vehicles with a minimum gap to the leading vehicle or at the end of the appropriate lane, if there is no other vehicle on this edge. The second important property to reach a high inflow is "departSpeed." Setting it to the critical speed determined previously, the vehicles already have the right speed to proceed with minimum headway.

Decreasing the time step of the simulation also increases the maximum flow of the main road. Since the other two properties already lead to a maximum flow of 4000 veh/h on the main road, which is the predicted capacity of the HBS, it is not necessary to change the default time step of 1 second.

Vehicle Types

According to the HBS scenario, at least two vehicle types have to be defined - a car type and a truck type. Only few properties of the vehicle types have a significant influence to the capacity of the on-ramp. One of them is the model parameter sigma. Using the default value of 0.5 leads to capacities around the predicted ones of the HBS, but this is insufficient, since the maximum inflow for the main road is 4800 veh/h instead of 4000 veh/h. That means the capacity of the main road is too high. A sigma of 1.0 makes the capacities describe a straight line above the HBS curve and decreases the maximum inflow to 3400 veh/h. Using values of around 0.7 for the car type and 0.9 for the truck type seems to be more reasonable. The maximum inflow of the main road is 4000 veh/h and the capacities follow a straight line. As the other properties have little impact to the capacity, they are set to realistic values shown in Table 15.1.

Road Network

It has already been mentioned that an on-ramp of type E1 is used, this is consisting of a main road with two lanes and a ramp with one lane. The main road has at least an initial 4 km and the ramp 500 m. The preceding edges are 1 km and 500 m respectively and are designed for the insertion of the vehicles. The merging area is 250 m followed by a 500 m road with two lanes.

As is standardly the case in Germany, trucks are not allowed to go faster than 80 km/h on the main road. The ramp is limited to 80 km/h for all vehicles.

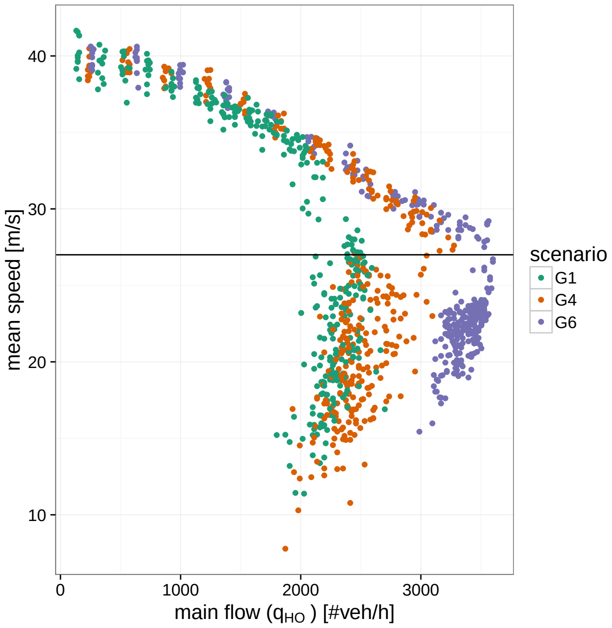


Figure 15.3: Fundamental diagram for three of the measured scenarios

Attribute	Value	Attribute	Value
id	car	id	truck
maxSpeed	50	maxSpeed	25
accel	3.5	accel	1.3
decel	4.5	decel	3.5
length	5	length	10
sigma	0.72	sigma	0.9
tau	1.2	tau	1.0
minGap	2.8	minGap	3.0
speedFactor	1.19	speedFactor	1.12
speedDev	0.1	speedDev	0.02
departLane	random	departLane	0
lcSpeedGain	1.0	lcSpeedGain	0.5

Table 15.1: Used vehicle types for the simulation with SUMO

15.3.2 Determining the Capacity from the Simulation

Figure 15.3 shows the measured fundamental diagram for three of the six simulated scenarios. The fundamental diagram consists of the mean speed depending on the entire flow, both referring to the main road before the ramp. Starting with free flow, the usual course of the fundamental diagram is a decreasing speed with an increasing flow; as soon as the capacity is reached, the flow decreases. As the state of maximum flow is not stable, there can be few measurements for that part of the state space and a determination of the capacity is not precise. To get around this problem, it is possible to choose a critical speed leading to maximum flows. Now the flows of the time intervals are used as the capacity where the speed falls below this critical speed.

15.4 Results

15.4.1 Evolution of the Simulated Flows

The mean speed evolution of the main road before the ramp has a similar course for all scenarios (as shown for example in scenario G4 in Figure 15.4). It starts with a value of around 40 m/s and decreases over time. At a certain point, the mean speed breaks down and fluctuates between the critical speed and 0 m/s. By construction, the mean flow of the main road before the ramp increases over time until it reaches a certain, mostly maximal value (Figure 15.5). Then it remaons constant or decreases marginally over the remaining time. Mostly, the time interval, where the speed falls below the critical speed, is also the interval with the highest flow. For the scenarios where the highest flow cannot exactly be extracted, the critical speed helps to determine a flow.

Smoothed Curves by Averaging per Scenario

Since the random number seeds of the individual scenarios show the same course, it is possible to average the speed and flow for each time interval to receive one smooth curve per scenario. The averaged values are plotted in Figure 15.6 and Figure 15.7. As expected, the speeds decrease and the flows increase with different slopes for the different scenarios.

15.4.2 Measured Capacities

Considering the extracted capacities in Figure 15.8, it is not clear whether they follow the HBS curve. They could also follow the capacity of the main road behind the ramp (boundary C_{HU}). To investigate the upper left part of the capacity, three additional "unrealistic" scenarios G0, G-1 and G-2 with higher ramp flows are executed. Although they mostly show no breakdown within

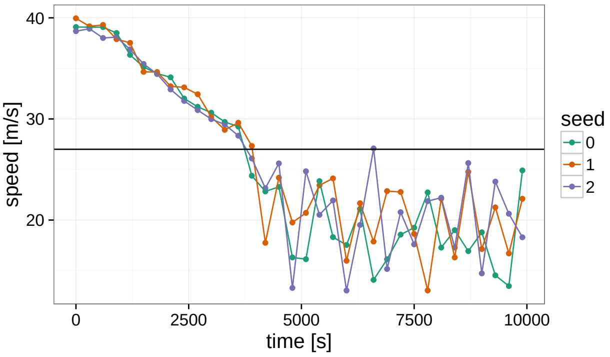


Figure 15.4: Speed evolution for scenario G4 (main road)

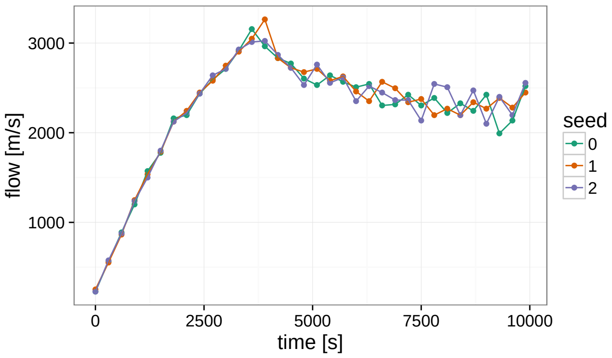


Figure 15.5: Flow evolution for scenario G4 (main road)

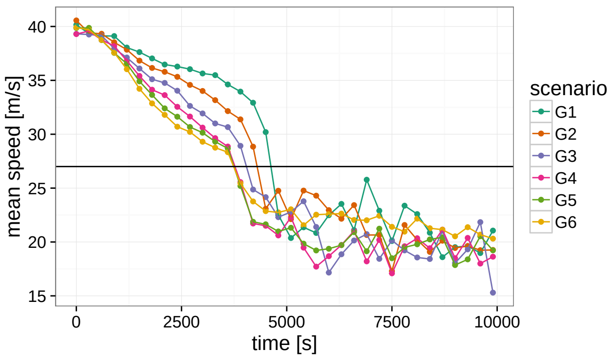


Figure 15.6: Speed evolution for all averaged scenarios (main road)

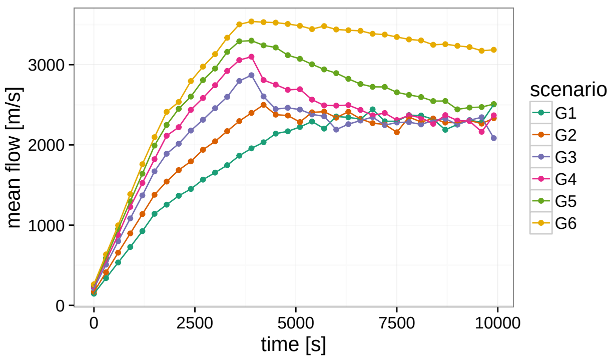


Figure 15.7: Flow evolution for all averaged scenarios (main road)

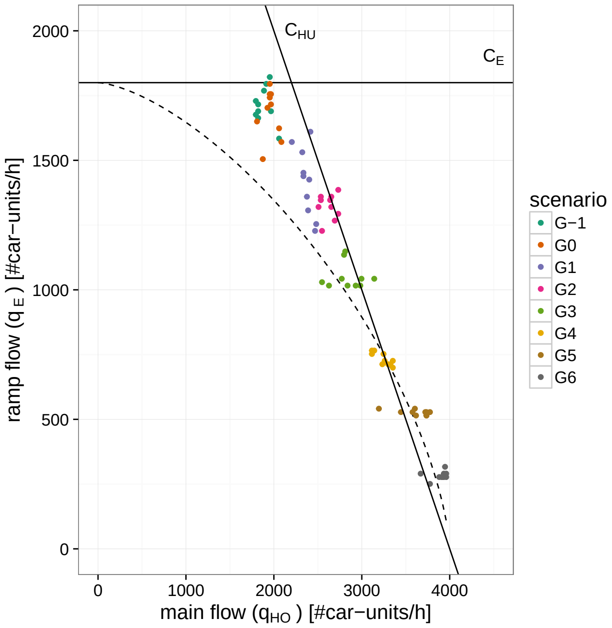


Figure 15.8: Capacities for all scenarios and seeds (dots), prediction by HBS (dashed line), capacity of the ramp C_E and the main road behind the ramp C_{HU}

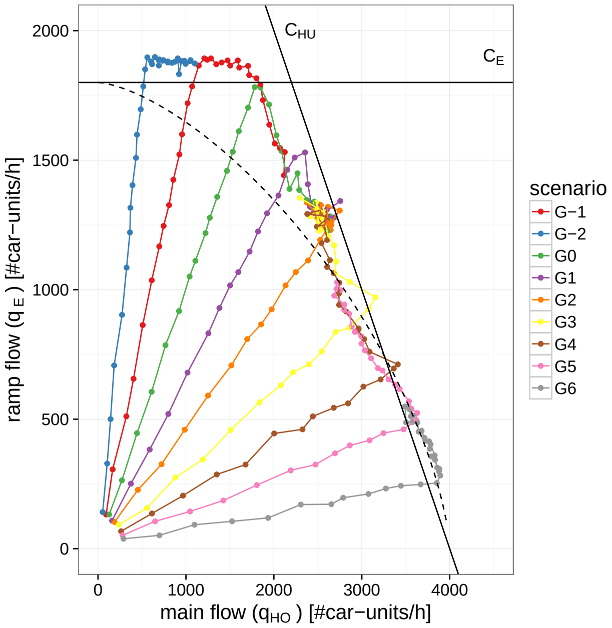


Figure 15.9: Flow courses for all scenarios (dots with lines), prediction by HBS (dashed line), capacity of the ramp C_E and the main road behind the ramp C_{HU}

the considered time range, the ramp inflow has a maximum value of around 1700 veh/h ($\approx$ 1850 car-units/h), which corresponds to the limit C_E given by the HBS.

Figure 15.9 shows all averaged flow evolutions over all random number seeds for each scenario. The possible flows seem to be limited by two straight boundaries - a maximum value for the flow of the ramp and a boundary similar to the capacity of the main road behind the ramp.

15.5 Discussion

SUMO can reproduce the capacity boundaries C_E and C_{HU}, as the capacities in Figure 15.8 and the larger values for the flow in Figure 15.9 show. The courses of the flows for scenario G-1 and G0 indicate that the simulation does not noticeably observe the limitation by the merging area C_M. This could be an issue of the way the vehicles interact while merging. A change of the lane change parameter "lcCooperative," which is responsible for the willingness to perform cooperative lane changing, does not achieve a better behavior. It alters the position of the maximum flows, but they still describe a straight line with almost the same slope. As well, the calibration of the other considered parameters does not change the qualitative behavior of the flows. There is always an upper boundary for the flow of the ramp and a boundary following a line such as C_{HU} with the same slope.

Overall, SUMO can reproduce the course of the capacity in a realistic range. Fine-tuning is necessary to shift the capacities of the ramp and the main road to the capacities given by the HBS. That however does not mean SUMO behaves realistically. As the capacities with higher ramp flow shows, the merging should be revised. As well, the influence of the other parameters needs to be clarified, since these parameters do not affect the capacity, but, rather, the microscopic behavior of the vehicles.

As a next step, real life data should be used to calibrate a SUMO simulation. This will show whether the found parameters will also reproduce other scenarios and could help to understand the problems of merging in this context.

Additionally, further freeway elements such as off-ramps and weaving sections should be investigated to get an overall overview of the freeway simulation.

15.6 Bibliography

[1] *HBS – Handbuch für die Bemessung von Straßenverkehrsanlagen*. Number ISBN: 978-3-86446-103-3. FGSV Verlag – Der Verlag der Forschungsgesellschaft für Straßen- und Verkehrswesen, 2015.

[2] Behrisch, Michael, Laura Bieker, Jakob Erdmann and Daniel Krajzewicz: *SUMO – Simulation of Urban MObility: An Overview*. In *Proceedings SIMUL 2011, The Third Conference on Advances in System Simulation*, pages 63 – 68, 2011.

16 LiSuM: Design and Development of a Middleware to couple Virtual LISA+ TLS Controller and SUMO Simulation

Maximiliano Bottazzi, Louis C. Touko Tcheumadjeu, Jan Trumpold, Jakob Erdmann, Robert Oertel; German Aerospace Center (DLR)
{maximiliano.bottazzi, louis.toukotcheumadjeu, jan.trumpold, jakob.erdmann, robert.oertel}@DLR.de

16.1 Abstract

Traffic signal control logic programs are analyzed and tested in traffic flow simulators before being put into operation in real traffic road intersections. LISA+ is a proprietary software tool used to plan and evaluate complex intersections and the control logic programs created with it can be directly uploaded to real controllers or tested with VISSIM, a proprietary microscopic 3D traffic simulator. On the other hand, Simulation of Urban Mobility (SUMO) is a free and open traffic simulation tool suite that facilitates simulation of traffic and the evaluation of infrastructure changes as well as policy changes before implementing them on the road. As mentioned above, LISA+ control logic programs can be currently tested only with proprietary software and a free and open alternative, like SUMO, is desirable. To close this gap this paper presents the design and development of a middleware called LiSuM that enables the communication and interaction between the virtual LISA+ traffic light signal (TLS) controller and SUMO. Furthermore LiSuM provides also a friendly graphical user interface (GUI), which allows managing all aspects related to the interaction between LISA+ and SUMO. The main features of the LiSuM Middleware as well as the provided interfaces are also described. The usage of LiSuM is demonstrated for a single intersection simulated with SUMO and controlled by LISA+.

Keywords: SUMO, LISA+, traffic simulation, virtual TLS controller, detector data, Middleware

16.2 Motivation

So far, most traffic-actuated controls for traffic lights can only be simulated with self-written scripts in SUMO (2) (5). The import of proprietary formats of complex signal programs such as those used with real traffic light controllers was not possible. To allow testing of LISA+ control algorithms within a SUMO simulation, the realization of a middleware that interfaces these programs is required. Such communication interfaces (4) already exist between the VISSIM simulation software and the virtual LISA+ controller (see Figure 16.1). The aim of these interfaces is to exchange the traffic demand data obtained from loop detector from the simulation (e.g. SUMO or VISSIM) and the switch commands for the signal heads (e.g. red, yellow, green) obtained from the virtual TLS controller (e.g. LISA+ controller) so that the simulation can transfer the obtained signal heads to the traffic light operating into the simulation. How the simulated controller processes the data and reacts depends on the traffic-actuated control logic planed with LISA+ OTMC (Open Method of Traffic Control) editor (4) (see Figure 16.1) before. Our software solution which realizes the communication interface

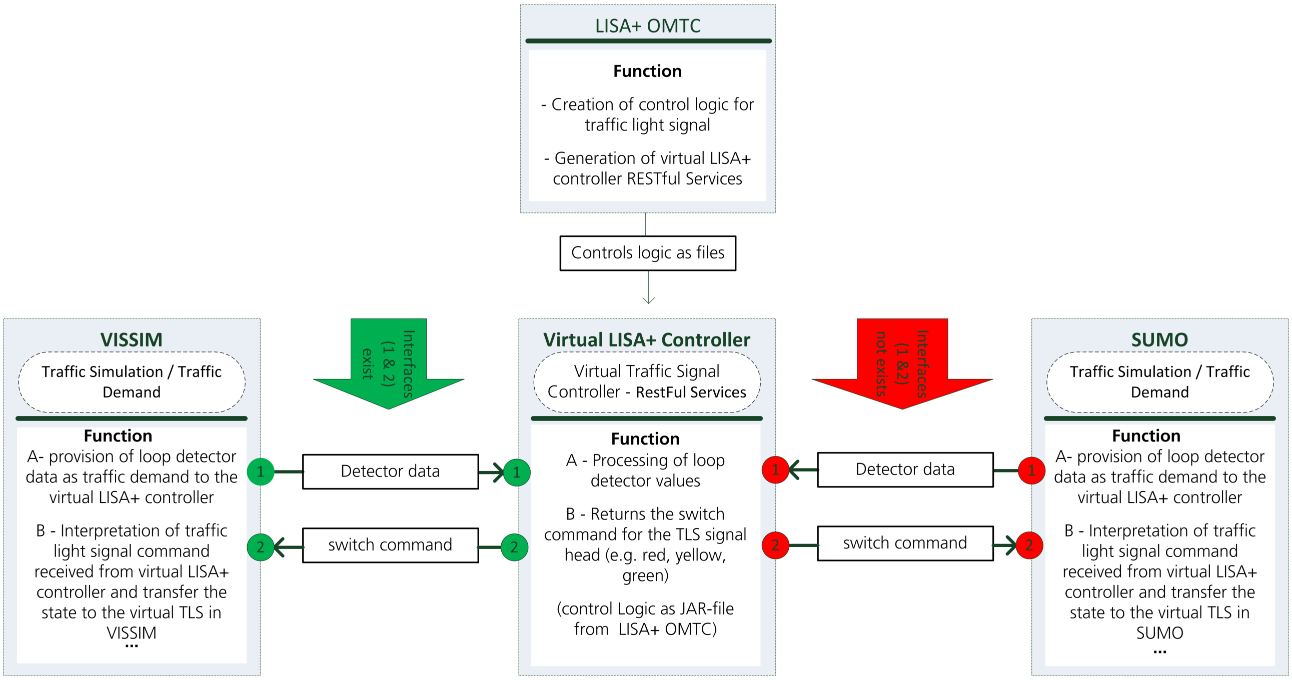

LISA+ OMTC
Function
- Creation of control logic for traffic light signal
- Generation of virtual LISA+ controller RESTful Services
Controls logic as files
VISSIM
Traffic Simulation / Traffic Demand
Function
A- provision of loop detector data as traffic demand to the virtual LISA+ controller
B - Interpretation of traffic light signal command received from virtual LISA+ controller and transfer the state to the virtual TLS in VISSIM
...
Interfaces (1 & 2) exist
Detector data
switch command
1
2
Virtual LISA+ Controller
Virtual Traffic Signal Controller - RestFul Services
Function
A - Processing of loop detector values
B - Returns the switch command for the TLS signal head (e.g. red, yellow, green)
(control Logic as JAR-file from LISA+ OMTC)
Interfaces (1 &2) not exists
SUMO
Traffic Simulation / Traffic Demand
Function
A- provision of loop detector data as traffic demand to the virtual LISA+ controller
B - Interpretation of traffic light signal command received from virtual LISA+ controller and transfer the state to the virtual TLS in SUMO
...

Figure 16.1: Overview about the communication between Virtual LISA+ TLS Controller, VISSIM and SUMO simulation [Designed by DLR]

between the virtual LISA+ TLS controller and SUMO is called LiSuM (LISA SUMO Middleware) middleware (see Figure 16.2). It provides the following functionalities:

- LiSuM is a standalone middleware that provides the SUMO interface to LISA+ controller helping to execute more complex traffic controls on the intersections than SUMO originally permits.
- SUMO communicates with the LISA+ virtual controller through LiSuM.
- LiSuM supports the RESTful service interface as well as TraCI (Traffic Control Interface) (6) interface therefore acts as interface converter between LISA+ RESTful interface and SUMO TraCI4J interface.
- LiSuM provides also friendly graphical user interface and allows the user to manage the whole interaction process (E.g. configuration, switching signal programs, start and stop) between SUMO and the virtual LISA+ controller.
- LiSuM interacts as mapping tool for example for detector, signal groups between SUMO and the virtual LISA+ controller as well as green phase, wherever naming convention in the two tools are different.

16.3 Introduction

In this section a short introduction about the following three tools LISA+, VISSIM and SUMO is given.

16.3.1 LISA+

LISA+ is a proven traffic engineering software for planning and evaluation of traffic signal lights developed, updated and commercialized by the German company Schlothauer & Wauer (4). The

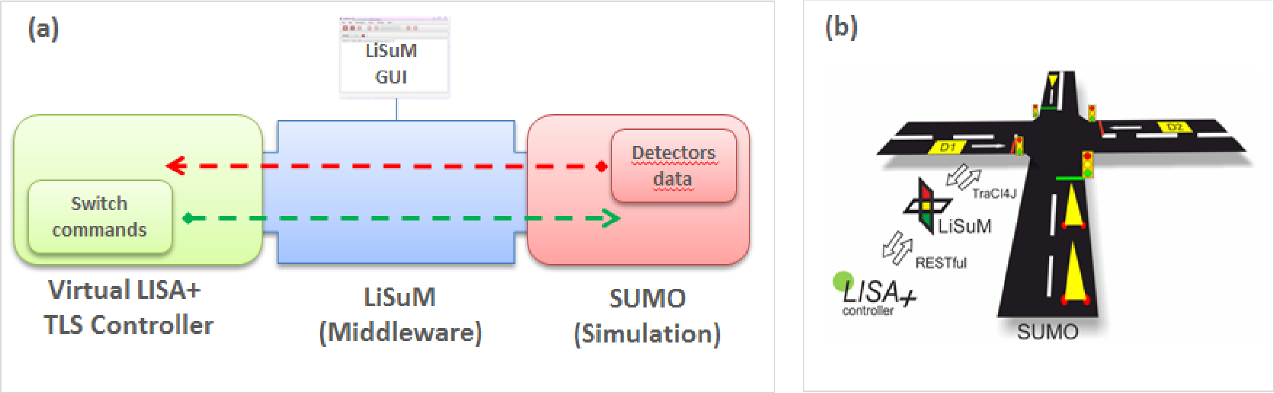


Figure 16.2: (a) SUMO communicates with the virtual LISA+ TLS controller through LiSuM middleware interface, (b) Schematic representation with RESTful interface between LISA+ and LiSuM and TraCI4J between SUMO and LiSuM

Scope of the Software is a traffic engineering workstation to create traffic signal controls for individual intersections and progressive systems, evaluate controls based on current guidelines, plan, test and simulate traffic-actuated controls and upload data to controllers directly and remotely. Target groups for LISA+ are traffic engineering companies, municipal administrations, signal manufacturers and universities. The LISA+ tool is mostly used in Europe (e.g. Germany, Austria, and Belgium) and in South America (e.g. Mexico, Brazil, and Columbia) (4). For the traffic simulation software VISSIM, Schlothauer & Wauer already provides an interface to LISA+. With this interface and LISA+ it is possible to run a traffic simulation with traffic-actuated traffic light controls built in LISA+. While the LISuM middleware is free software and is customized to support the LISA+ controller, it is recommended to have the LISA+ tools installed to run LiSuM middleware. Due to the fact that LISA+ is a commercial tool additional cost is needed to acquire the LISA+ planning tools as well as the virtual LISA+ TLC controller. The software license to used LISA+ products can be directly purchase by Schlothauer & Wauer company (4)

16.3.2 VISSIM

VISSIM is a microscopic multi-modal traffic flow simulation software package developed by PTV Planung Transport Verkehr AG in Karlsruhe, Germany (1). VISSIM was first developed in 1992 and is today a global market leader for commercial traffic flow simulation software. "Multi-modal simulation" describes the ability to simulate more than one type of traffic. All these types can interact mutually. In VISSIM the following types of traffic can be simulated: Vehicles, (cars, buses, and trucks), Public transport (trams, buses), Cycles (bicycles, motorcycles), Pedestrians and Rickshaws. The scope of application ranges from various issues of traffic engineering (transport engineering, transportation planning, signal timing), public transport, urban planning over fire protection (evacuation simulation) to 3D visualization (computer animation, architectural animation) for illustrative purpose and communication to the general public. To simulate traffic lights VISSIM can connect with an interface to the LISA+ software developed by Schlothauer & Wauer. The simulation software sends information about the current detector state and the signal state to the LISA+ interface and the simulated LISA+ traffic light controller. The simulated controller processed all received information and sends traffic signal switch commands back to VISSIM. Multiple intersections can be controlled simulataneously. How the simulated controller processes the data and reacts depends on the traffic-actuated control planed with LISA+ before.

16.3.3 SUMO

SUMO (Simulation of Urban Mobility) is an open source, highly portable, microscopic road traffic simulation package designed to handle large road networks. It is mainly developed by employees of the Institute of Transportation Systems at the German Aerospace Center (5) (2). SUMO is licensed under the GPL. It allows to simulate how a given traffic demand which consists of single vehicles moves through a given road network. The simulation allows addressing a large set of traffic management topics. It is purely microscopic: each vehicle is modelled explicitly, has an own route, and moves individually through the network. Simulations are deterministic by default but there are various options for introducing randomness. The software works on command line or can be used with the additional SUMO-GUI application which provides a graphical user interface to the simulation. Furthermore, SUMO has a list of additional tools like TraCI (Traffic Control Interface). TraCI giving the access to a running road traffic simulation, it allows to retrieve values of simulated objects and to manipulate their behavior "on-line". It can be used for adaptive traffic light control by a python script, for example.

16.4 Software Conception of the LISUM Middleware

This section describes the design and the development of the new software component LiSuM to couple virtual LISA+ controller and SUMO. Figure 16.3 shows how these components interact. It consists of communication interfaces, a graphical user interface and the configuration file. All these components are described in details in the following section.

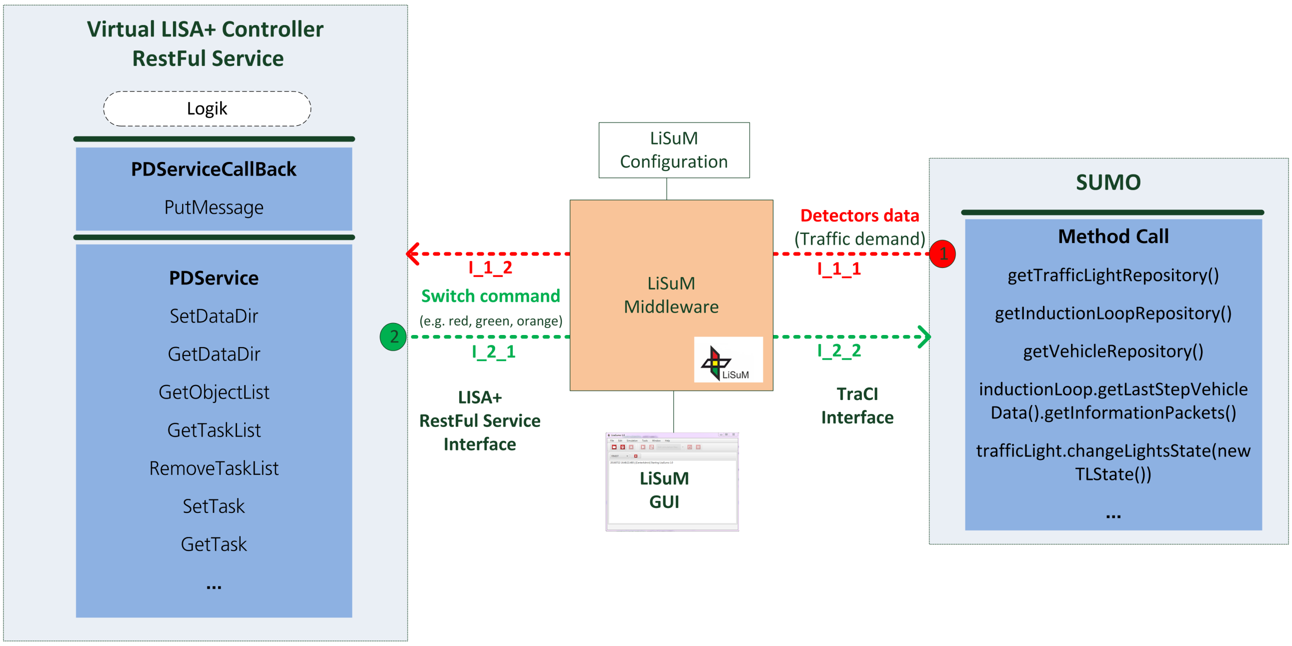


Figure 16.3: Overview about the LiSuM System

16.4.1 LiSuM Communication Interfaces

LiSuM middleware provides communication interfaces to couple virtual LISA+ controllers and SUMO. On one side, it communicates with the LISA+ controller over its RESTful interface(3), the communication protocol is REST/HTTP and the data format is described using WSDL/XSD files provides by Schlohauer & Wauer (4) and used to describe SOAP services. On the other side, LiSuM communicates with SUMO over TraCI4J (Traffic Control Interface for Java) interface (6).

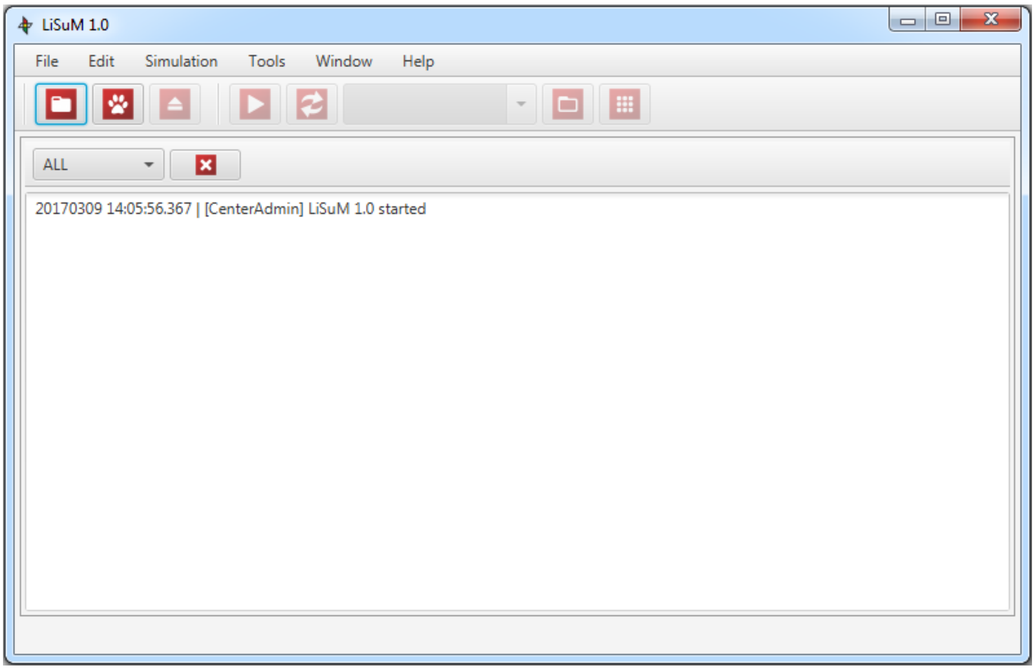
Table 16.1 gives an overview about the implemented interfaces by LiSuM. To retrieve or send the detector data and switch command for header signal, LISA+ controller server provides some RESTful services and SUMO some appropriate methods for this purpose (see Figure 16.3). Using the provides RESTful web services (LISA+) and TraCI4J (SUMO) interface the LiSuM middleware calls the corresponding method or services to transfer detector data from to the LISA+ Controller and send switch commands for all controlled traffic lights to the SUMO simulation. The LISA+ side of the interface is a simulated traffic light controller in an executable stand-alone Java JAR called LISA+ OMTC that simulates control logic programs. The virtual LISA+ controller acts as virtual RESTful Server and need to be installed in order to run LiSuM.

Table 16.1: Supported communication interfaces by LiSuM

No	Interface Identifier	Interface Type	Interface Description	Exchange Data	Source Component	Destination Component
1	I_1_1	TraCI	Interface to retrieve the detector data from SUMO to LiSuM	Detector data	SUMO	LiSuM
2	I_1_2	RESTful	Interface to send the detector data from LiSuM to LISA+	Detector data	LiSuM	LISA+
3	I_2_1	RESTful	Interface to retrieve the switch command from LISA+ to LiSuM	Switch Command	LISA+	LiSuM
4	I_2_2	TraCI	Interface to send the switch command from LiSuM to SUMO	Switch Command	LiSuM	SUMO

16.4.2 LiSuM Graphical User Interface (GUI)

In addition to the communication interface features to couple the virtual LISA+ TLS controller and SUMO, the LiSuM middleware provides also a graphical user interface. It main window is depicted in the Figure 16.4. The LiSuM GUI consists of menu items, toolbar buttons, and management console to the activities of the running processes during each simulation step, a configuration file and a Control Unit Management panel. The LiSuM GUI provides a series of functions linked to the toolbar buttons [see Table 16.2] and the menu items (and shortcut keys combinations) which allows to manage all aspects related to the interaction between LISA+ and SUMO such as opening a simulation project, starting a SUMO GUI instance, editing simulation project files without switching application and managing the LISA+ control units.

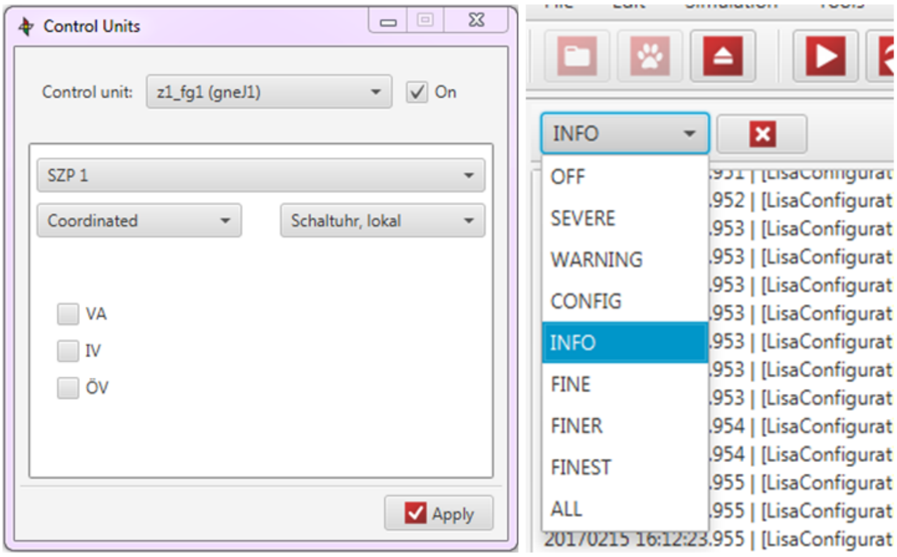


```
20170309 14:05:56.367 | [CenterAdmin] LiSuM 1.0 started
```

Figure 16.4: LiSuM GUI main window

16.4.3 LiSuM Control Unit Management

The Control Units Management dialog (Figure 16.5) allows selecting different signal programs for the each of the controlled traffic lights. It allows starting the programs and switching them to traffic actuated operation. By default, control units that do not have any Sumo intersection assigned are disabled. The communication with LISA+ can be toggled via the Ónćheckbox.



Control unit: z1_fg1 (gneJ1) ☑ On

SZP 1

Coordinated

Schaltuhr, lokal

□ VA

□ IV

□ ÖV

```
.951 | [LisaConfigurat
.952 | [LisaConfigurat
.953 | [LisaConfigurat
.953 | [LisaConfigurat
.953 | [LisaConfigurat
.953 | [LisaConfigurat
.953 | [LisaConfigurat
.953 | [LisaConfigurat
.953 | [LisaConfigurat
.954 | [LisaConfigurat
.954 | [LisaConfigurat
.955 | [LisaConfigurat
.955 | [LisaConfigurat
20170215 16:12:23.955 | [LisaConfigurat
```

Figure 16.5: LiSuM Control Unit Management window

16.4.4 Detector and Signal Group Mapping between LISA+ and SUMO

LiSuM interacts as mapping tool for detectors and signal groups between SUMO and the virtual LISA+ controller, because naming conventions in the two tools are different. This mapping is defined in the LiSuM configuration file (see Section 16.6 for more details). The differences in representation and naming between LISA+ and SUMO are described below:

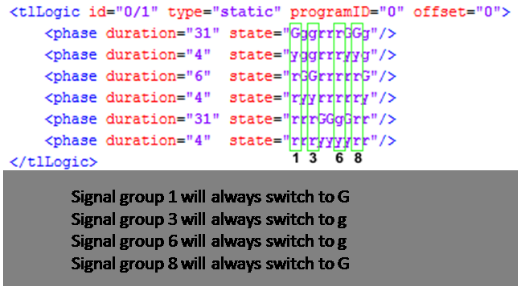
Table 16.2: LiSuM toolbar functions

No	Toolbar Icon	Function, shortcut and menu item
1	📁	Opens the file selector to open a simulation project(Ctrl+O)(File > Open)
2	🐾	Opens the last simulation project (Ctrl+1)
3	⏏	Closes the current simulation project (Ctrl+W) (File > Close)
4	▶	Opens the SUMO-GUI (Ctrl+P) (Simulation > Open SUMO)
5	🔄	Reloads the files of the current opened simulation project (Simulation > Reload simulation project files)
6	🗀	Explores the folder of the current opened simulation project
7	▦	Opens the control units management window(Ctrl+M) (Tools > Control Units Management)

- Induction loops
  - LISA+ assigns every detector a name as combination of character and number (e.g. D1 etc.), while SUMO allows arbitrary strings (e.g. myLoop1, etc.). `<detector lisa="D1" sumo="myLoop1" />`
- Traffic lights
  - Different names, in LISA+ format central number and field device number (z1_fg1) in SUMO any name (gneJ1). `<controlUnit lisa="z1_fg1" sumo="gneJ1" >`
  - LISA+ assigns every signal group a name as combination of character and number (e.g. K1, K2, etc.), while SUMO uses only a number (e.g. 0, 1, 2, etc.). `<signalGroup lisa="K1" sumo="0" />`

16.5 The Green Phase Mapping between LISA+ and SUMO

While LISA+ only defines and uses only one green phase, SUMO uses two different types of them: g (no priority) and G (priority). This distinction corresponds to additional signals which may signal full priority to left-turning vehicles. To do this modeling choice in SUMO, when LISA+ switches a signal group to green, LiSuM has to decide which of the two types of green it must send to the corresponding SUMO signal group. The ideal solution would be to represent to model the correspondence with the additional signals but the current version of LiSuM uses a simplified solution: LiSuM decides beforehand which green phase each SUMO signal group will ALWAYS receive based on the Gs and gs that each signal group has in their configuration strings in the net.xml file of the simulations project (tag tlLogic), having g priority over G. In other words, if a signal group has at least one g, that group will always switch to g, otherwise to G. This solution (see Figure 16.6) achieves save traffic behavior but slightly reduces travel speed in some situations.



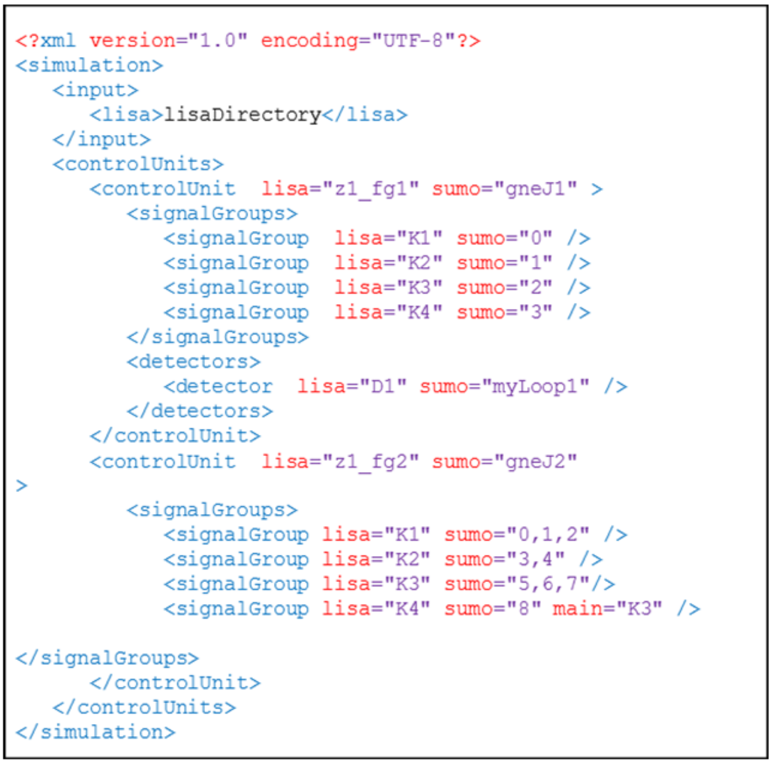
```
<tlLogic id="0/1" type="static" programID="0" offset="0">
    <phase duration="31" state="GggrrrGGg"/>
    <phase duration="4"  state="yggrrryyg"/>
    <phase duration="6"  state="rGGrrrrrG"/>
    <phase duration="4"  state="ryyrrrrry"/>
    <phase duration="31" state="rrrGGgGrr"/>
    <phase duration="4"  state="rrryyyyrr"/>
</tlLogic>
```

Signal group 1 will always switch to G
Signal group 3 will always switch to g
Signal group 6 will always switch to g
Signal group 8 will always switch to G

Figure 16.6: Some examples

16.6 LiSuM Configuration File

The LiSuM configuration file is a XML file (always named lisum.xml) which contains all necessary information used to control the way that LISA+ communicates with SUMO during the execution of a simulation. It is basically composed of the mandatory elements input and controlUnits that is composed of the two non-mandatory tags controlUnits and detectors. Since LISA+ and SUMO use different naming conventions for their elements, the fields controlUnits and detectors tell LiSuM how they are called in each system. The configuration file shown in Figure 16.7 above declares the following:



```
<?xml version="1.0" encoding="UTF-8"?>
<simulation>
   <input>
      <lisa>lisaDirectory</lisa>
   </input>
   <controlUnits>
      <controlUnit  lisa="z1_fg1" sumo="gneJ1" >
         <signalGroups>
            <signalGroup  lisa="K1" sumo="0" />
            <signalGroup  lisa="K2" sumo="1" />
            <signalGroup  lisa="K3" sumo="2" />
            <signalGroup  lisa="K4" sumo="3" />
         </signalGroups>
         <detectors>
            <detector  lisa="D1" sumo="myLoop1" />
         </detectors>
      </controlUnit>
      <controlUnit  lisa="z1_fg2" sumo="gneJ2"
>
         <signalGroups>
            <signalGroup lisa="K1" sumo="0,1,2" />
            <signalGroup lisa="K2" sumo="3,4" />
            <signalGroup lisa="K3" sumo="5,6,7"/>
            <signalGroup lisa="K4" sumo="8" main="K3" />

</signalGroups>
      </controlUnit>
   </controlUnits>
</simulation>
```

Figure 16.7: LiSuM sample configuration file

- The path of the directory where the LISA+ control logic files generate from the LISA+ OMTC Editor tool are locate is specified in the configuration file in the mandatory tags input and the element lisa as shown here <lisa>lisaDirectory</lisa>. (e.g. lisaDirectory = C:/lisa/). The generated LISA+ control logic files consist for example of followings files: z1_fg1.jar, z1_fg1.kfg, z1_fg1.szp, z1_fg1.xml

According to the configuration file, two control units are configured. The explanation is as follows:

- the first control unit:
  - is called z1_fg1 in LISA+ and gneJ1 in SUMO..
  - mapping of LISA+'s control unit z1_fg1 with gneJ1 in SUMO is provided.
  - contains one detector D1 in LISA+ and myLoop1 in SUMO and the mapping table.
  - is composed of four signal groups (K1, .., K4) for LISA+ and four (0, .., 4) for SUMO and contains the mapping information.
- the second control unit:
  - is called z1_fg2 in LISA+ and gneJ2 in SUMO.
  - contains also four signal groups (K1, .., K4) for LISA+ and nine (0, 1, .., 8, 9) for SUMO. It contains also the mapping information for example; the LISA+ signal group K2 controls the SUMO signal groups 3 and 4.
  - the fourth signal group contains the attribute main set to K3. This means that if the LISA+ signal group K4 is disabled (OFF) at any time of the cycle the SUMO signal group 8 will take the state of K3.

16.6.1 Getting started with LiSuM

LiSuM is licensed under the GPL and its current version is the 1.0.0 and was built on Java and thus can be run on any operating system. LiSuM is scheduled to be made available as part of the SUMO distribution. It is run by executing its jar file. It requires the Java SE Runtime Environment (version 8 or later), SUMO (version 0.29.0 or later) and LISA+ TLS planning tool and virtual controller installed in advance before beginning with the installation. The LISA+ tool and its virtual controller component is a commercial product and can be obtained directly from the Schothauer & Wauer company (4). When LiSuM is started for the first time, the user is prompted to select a the workspace directory. The workspace is the directory where LiSuM looks for existing simulation projects, where new ones should be stored and where the system preferences are saved. If needed use the system preferences window to change the workspace path. For better understanding of how LiSuM works, two sample projects (sampleSimulation and simpleSampleSimulation) are provided as part of the distribution.
First step: The first step consists of creating a new LiSuM simulation project, which is a directory containing:

- A LiSuM configuration file (lisum.xml)
- SUMO files (*.add.xml, *.net.xml, *.rou.xml, *.sumocfg, etc)
- A directory containing LISA+ control units files (exported from LISA+ OMTC)

Second step: The second step is to open LiSuM, set the SUMO path and select a workspace in the system configuration dialog and open a simulation project. In the tool menu, select "Start Lisa+ Virtual Controller" to start an instance of the virtual LISA+ Controller. Pressing Ctrl+p or clicking on the "Play" button on the toolbar will open an instance of the SUMO GUI, which will take control over the system. Almost all menus, toolbars and dialog windows of LiSuM get blocked and from here on the simulation may be started, paused, resumed and stopped from SUMO. Only the Control Units Management dialog window stays enabled so it is possible to change the control units settings during the execution of the simulation.

16.7 LiSuM Example: Single Intersection Control

In this section the exemplary simulation of a single intersection with a LISA+ TLS controller and SUMO is shown.

16.7.1 Using LISA+ OMTC Editor to plan and generate the control logic for an exemplary intersection with TLS and detector

The topology of the sample intersection selected to demonstrate the single-intersection control is depicted in Figure 16.8. It consists of three traffic lights and one loop detector. The simple example intersection has a major road from the north to the south and one minor road from the east. It has also single-lane approaches and added lanes for turning (see Figure 16.8). With LISA+ is a conventional vehicle-actuated control based on three phases build for this example intersection (see Figure 16.9).

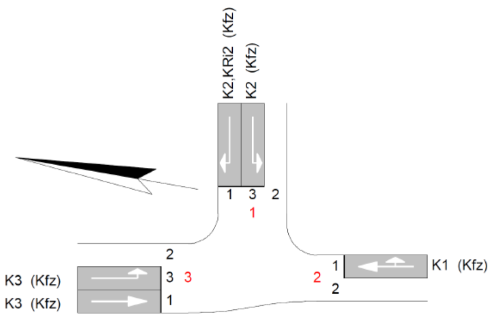


Figure 16.8: Topology and signal groups of the example intersection

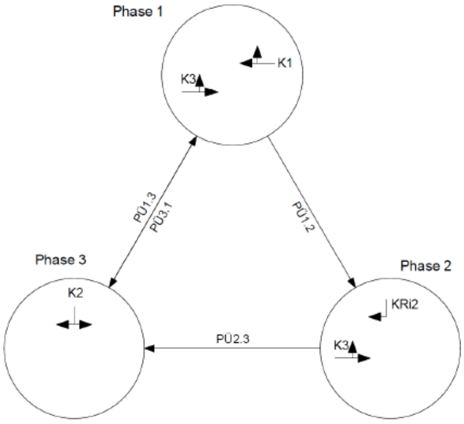
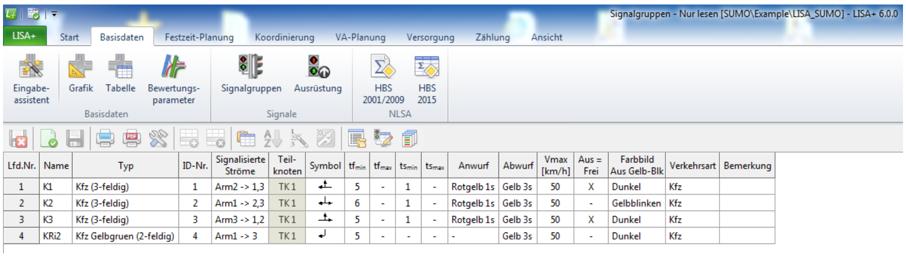


Figure 16.9: Sketch and phase plan of the example intersection

- Phase one gives green to the approaches from north and south.
- Phase two gives green only for the north approach for the better traffic flow from north to east if it is necessary.
- Phase three gives green to the minor road. Furthermore, the minor road only gets green on demand and all phases have a green time adjustment by vehicle time gaps. A time gap from 1.5 seconds or more stops the current phase. The information about vehicle presence and time gaps are detected by induction loops in SUMO.

The signal groups (K1, K2 and K3) corresponding to the single traffic light signals located in the sample intersection is planned with the LISA+ Editor as shown in the Figure 16.10. The next step is to use LISA+OMTC Tool (see Figure 16.11) to plane the traffic-actuated control logic for the single signal group (K1, K2, K3) and the loop detector (D1). After that the files containing the traffic actuated control logic can be generated and save into the LISA directory into the LiSuM middleware as described in Section 16.6.



Lfd.Nr.	Name	Typ	ID-Nr.	Signalisierte Ströme	Teil-knoten	Symbol	tf_{min}	tf_{max}	ts_{min}	ts_{max}	Anwurf	Abwurf	Vmax [km/h]	Aus = Frei	Farbbild Aus Gelb-Blk	Verkehrsart	Bemerkung
1	K1	Kfz (3-feldig)	1	Arm2 -> 1,3	TK 1	↰↑	5	-	1	-	Rotgelb 1s	Gelb 3s	50	X	Dunkel	Kfz	
2	K2	Kfz (3-feldig)	2	Arm1 -> 2,3	TK 1	↔	6	-	1	-	Rotgelb 1s	Gelb 3s	50	-	Gelbblinken	Kfz	
3	K3	Kfz (3-feldig)	3	Arm3 -> 1,2	TK 1	↑↱	5	-	1	-	Rotgelb 1s	Gelb 3s	50	X	Dunkel	Kfz	
4	KRi2	Kfz Gelbgruen (2-feldig)	4	Arm1 -> 3	TK 1	↲	5	-	-	-	-	Gelb 3s	50	-	Dunkel	Kfz	

Figure 16.10: Planed signal groups in LISA+ OMTC

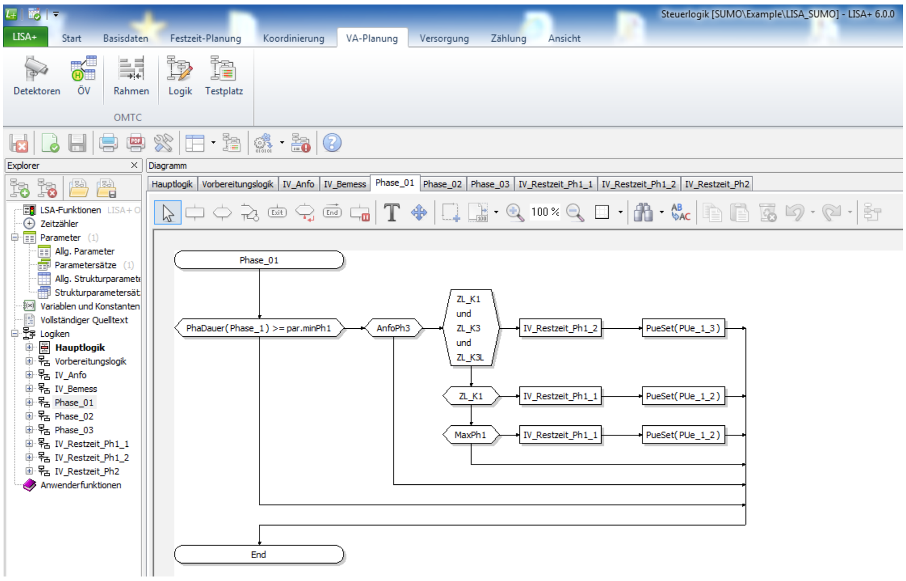
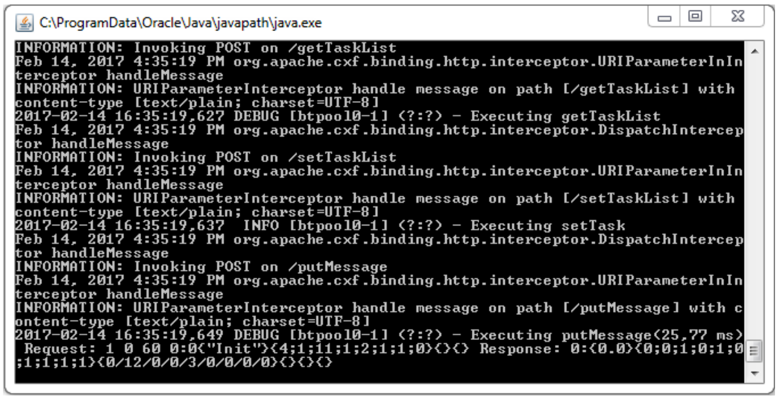


Figure 16.11: Planed traffic-actuated control logic in LISA+ OMTC [4]

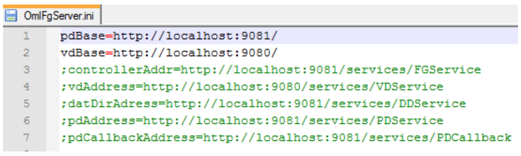
16.7.2 Virtual LISA+ TLS Controller (LISA+ RESTful Server)

The virtual LISA+ TLS controller is provided as an executable stand-alone Java JAR file containing the LISA+ RESTful Server and is used to simulate control units devices and run control logics on it. Therefore it is necessary to have LISA+ installed to be able to run LiSuM. When starting a simulation using LiSuM GUI an instance of a LISA+ RESTful server must be running (see Figure 16.12). Per default the LISA+ RESTful server files can be found in the directory OmlFgServer. Per default, LiSuM searches for simulations in the simulations directory, which can be found in the installation directory (../DLR/LiSuM). This can be changed in the System Preferences Window (Menu: Tools –> Preferences). To start an instance goes to Menu: Tools –> Start LISA+ RESTful Server. The control logic (generated files in Section 16.7) previously created using LISA+ OMTC including all its parameter for the sample intersection is uploaded to the virtual LISA+ controller. The control logic files can be found in the OmlFgServer directory located in the LiSuM installation directory. The communication port and the RESTful web service address can be found and edited in the OmlFgServer.ini file (see Figure 16.13), if necessary.



```
INFORMATION: Invoking POST on /getTaskList
Feb 14, 2017 4:35:19 PM org.apache.cxf.binding.http.interceptor.URIParameterInIn
terceptor handleMessage
INFORMATION: URIParameterInterceptor handle message on path [/getTaskList] with
content-type [text/plain; charset=UTF-8]
2017-02-14 16:35:19,627 DEBUG [btpool0-1] (?:?) - Executing getTaskList
Feb 14, 2017 4:35:19 PM org.apache.cxf.binding.http.interceptor.DispatchIntercep
tor handleMessage
INFORMATION: Invoking POST on /setTaskList
Feb 14, 2017 4:35:19 PM org.apache.cxf.binding.http.interceptor.URIParameterInIn
terceptor handleMessage
INFORMATION: URIParameterInterceptor handle message on path [/setTaskList] with
content-type [text/plain; charset=UTF-8]
2017-02-14 16:35:19,637  INFO [btpool0-1] (?:?) - Executing setTask
Feb 14, 2017 4:35:19 PM org.apache.cxf.binding.http.interceptor.DispatchIntercep
tor handleMessage
INFORMATION: Invoking POST on /putMessage
Feb 14, 2017 4:35:19 PM org.apache.cxf.binding.http.interceptor.URIParameterInIn
terceptor handleMessage
INFORMATION: URIParameterInterceptor handle message on path [/putMessage] with c
ontent-type [text/plain; charset=UTF-8]
2017-02-14 16:35:19,649 DEBUG [btpool0-1] (?:?) - Executing putMessage(25,77 ms)
 Request: 1 0 60 0:0("Init"){4;1;11;1;2;1;1;0}{}{} Response: 0:{0.0}{0;0;1;0;1;0
;1;1;1;1}{0/12/0/0/3/0/0/0/0}{}{}{}
```

Figure 16.12: Running Terminal window of LISA+ RESTful Server (as virtual LISA+ TLS Controller)

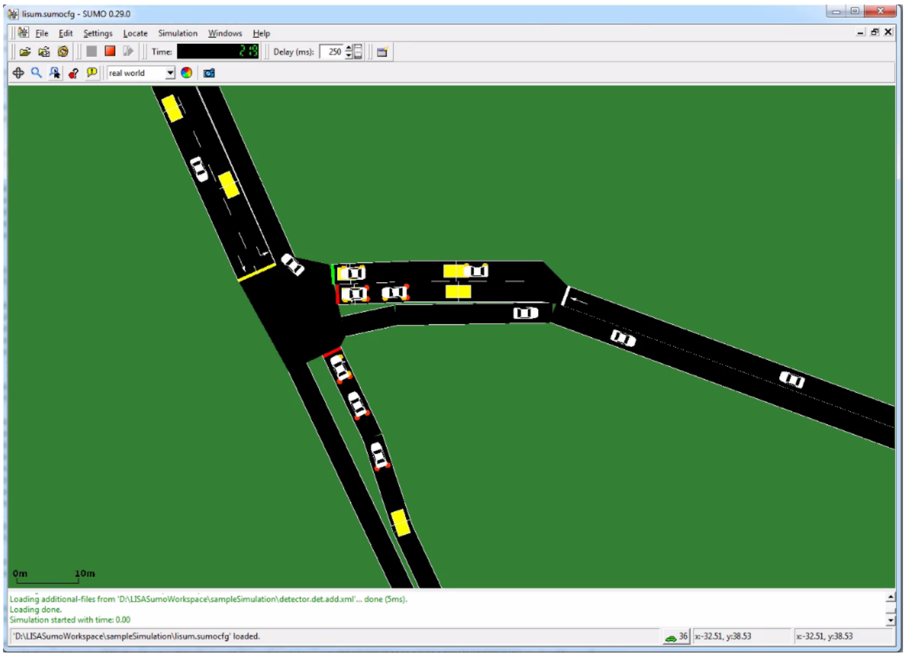


```
pdBase=http://localhost:9081/
vdBase=http://localhost:9080/
;controllerAddr=http://localhost:9081/services/FGService
;vdAddress=http://localhost:9080/services/VDService
;datDirAdress=http://localhost:9081/services/DDService
;pdAddress=http://localhost:9081/services/PDService
;pdCallbackAddress=http://localhost:9081/services/PDCallback
```

Figure 16.13: Overview about the OmlFgServer.ini file of LISA+ RESTful Server

16.7.3 SUMO Simulation

After starting the SUMO Simulation using LiSuM GUI (start button in toolbar or simulation item in menu bar). The controls created previously with LISA+ can be directly simulated with the simulation tool SUMO. The exchange of traffic demand information from configured loop detector in SUMO and the corresponding switch commands of the header signal (e.g. red, yellow and green) from the virtual LISA+ TLS controller happen via the implemented interfaces of LiSuM. The result of the simulation with the LiSuM interface can be shown the green light allocation in SUMO (see Figure 16.14 and 16.15) depending on the given traffic demand and the control method described previously in Section 16.7.



Loading additional-files from 'D:\LISASumoWorkspace\sampleSimulation\detector.det.add.xml'... done (5ms).
Loading done.
Simulation started with time: 0.00

'D:\LISASumoWorkspace\sampleSimulation\lisum.sumocfg' loaded.

Figure 16.14: Sample intersection in SUMO

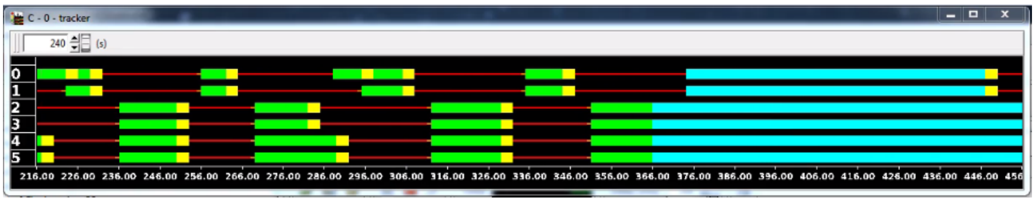


Figure 16.15: Signal timeline for the Sample intersection in SUMO

16.7.4 Performance Measurement of the Single Example Intersection

To compare the effect of LiSuM on simulation speed, the example scenario was executed with LiSuM and without out (using a fixed traffic light control program) on the same computer. Without LiSuM, the example scenario runs with a real time factor of 290 (simulating 290 minutes of traffic can be accomplished in a single minute of computing time). With LiSuM the overall simulation speed is reduced to a real time factor of 85 due to communication and synchronization delays. The LiSuM middleware can simulate multiple traffic light controllers at the same time and due to these preliminary results we expect scenarios with at least 10 controllers to be feasible.

16.8 Conclusion and Outlook

Only a small number of traffic-actuated controls schemes is available in SUMO and therefore most traffic lights can only be simulated with custom control scripts that must be developed on a case-by-case basis. To allow SUMO to simulate complex scenarios for traffic light signal (TLS) control like single intersections, coordination and adaptive controls, the realization of the communication interface between SUMO and existing virtual TLS controller is required. To close this gap DLR designed and developed a new software tool called LiSuM. LiSuM allows the communication between SUMO and the virtual LISA+ controller. LISA+ is a commercial product from the German company Schlothauer & Wauer that provides a series of tools for traffic signal and control logic planning. Control logic created with LISA+ OMTC is uploaded to the virtual LISA+ controller and can be directly simulated with the SUMO. LiSuM middleware provides communication interfaces to exchange detector data as traffic demand provided by SUMO and the switch commands of header signals (red, yellow and green) obtained from the virtual controller. LiSuM also provides a graphical user interface, which allows managing all aspects related to the interaction between the virtual LISA+ controller and SUMO. To facilitate this interaction, name-mappings between the LISA+ and SUMO domains must be configured beforehand. This paper describes the design and development of the LiSuM tool and demonstrates its usage with an example scenario.

16.9 Bibliography

[1] AG, PTV: *PLANUNG TRANSPORT VERKEHR, VISSUM*, 2017.

[2] Krajzewicz Daniel, Jakob Erdmann, Michael Behrisch and Laura Bieker: *Recent Development and Applications of SUMO - Simulation of Urban MObility*. International Journal On Advances in Systems and Measurements, December 2012.

[3] RESTfull: *RESTful web service*, 2017.

[4] SCHLOTHAUER and WAUER: *LISA+*, 2017.

[5] SUMO: *Simulation of Urban Mobility*, 2017.

[6] SUMO: *TraCI Traffic ControlInterfaces description*, 2017.

ISSN 1866-721X