SUMO Conference Proceedings
1 | 2020

TIB OPEN PUBLISHING

SUMO User Conference

26 – 28 October 2020
Virtual Event

SUMO
User Conference
October 26-28, **2020**

Editors:

Pablo Alvarez Lopez
Olaf Angelo Banse Bueno
Maria Giuliana Armellini
Michael Behrisch
Laura Bieker-Walz
Jakob Erdmann
Yun-Pang Flötteröd
Robert Hilbrich
Ronald Nippold
Johannes Rummel
Matthias Schwamborn
Peter Wagner
Melanie Weber

DLR Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

SUMO Conference Proceedings

SUMO Conference Proceedings (SCP) is dedicated to publish the proceedings of the SUMO user conference.

Traffic simulations are of immense importance for researchers as well as practitioners in the field of transportation. SUMO has been available since 2001 and provides a wide range of traffic planning and simulation applications. SUMO consists of a suite of tools covering road network imports and enrichment, demand generation and assignment, and a state-of-the-art microscopic traffic simulator capable of simulating private and public transport modes, as well as person-based trip chains. Being open source, SUMO is easily extensible by new behavioral models and can be dynamically controlled via a well-defined programming interface. These and other features make SUMO one of the most popular open source traffic simulators with a large and international user community.

The SUMO Conference aims in bringing SUMO users and people interested in traffic simulation and modelling together to exchange their research results, used models and tools and discuss their findings. The papers of this conference can be found here publicly available.

ISSN (online): 2750-4425

SUMO Conference Proceedings (SCP) are published by TIB Open Publishing (Technische Informationsbibliothek, Welfengarten 1 B, 30167 Hannover) on behalf of DLR Institute of Transportation Systems.

Volume 1

SUMO User Conference 2020

26 – 28 October 2020, Virtual Event

Conference papers

https://doi.org/10.52825/scp.v1i

Editors

Pablo Alvarez Lopez, German Aerospace Center, Institute of Transportation Systems

Olaf Angelo Banse Bueno, German Aerospace Center, Institute of Transportation Systems

Maria Giuliana Armellini, German Aerospace Center, Institute of Transportation Systems

Michael Behrisch, German Aerospace Center, Institute of Transportation Systems

Laura Bieker-Walz, German Aerospace Center, Institute of Transportation Systems

Jakob Erdmann, German Aerospace Center, Institute of Transportation Systems

Yun-Pang Flötteröd, German Aerospace Center, Institute of Transportation Systems

Robert Hilbrich, German Aerospace Center, Institute of Transportation Systems

Ronald Nippold,German Aerospace Center, Institute of Transportation Systems

Johannes Rummel, German Aerospace Center, Institute of Transportation Systems

Matthias Schwamborn, German Aerospace Center, Institute of Transportation Systems

Peter Wagner, German Aerospace Center, Institute of Transportation Systems

Melanie Weber, German Aerospace Center, Institute of Transportation Systems

Review process

The papers are reviewed by at least two independent reviewers. All papers which are submitted by authors from DLR are only reviewed by external steering committee members to avoid conflicting interests. After the review, the author receives the decision whether the paper was accepted or has been rejected. Additionally, they are receiving remarks, questions and hints how to improve the quality of the papers for the final version of the publication. The authors have at least two weeks to rework and submit the final version of their paper.

Financing

The organization of the conference and publication of the papers are financed by the conference fee of the participants of the SUMO conference.

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.95

Published: 28 Jun. 2022

Extending the Intelligent Driver Model in SUMO and Verifying the Drive Off Trajectories with Aerial Measurements

Dominik Salles[1], Stefan Kaufmann[2] and Hans-Christian Reuss[3]

[1] Research Institute of Automotive Engineering and Vehicle Engines Stuttgart (FKFS), Germany
[2] IT-Designers GmbH, Esslingen, Germany
[3] University of Stuttgart – Institute of Automotive Engineering (IFS), Germany
dominik.salles@fkfs.de

Abstract

Connected and automated driving functions are key components for future vehicles. Due to implementation issues and missing infrastructure, the impact of connected and automated vehicles on the traffic flow can only be evaluated in accurate simulations. Simulation of Urban Mobility (SUMO) provides necessary and appropriate models and tools. SUMO contains many car-following models that replicate automated driving, but cannot realistically imitate human driving behavior. When simulating queued vehicles driving off, existing car-following models are neither able to correctly emulate the acceleration behavior of human drivers nor the resulting vehicle gaps. Thus, we propose a time-discrete 2D Human Driver Model to replicate realistic trajectories. We start by combining previously published extensions of the Intelligent Driver Model (IDM) to one generalized model. Discontinuities due to introduced reaction times, estimation errors and lane changes are conquered with new approaches and equations. Above all, the start-up procedure receives more attention than in existing papers. We also provide a first evaluation of the advanced car-following model using 30 minutes of an aerial measurement. This dataset contains three hours of drone recordings from two signalized intersections in Stuttgart, Germany. The method designed for extracting the vehicle trajectories from the raw video data is outlined. Furthermore, we evaluate the accuracy of the trajectories obtained by the aerial measurement using a specially equipped vehicle.

1 Introduction

In the last few decades, many traffic simulation software packages have been developed to study traffic flow and movement patterns of pedestrians. Multiple reviews regarding performance, usability and portability of the programs SUMO (Lopez et al. 2018), PTV Vissim, Aimsun, Paramics, MATSim, CORSIM and TRANSIM have been published in Kotusevski and Hawick (2009), Ejercito et al. (2017) and Dallmeyer (2014). SUMO's ability to handle large networks and its open source framework make it attractive for researchers. The source code, written in C++, can be fully examined and modified. This creates a foundation for integrating custom car-following models and devices. Another notable add-on is the "Traffic Control Interface" (TraCI) (Wegener et al. 2008), which enables the communication with SUMO via the Transmission Control Protocol (TCP).

Simulation studies and modifications utilizing SUMO are often related to connected and autonomous driving functions. VSimRTI (Queck et al. 2008) and Veins (Sommer et al. 2008) are two exemplary communication tools, which use TraCI to connect the traffic simulation SUMO with network simulators (NS2/NS3 or OMNET++), so that real-world network protocols and V2V communication can be developed and tested. A multitude of researchers have carried out investigations of how connected and autonomous driving will change the road capacity and future traffic flow, by using custom programmed devices and extensions, such as the Adaptive Cruise Control (ACC) and Cooperative Adaptive Cruise Control (CACC) models (Milanés and Shladover 2014; Xiao et al. 2017; Xiao et al. 2018). Among them, Alekszejenkó and Dobrowiecki (2019) present an intelligent traffic control algorithm coupled with platooning vehicles developed in SUMO to improve urban traffic flow. They point out that future work would need to analyze the impact of human drivers in the scenario to better quantify the improvements. Richter et al. (2019) actually study the effect of mixed traffic (autonomous vehicles and human drivers) on a highway. They use the Krauss model (Krauss et al. 1997) and define smaller time headways, reaction times and sigma values (driver imperfection) for autonomous vehicles.

Other studies in this field, e.g., Derbel et al. (2012) and Zhou et al. (2016), use the IDM to represent the automated vehicles. They show that the original IDM is particularly well equipped to replicate automated driving, while the human driving behavior is either simulated using the Two Velocity Difference Model (Derbel et al. 2012) or the Full Velocity Difference Model (Zhou et al. 2016).

SUMO is also often used for extracting realistic trajectories and their characteristic values. The trajectories are then used to retrieve typical driving cycles to calculate the energy consumption of vehicles (Macedo et al. 2013; Pfeil 2019; Donateo et al. 2010). These studies use SUMO to incorporate the effect of infrastructure and traffic dynamics on the consumption. Grumert et al. (2015) and Erdağı et al. (2019) go one step further, not merely focusing on the energy used, but also taking the emissions into account. The Krauss model was used for these investigations, which result in realistic velocities and traffic flow, but may produce unrealistic accelerations and therefore emissions. When using macroscopic values in combination with, e.g., the Handbook Emission Factors for Road Transport (HBEFA), the calculated values can be fairly accurate, but when extracting single trajectories, the results can highly depend on the acceleration. For this reason, a car-following model is needed that can produce realistic accelerations and jerks.

2 Related Work

According to the thorough review in Saifuzzaman and Zheng (2014), car-following models can be categorized as follows: safety distance models, Cellular Automata, Optimal Velocity Models, desired measures models, Gazis-Herman-Rothery models and their extensions as well as models based on the human perspective. The last category contains models with perception thresholds, models based on risk-taking of the driver or driving by visual angle. Enhancements include fuzzy logic, distraction and driver errors.

Safety distance models are designed to always provide a safe distance and prevent collisions. In the Gipps model (Gipps 1981), this is achieved without considering the speed of the respective vehicle. The virtual driver intends to always brake with parameter *b*, expecting the leader to decelerate with the identical value. In heterogeneous traffic, this can result in collisions. When the braking parameter *b* of the leader is assumed to be higher, unrealistically large gaps form between the vehicles.

Optimal Velocity Models (OVM), originally developed by Bando et al. (1995), use a different approach. They employ a constant sensitivity coefficient to describe the reactivity of the driver. The model, however, is not free of collisions, and when the coefficient is selected to avoid accidents, it induces high accelerations and decelerations. Continuing developments led to the Generalized Force Model (Treiber et al. 2000) and the Full Velocity Difference Model (FVDM) (Jiang et al. 2001). In contrast to the OVM, the FVDM takes the velocity of the leader into account and, thus, stabilizes the model.

Human drivers do not react to every single change of the environment. Wiedemann incorporated this observation into his psycho-physical model (Wiedemann 1974). The model features perception thresholds, which are only exceeded when the variables change significantly. In addition, The model differentiates between four different driving modes, Free driving, approaching, following and strong braking, and it has many more functions and parameters than other car-following models.

In contrast, Cellular Automata (CA) and the Krauss model are much less detailed. Cellular Automata as car-following models were first introduced by Nagel and Schreckenberg (1992). They operate in a time- and space-discrete fashion. The road is therefore divided into equally large cells that can only be occupied by one vehicle at a time. The vehicle's speed is randomized to produce stochastic behavior. Despite its simple nature, this model can realistically reproduce traffic phenomena. The Krauss model, the default car-following model in SUMO (Krauss et al. 1997), can be considered a space-continuous version of the Nagel and Schreckenberg model. It combines the advantages of the CA model with those of the Gipps model. The Krauss model operates collision-free and contains only a few functions and parameters. Additional advantages are the asymmetrical acceleration-deceleration behavior and the emulation of the human reaction time with the simulation step time. Therefore, this model can well replicate many observed traffic phenomena (Alazzawi et al. 2018), despite producing unrealistic acceleration and jerk patterns.

This research article focuses mainly on the realistic representation of human acceleration patterns using car-following models. The selected integration scheme has a significant influence on this representation. As pointed out by Treiber and Kanagaraj (2015), the ballistic update is always more accurate than the Euler method or faster with the same accuracy. In SUMO, you can choose either one. Due to discontinuities, such as lane changes, the standard fourth-order Runga-Kutta method is not applicable for multi-lane traffic simulations. Therefore, car-following models like the OVM, FVDM, GHR and IDM have to be described in a time-discrete fashion, although they were all developed as time-continuous models. In this study, the Euler method was used, as it is the default integration scheme in SUMO.

The original IDM produces realistic accelerations and jerks almost independently of the integration method and has been continuously modified since its introduction. Shortly thereafter, Treiber and Helbing (2004) concluded that human drivers leave larger gaps when driving off than the IDM predicts, although the jerk remains realistically small in all driving situations. It should not exceed 1.5 m/s^3 (Treiber and Kesting 2013b). To achieve this goal, all discontinuities that can occur in SUMO's traffic simulation must be taken into account.

They include:

- Change of speed limits
- Reaction times, light signals and priorities at junctions before drive off
- Reaction times before braking
- Greater gaps after lane changes
- Smaller gaps after lane changes
- Driving behavior near the minimal gap

The Extended Intelligent Driver Model (EIDM) is introduced in the following section. Chapter 4 presents the case study and aerial measurement method. Similar databases to the here presented measurement method have recently been released and published by Krajewski et al. (2018) and Bock et al. (2019). Their drones observed German highways and unsignalized intersections, but urban traffic phenomena at signalized intersections are not included. The advantages of the drone measuring method are pointed out in the above mentioned articles: naturalistic driving behavior and the possibility to simultaneously capture the movement of multiple vehicles without any occlusion. The authors compare their datasets with that of the New Generation SIMulation (NGSIM) set (Kovvali et al.) and other urban intersection data. The NGSIM dataset includes trajectories of vehicles crossing intersections, but it is not possible to use the raw trajectories without first smoothing the data or re-extracting the trajectories from the video (Thiemann et al. 2008; Krajewski et al. 2018; Coifman and Li 2017). Consequently, the acceleration patterns are considered unrealistic.

3 Car-following model

This chapter starts off by reviewing the original IDM and its substantial enhancements so far. Subsequently, the EIDM is presented with all its modifications. Figure 1 defines the nomenclature, since many different definitions can be found in the literature. All listed parameters belong to the following vehicle and all variables regarding the leader will carry the subindex *n*.

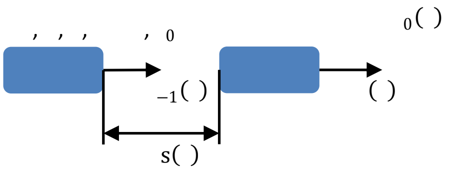


Figure 1: Notation of the car-following model

3.1 Improved Intelligent Driver Model

The IDM, first introduced as a time-continuous model (Treiber et al. 2000), consists of two main equations and five parameters, the desired time headway , the maximum acceleration , the desired deceleration , the minimum gap $_0$ and the acceleration exponent . The desired gap

$$ {}^{*}{}_{-1}(\ ) = {}_{0} + (0,\ {}_{-1}(\ ) * - \frac{{}_{-1}(\ ) * (\ (\ ) - {}_{-1}(\ ))}{2 * \sqrt{\qquad *}}) \tag{1}$$

depends on three of those parameters and the velocities ( ) and $_{-1}$( ). The acceleration

$$ (\ + \Delta\ ) = [1 - (\frac{{}_{-1}(\ )}{{}_{0}(\ )}) - (\frac{{}^{*}{}_{-1}(\ )}{(\ )})^{2}] \tag{2}$$

is determined by calculating the ratio between the current velocity $_{-1}( )$ and the desired velocity $_0( )$ and the ratio between the desired gap $^*{}_{-1}( )$ and the actual gap $( )$. The latter ratio represents the intelligent braking strategy and assures a collision-free execution of this model. However, this term does not allow following vehicles to reach the desired velocity in homogeneous traffic conditions and induces ever larger gaps. The Improved Intelligent Driver Model (IIDM) accounts for this negative trait by changing the model characteristics close to the desired velocity (Treiber and Kesting 2013b). The new term of free acceleration leads to more realistic gaps between vehicles. The authors calculate $( )$ by differentiating between two cases: $_{-1}( ) \leq {}_0$ and $_{-1}( ) > {}_0$. The EIDM calculates the free acceleration

$$( ) = \qquad [1 - (\frac{_{-1}( )}{_0( )}) \ ] \tag{3}$$

without using the case distinction of the IIDM, but by linearizing the changes in the desired velocity $_0( )$ (see Section 3.4). The following equation of the resulting acceleration $(\ + \Delta \ )$ further differs from that of the IIDM. Instead of calculating the exponent with $( )$, its absolute value is used. If the exponent were negative, the acceleration would be unsteady at $_0( )$.

The resulting acceleration

$$(\ + \Delta \ ) = \begin{cases} [1 - (\frac{^*{}_{-1}( )}{( )})^2 \ ] & ^*{}_{-1}( ) \geq \ ( ) \\ ( )\,[1 - (\frac{^*{}_{-1}( )}{( )})^{\frac{2*}{|\quad ( )|}} \ ] & h \end{cases} \tag{4}$$

differentiates between two cases: driving at distances lower than the desired gap and higher than the desired gap.

3.2 Human Driver Model

The Human Driver Model (Treiber et al. 2006) was developed by the authors of the IDM. To generate human driver behavior in the model, they introduced a reaction time, imperfect estimation capabilities and temporal and spatial anticipation. Spatial anticipation has not yet been integrated into the EIDM in SUMO.

Due to the time-continuous form of the IDM, the reaction time was introduced with a Delay Differential Equation (DDE). As SUMO runs in a time-discrete fashion, the variables of the last several time steps would need to be stored in vectors to employ this method, thereby requiring significant amounts of memory. In addition, the driver always reacts to an earlier state $-$ ago and the model needs to be carefully calibrated to stay stable. To solve these issues, Action Points (APs) are introduced. Simulating with APs implies that the driver can instantaneously process any information at the action time . Between two APs, the model uses the variables from the last AP update. Furthermore, can be varied throughout the simulation to overcome stability issues when the driver needs to react quickly.

The estimation errors are modeled using a Wiener process, which is defined by the variable , determined at step *i*, using the correlation time $\tilde{}$, a randomized number of variance 1 and the time step Δ of SUMO:

$$= -\frac{\Delta}{\tilde{}} * \quad _{-1} + \sqrt{\frac{2\Delta}{\tilde{}}} * \qquad (5)$$

The variable is then used to calculate the estimated distance ( ), the estimated velocity of the leader ( ) and a driving error ( ), which is added to the acceleration term. For the systematic derivation, see Treiber et al. (2006).

In equations (6), (7) and (8), the variables ( ), ( ), ( ) are the corresponding Wiener processes, represented in (5). The parameters , and describe the respective magnitude of the errors.

$$(\ ) = \ (\ ) * \quad ^{(\ )} \qquad (6)$$

$$(\ ) = -\ (\ ) \quad (\ ) + \ (\ ) \qquad (7)$$

$$\tilde{}\ (\ ) = \ (\ ) + \quad (\ ) \qquad (8)$$

By introducing a reaction time, the model can become unstable and simulate accidents. To prevent those the model uses anticipation terms. The driver anticipates the velocity of the leader and his own acceleration to remain constant until the next AP, resulting in the predicted velocities and distances in (9), (10) and (11).

$$_{-1}\ (\ ) = \ _{-1}(\quad) + (\ - \quad) * \ (\quad) \qquad (9)$$

$$(\ ) = \quad (\quad) \qquad (10)$$

$$(\ ) = \quad (\ ) - (\ - \quad) * \Delta \ _{-1}(\quad) \qquad (11)$$

3.3 Enhanced Intelligent Driver Model

The Enhanced IDM improves the lane changing behavior of the original IDM, since it was first developed as a single-lane model (Kesting et al. 2010). This model reduces the deceleration when gaps are instantaneously reduced after lane changes and, nevertheless, remains collision-free. A new equation calculates the Constant Acceleration Heuristic (CAH) ( ) as follows, taking the acceleration ( ) of the leader into account.

$$(\ ) = \begin{cases} \dfrac{^{2}\ _{-1}\tilde{}}{^{2} - 2\mathrm{s(t)}\tilde{}} & (\ _{-1} - \ ) \leq -2\mathrm{s(t)}\tilde{} \\ \tilde{}\ - \dfrac{(\ _{-1} - \ )^{2}}{2\mathrm{s(t)}} & h \end{cases} \qquad (12)$$

$$= \begin{cases} 0 & _{-1} - \quad < 0 \\ 1 & _{-1} - \quad \geq 0 \end{cases} \qquad (13)$$

$$\tilde{} \ = \min(\ (\ ), \quad) \qquad (14)$$

is the Heaviside step function. The CAH-model cannot operate as a stand-alone model, it is used as an extension of the IDM. The acceleration is calculated using the new coolness parameter , with values between 0 and 1. It describes how “cool” a driver reacts when gaps are reduced.

$$= \{ \begin{matrix} & & & & & \geq \\ (1- \quad) & + & [\quad + \quad * \tanh (\frac{\quad - \quad}{\quad})] & & h & \end{matrix} \tag{15}$$

This results in a temporary acceptance of lower gaps. Without this modification, the vehicles in SUMO often do not change lanes or brake hard after lane changes when using the IDM.

3.4 Further EIDM enhancements

In order to create a realistic human driver model in SUMO, a few more modifications have to be carried out. First of all, the introduced estimation errors und reaction times cause problems when decelerating to 0 m/s. Acceleration jumps occur at small gaps because the predicted and estimated values are wrought with intentional errors. Consequently, the virtual driver cannot smoothly approach the minimal gap $_0$. That problem is solved by introducing equation (16). When reaching the gap $_0 +$ γ (minimal gap plus threshold), the vehicle is forced to decelerate further, although the desired gap $^*{}_{-1}(\ )$ might be smaller than the actual gap $(\ )$. This leads to vehicles stopping prior to $_0$, but overcomes the effect of oscillating accelerations at low gaps. γ values between 0.3 and 0.5 were empirically determined to be suitable.

$$^*{}_{-1}(\ ) = \{ \begin{matrix} (\ ) + 0.05 & & ^*{}_{-1}(\ ) < \ (\ ) < \ _0 + \gamma \\ ^*{}_{-1}(\ ) & & h \end{matrix} \tag{16}$$

Analog to the definition of the speed factor in SUMO, every vehicle is assigned an individual minimal gap $_0$ from a normal distribution.

Another adjustment is made regarding changing speed limits. The presented equations in Section 3.1 do not take changing speed limits into account. We therefore use the simple linear function in (17) to continuously change the desired velocity when the speed limit changes. The model receives a new parameter to look $*$ $_{-1}$ meters ahead. This results in a model-internal desired velocity $_0$ when driving near two edges with different speed limits $_0$ and $_0{}^{+1}$. The distance to the upcoming edge is represented by s (t).

$$_0\ (\ ) = \{ \begin{matrix} _0 & - (\ _0 - \ _0{}^{+1}) * \Delta\ / & & \mathrm{s}\ (\mathrm{t}) < & * \quad _{-1} \\ _0 & & & h & \end{matrix} \tag{17}$$

Furthermore, all turns at junctions receive a speed limit according to Table 1, which is used to limit the velocity when turning. The parameters in Table 1 refer to the turn categories in SUMO, which are defined by the turn’s radius. Alternatively, the limits could be calculated using the specific radius of the turn or street curvature.

To update the desired velocity with the maximum assigned speed of the next turn, the model uses Equation (17) with a look ahead of $*$ $_{-1}$ in order for the vehicle to reach that speed before turning.

Parameter LINKDIR	TURN	TURN _LEFTHAND	LEFT	RIGHT	PARTLEFT	PARTRIGHT
$_0$ [m/s^2]	5.0	5.0	9.0	8.0	12	12

Table 1: Speed limits for turns at junctions in SUMO

According to Wagner and Lubashevsky (2003), the time period between subsequent human driver decisions can amount to several seconds. For the model, such large reaction times result in hard braking behavior. This is solved by introducing variable APs (Treiber and Kesting 2017). Equation (18) is based on a similar formula, but uses a predefined action time and a constant threshold instead of a random number. This stabilizes the model during critical events, still allowing for potentially long reaction times. The modeled driver reacts instantaneously when the car-following model calculates negative acceleration changes smaller than .

$$(\ ) - \ (\ - \ \ ) < \qquad (18)$$

APs have an additional effect on the model: At standstill, just before drive off, the acceleration can jump to a value as high as . In reality, the jerk is limited by the inertia of the vehicle and the powertrain. This characteristic is incorporated in the EIDM by applying a simple hyperbolic tangent function, thereby introducing a new correction factor ( ) (see Equation (19)) that limits the jerk during drive off. This requires the detection of the time at drive off and the definition of a new parameter: the time duration between drive off and reaching the maximal acceleration.

$$(\ ) = \begin{cases} (\tanh(((\ -\ \ ) * 2/\ \ -\ \ ) *\ \ ) + 1)/2 & \mathrm{t} -\ \ \le \\ 1 & h \end{cases} \qquad (19)$$

The new ( )-function is multiplied with the maximal acceleration , thus better replicating drive off procedures of measured trajectories. Examples showing the correction factor over time for specific parameter sets are plotted in Figure 2. Parameter defines the flatness of the acceleration curve and should optimally take on values between 1.5 and 3. A change in parameter shifts the curve in the direction of the x-axis. Lowering the value results in reaching higher values earlier in time.

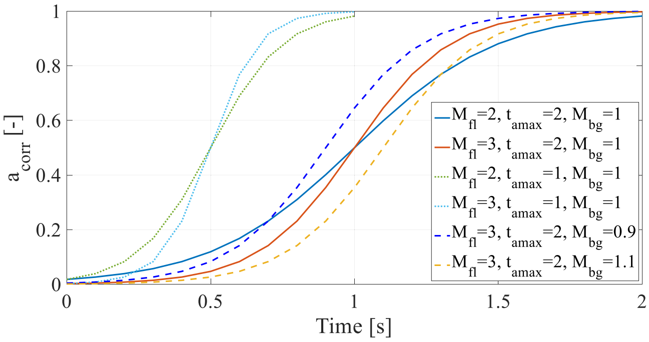


Figure 2: Drive off correction factor

Finally, the decrease of the ratio $^{*}{}_{-1}(\ )/\ (\ )$ is limited. This modification produces more realistic jerks, especially in a simulation environment with junctions, traffic lights and lane changes, where the actual gap $(\ )$ and the desired gap $^{*}{}_{-1}(\ )$ can instantaneously change. These discontinuities can be countered by the following Equation (20), which limits the change of the ratio to a specific magnitude. For $^{*}{}_{-1}(\ ) \geq\ (\ )$ (see Equation (4)), this adjustment guarantees that the jerk of the vehicle never exceeds , which is a freely selectable, positive parameter.

$$\frac{^{*}{}_{-1}(\ )}{(\ )} = \begin{cases} \sqrt{\left(\frac{^{*}{}_{-1}(\ -\Delta\ )}{(\ -\Delta\ )}\right)^{2} - \frac{\Delta\ *}{\ }} & \left(\frac{^{*}{}_{-1}(\ -\Delta\ )}{(\ -\Delta\ )}\right)^{2} - \left(\frac{^{*}{}_{-1}(\ )}{(\ )}\right)^{2} > \frac{\Delta\ *}{\ } \\ \frac{^{*}{}_{-1}(\ )}{(\ )} & h \end{cases} \tag{20}$$

To simulate driving on multi-lane roads, car-following models need to be coupled with a lane change model. This study uses the default lane change model integrated in SUMO (Erdmann 2014). The model is modified for an improved performance with the EIDM. Whether the modifications also improve the operation of the other car-following models, will require further investigation and, therefore, are not proposed in this paper.

4 Case study

In this chapter, real-world trajectories are compared to traffic simulation results generated with SUMO and the EIDM. First, the environment and the specific configuration of the scenario are introduced. The details of the aerial measurements are described, including an accuracy evaluation. The results are then compared to those of the SUMO simulation by means of time headways, speeds and other characteristic values.

4.1 Environment

Figure 3 shows one frame of drone videos recorded on Monday, July 2, 2018, between 07:00 and 07:35. It depicts a junction in Stuttgart, Germany, often referred to as “Neckartor”. The geographical coordinates are approximately 48°47'18.7"N 9°11'28.9"E. The video frame extends 230 m from left to right. The arrows represent the street and direction of traffic considered in this study. The vehicles drove freely after passing the intersection with a speed limit of 50 km/h. The light signal at this junction operated in a fixed-time fashion with a cycle time of 120 s, divided into the following phases in the direction of the arrows: 33 s of green light, 3 s yellow, 24 s red, 34 s green, 3 s yellow and 23 s red. Between red and green phases, the light signals switched to red-yellow for 1 s.

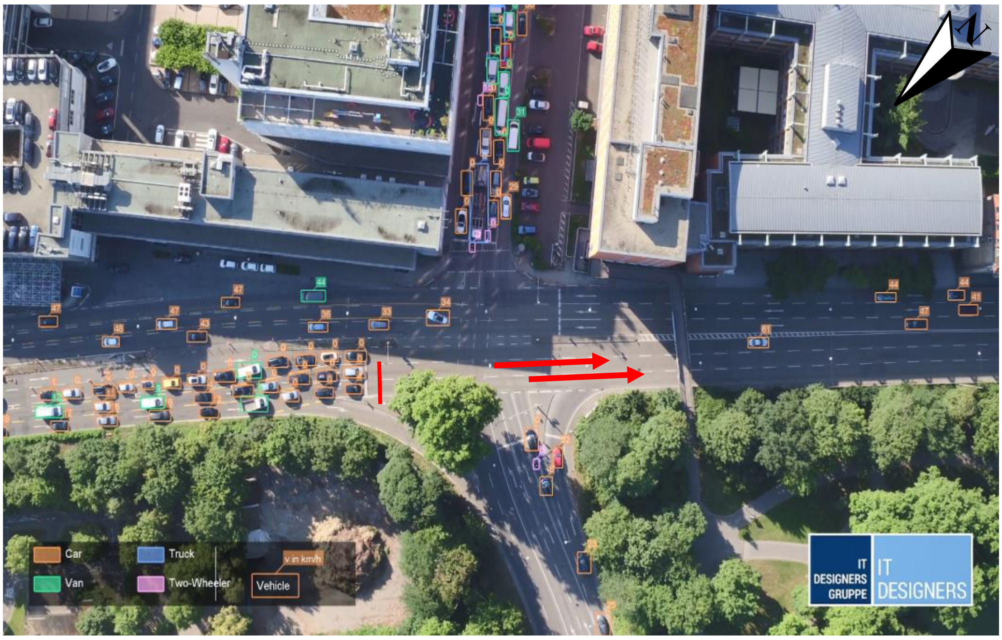


Figure 3: Drone recording of the signalized junction with bounding boxes and vehicle types

Specific configurations and anomalies need to be considered:

- The lanes before the junction are narrower than behind the junction.
- Only phases, when exclusively passenger cars and vans crossed, are analyzed, this results in disregarding three green phases when heavy duty vehicles crossed.

- The red line shown in Figure 3, used as observation reference line, is located approximately 3.5 m to the right of the actual stop line, because some vehicles stopped slightly past it.
- Only the two left lanes are analyzed, because many vehicles in the right lane are temporarily covered by a tree and some also turn right.

In sum, 1050 vehicle trajectories and 540 accelerations starting at 0 km/h were observed (30 green phases consisting of 9 vehicles in 2 lanes each).

4.2 Drone data

The full dataset contains over three hours of video recordings in segments of 7-15 minutes. The dataset used for the evaluation of the car-following model consists of 5 videos, each about 7 minutes long. The videos were recorded with a Zenmuse X5R camera mounted onto a DJI Inspire 1 quadcopter. The camera generated RAW-files with a resolution of 3850p25.

Figure 4 shows the applied traffic measurement process. After the video is recorded, geolocations are referenced at about 10 different reference positions within a single frame. The references are used to define the camera location, which is then tracked for all frames of the recording. The camera tracking algorithm uses automatically detected, well trackable, stationary markers, a method previously used by Kaufmann et al. (2018).

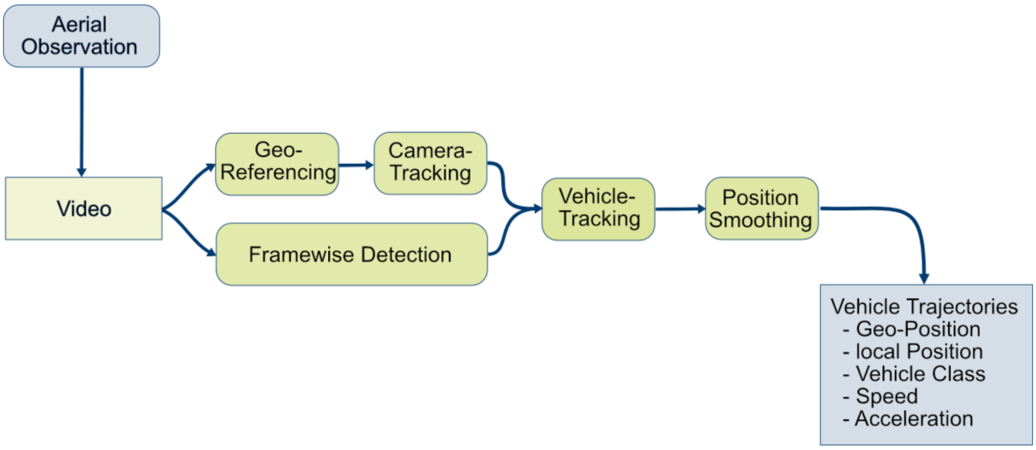


Figure 4: UAV based traffic measurement process

The frame-by-frame vehicle detection uses an artificial neural network for object detection. For this recording, a Faster R-CNN (Ren et al. 2015) with Resnet101 (He et al. 2016) is chosen. Afterwards, the screen positions of the detections are converted into geographic coordinates, using the camera model information and the tracked camera position for each frame.

The tracking algorithm eventually joins the single frame detections to form vehicle trajectories. First, an Intersection over Union (IOU) tracker (Bochinski et al. 2017) combines the detections that closely overlap in successive frames into short vehicle trajectories. Then we use a particle filter to predict the path of these short trajectories and connect them with others. This leads to trajectories that usually cover the entire recording area and correspond to a single vehicle. Some mismatches are adjusted manually.

In order to reduce measurement errors, the vehicle positions are smoothed, after which the velocity is immediately derived. This will be discussed in detail in the next section.

4.3 Measurement accuracy evaluation

The applied measuring method contains two main sources of error: First, the bounding boxes of the same vehicle differ slightly in each frame. This results in a position jitter. Secondly, even the tracking of the stationary reference points causes a small jitter in the camera position, which affects the transformation of the vehicle positions into world coordinates.

For a measurement with 25 frames/s and with a position error , the error propagates to the speed with $= 1/ \quad = 25\,1/\ \Delta$. The acceleration error increases quadratically $\Delta \ = 1/\ ^{2} \quad = 625\ 1/\ ^{2}$. As solution for this problem, we apply an averaging procedure using a moving linear regression (MLR) as shown in Figure 5.

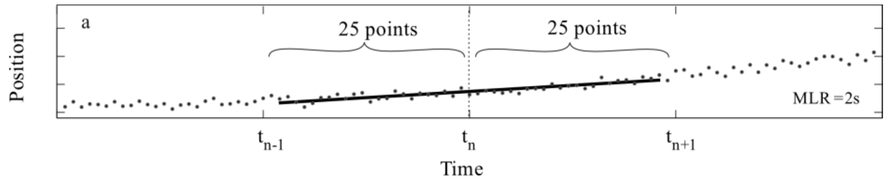


Figure 5: Explanation of the linear regression procedure. With a framerate of 25 fps, the MLR interval is 2 s (Kaufmann et al. 2018).

We take a measurement point related to the time instant . Then, we perform the linear regression for all measurements within a time interval $- \quad \le \ \le \quad +$, where is the time interval duration of the linear regression. In the following, we refer to as the MLR interval.

In order to obtain the vehicle speed $^{2} = \ ^{2} + \ ^{2}$, we first calulate and separatly via the MLR, using the same interval. Afterwards, we use the MLR to obtain the acceleration from the speed using a different interval.

In order to find suitable MLR intervals, a measurement was carried out with a reference vehicle in the MEC-View research project (Gabb et al. 2019). The vehicle was equiped with an Automotive Dynamic Motion Analyzer (ADMA), a highly precise Inertial Measurement Unit (IMU) that uses a Differential Global Positioning System (DGPS). We use the speed measurement and the derived acceleration as the ground truth for a comparison of the smoothing parameters.

Figure 6 shows the first measured scenario: the vehicle stops at an intersection. In the second scenario, in Figure 7, the vehicle drives across an intersection. The second column shows the measurement results using an MLR with an interval of 1 , and the third column shows the results using an MLR with an interval of 2 . The first row shows the measured speed, the second row only the measured ADMA acceleration, the third row shows the acceleration as derivative of the speed, and the last row shows the resulting acceleration using an MLR with an interval of 1 .

The results in Figure 6 show that a speed MLR interval of 1 leads to a strong acceleration error propagation. Only an MLR interval of 2 reduces the error sufficiently, but may sometimes cut off short-term spikes. Nevertheless, we consider the data quality to be quite usable.

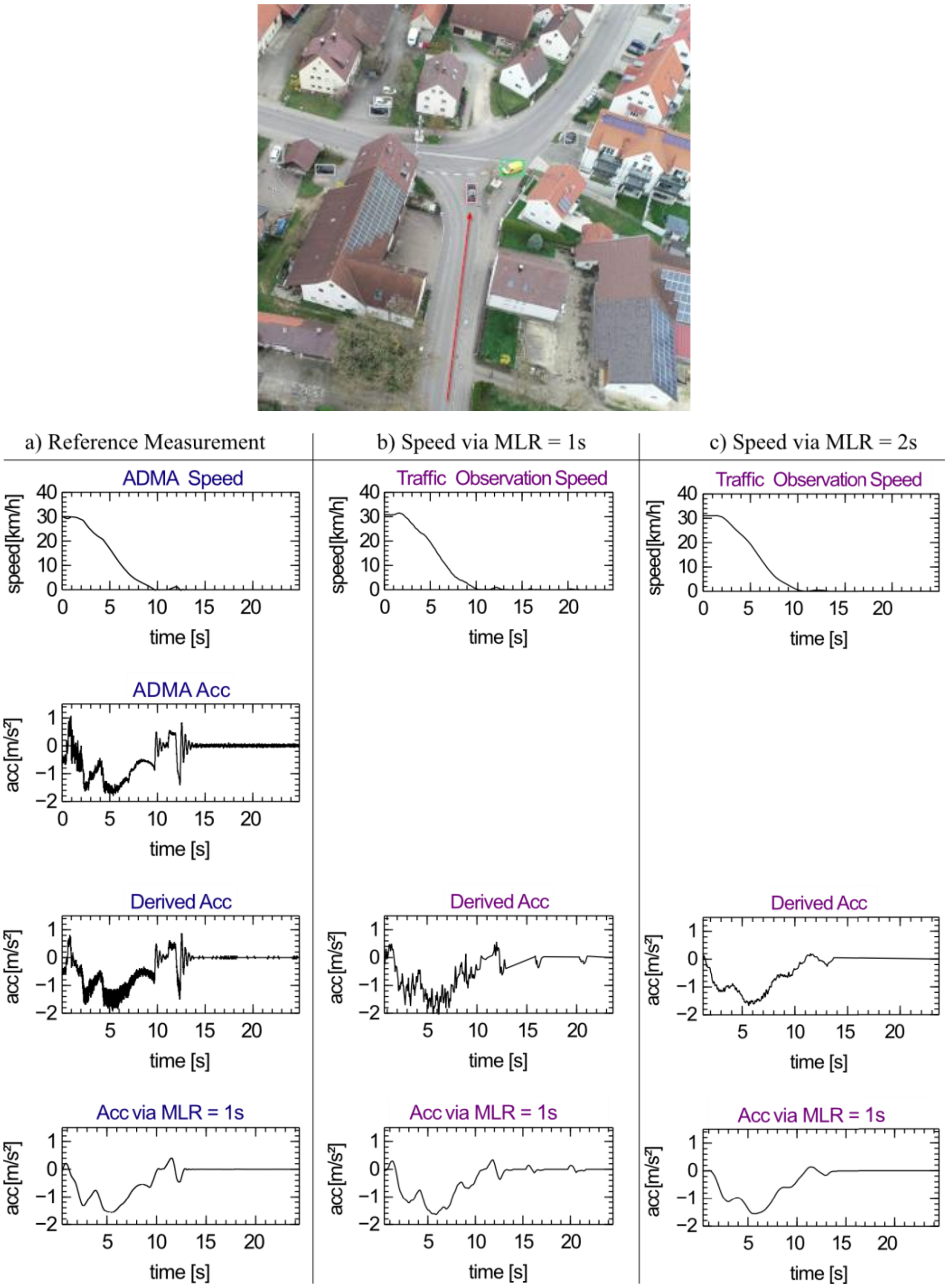


Figure 6: Reference Measurement Scenario 1: Vehicle stops at an intersection. **a)** the ADMA ground truth, **b)** the UAV based measurement with an MLR of 1 s and **c)** the UAV based measurement with an MLR of 2 s

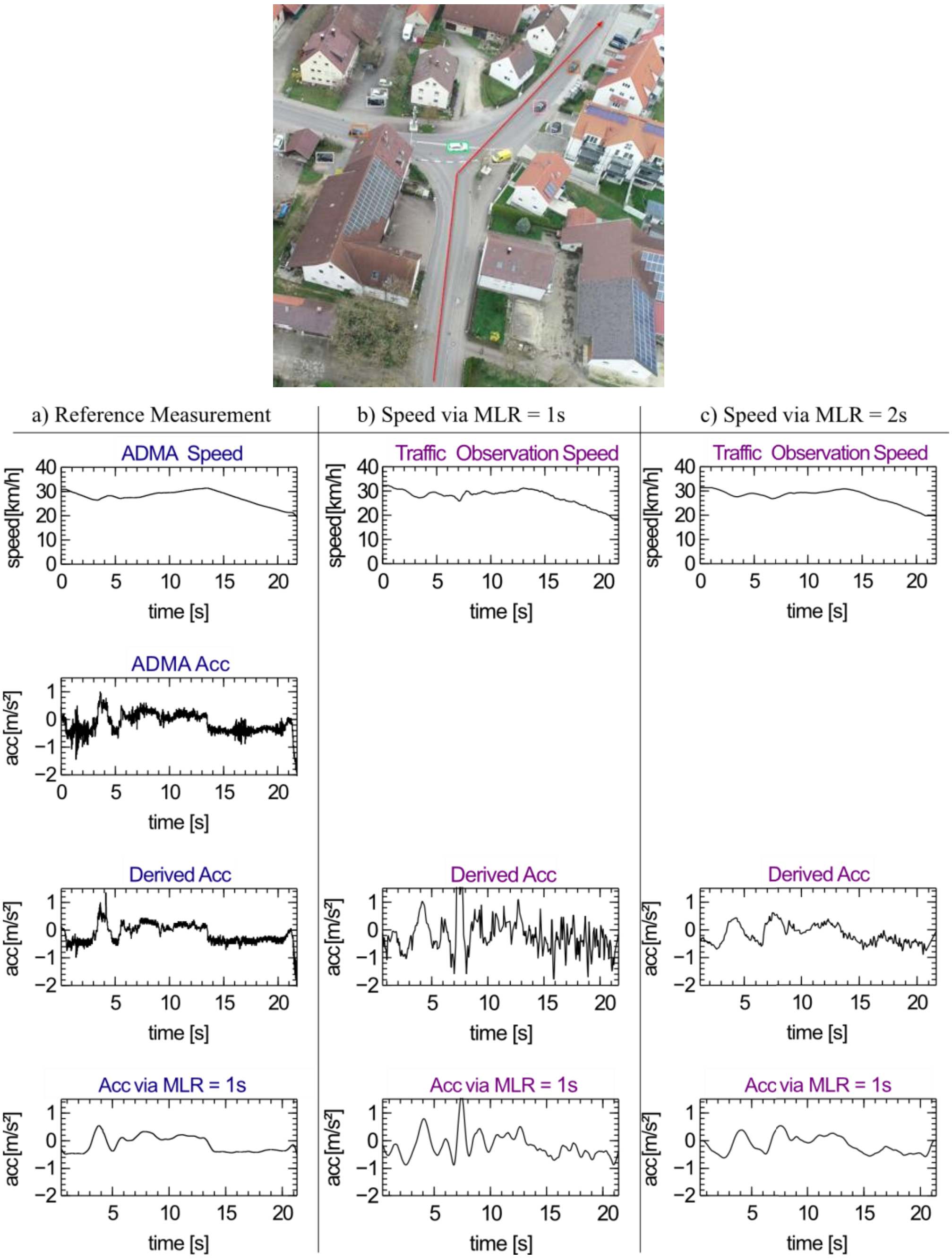


Figure 7: Reference Measurement Scenario 2: Vehicle merges at an Intersection. **a)** the ADMA ground truth, **b)** the UAV based measurement with an MLR of 1 s and **c)** the UAV based measurement with an MLR of 2 s

4.4 Analysis of the drive off trajectories

The aerial measurement method cannot detect the initial movement of the vehicles until they reach an average speed of about 1 m/s. An offset has to be applied to the trajectories in order to remedy this undesirable feature. For this reason, data from multiple vehicle measurements are extracted and used for comparison. The data is provided by the FKFS, where over the past several years many vehicle studies have been carried out, each collecting a vast amount of data at high measurement resolutions. Vehicle, route and driver specifications of the data samples can be found in the literature (Fried 2004; Rumbolz et al. 2010; Wagner et al. 2010).

The data was entirely collected in the broader area of Stuttgart. To identify a comparable offset all start-up procedures that fulfill the following specifications are extracted:

- The vehicle reaches 10 km/h in less than 3 s.
- The vehicle reaches 30 km/h in less than 8 s.
- The vehicle is slower than 60 km/h for the full 10 s of drive off.
- Right before the start-up the vehicle comes to a standstill for at least 2 s.
- The acceleration is between -5 m/s^2 and 5 m/s^2 for the full 10 s of drive off.
- The vehicle has an internal combustion engine.

This specific method resulted in over 2000 comparable drive off trajectories, showing that the vehicles reach 1 m/s after an average time of 0.8 s and their maximal acceleration after an average time of 2.3 s. The average time the vehicles need to reach 1 m/s is then added to every drive off detected by the aerial measurements.

Figure 8a) shows the mean accelerations of the drive off procedures of the first nine vehicles (with offset) after the light signal turns green. A drop and rise of the acceleration between approximately 3 s and 8 s can be recognized. This phenomenon originates from powertrains with manual transmissions. The difference between vehicles with automatic and those with manual transmissions becomes evident when comparing their mean acceleration curves. Figure 9 shows curves from above mentioned measured drive offs of the various vehicle studies. The curves in Figure 8b) and dashed lines in Figure 9 represent the absolute standard deviation of the corresponding acceleration curves, which are approx. ±0.5 m/s^2 over the whole time period except before the acceleration peak is reached and during gear shifting, where the absolute standard deviation is higher because of different gear shifting times and durations of the drivers.

Interestingly Figure 8 reveals that the mean maximal acceleration drops until it reaches a plateau with the fourth and following vehicles. This corresponds to the time headways, shown in Figure 10a), which also drop significantly until the fourth vehicle passes the intersection.

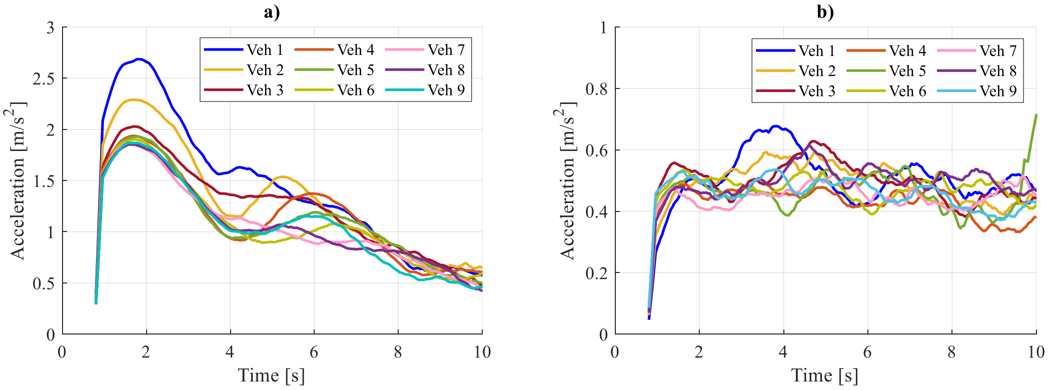


Figure 8: a) Mean acceleration curves of the first 9 vehicles in the queue from the aerial measurement data and **b)** their corresponding absolute standard deviation

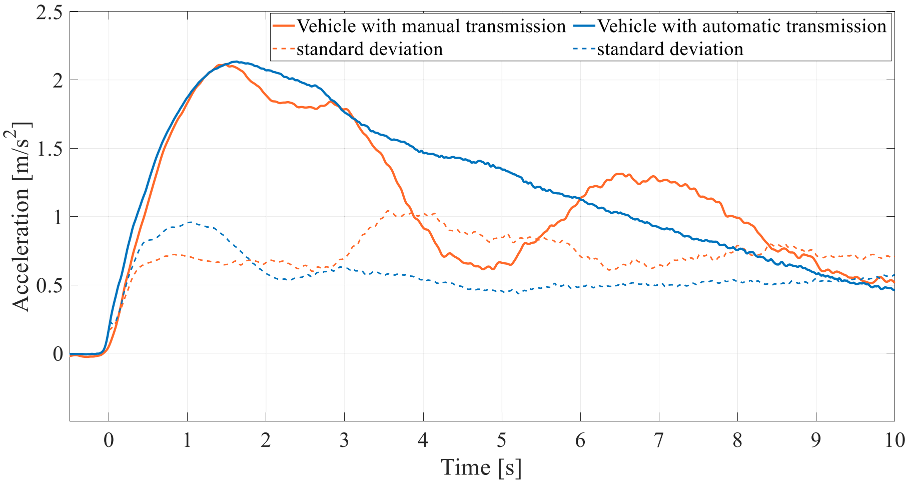


Figure 9: Mean acceleration (solid lines) and absolute standard deviation (dashed lines) curves of vehicles with a manual and an automatic transmission when driving off

4.5 Simulation

To set up the simulation environment delineated in Figure 3, a respective map is extracted from OpenStreetMap and converted to a SUMO network. After manual changes to the map, such as changing the traffic light program to the one used that day and insuring the correct edge and lane configurations, the traffic flow is inserted by defining one flow passing the intersection in the direction of the above mentioned arrows. The flow is defined in a manner to ensure that all vehicles passing the traffic light during a green phase have before come to a complete stop. The simulation duration is identical to that of the real-world measurement (30 green phases in 35 minutes).

Apart from the car-following model, the selection of the specific parameter set for each vehicle has a major effect on the simulation. As an extensive parameter identification is not part of this publication, we run simulations with each of the three car-following models. Every simulation contains four different sets of vehicle parameters, distributed evenly throughout the simulation. Table 2 shows these sets, where parameters existing in each model are varied. Specific parameters for the EIDM, listed in Table 3, are identical for all vehicles and are taken from literature or, in the case of newly introduced parameters, derived from first empirical observations. Lastly, SUMO version 1.0.1 is used with a time step Δ of 0.1 s for the Intelligent Driver Models and 0.5 s for the Krauss Model.

CF-Model	*Length [m]*	*MinGap [m]*			*speedFactor [-]*				*Accel [m/s²]*	*Decel [m/s²]*	*Tau [s]*	*ActionStep [s]*	[-]
		Mean	*Min*	*Max*	*Mean*	*Dev*	*Min*	*Max*					
Krauss Model	3	2.5	2.5	2.5	1.1	0.2	0.9	1.4	2.25	1.75	1.1	0.5	-
	4								2.50	2.00	1		
	4.5								2.75	2.25	0.9		
	5								3.00	2.50	0.8		
Intelligent Driver Model	3	2.5	2.5	2.5	1.1	0.2	0.9	1.4	2.25	2.5	1.1	0.1	4
	4								2.50	2.75	1		
	4.5								2.75	3.00	0.9		
	5								3.00	3.25	0.8		
Extended Intelligent Driver Model	3	2.5	2.0	3.0	1.1	0.2	0.9	1.4	2.40	2.40	1.2	0.5	2
	4								2.70	2.70	1.1	0.5	
	4.5								3.00	3.00	1	0.4	
	5								3.50	3.50	0.9	0.4	

Table 2: Parameter sets for the SUMO simulations

CF-model	*[s]*	~ *[s]*	~ *[s]*	*[-]*	*[-]*	*[-]*	*[-]*	*[s]*	*[-]*	*[-]*	*[m/s³]*
EIDM	4	10	3	0.1	0.02	0.1	0.99	1.2	2	0.7	3

Table 3: Additional parameters for the EIDM

4.6 Comparison between observations and simulations

This chapter provides a first brief comparison between the real-world data and the results of the simulations. As the model parameters are selected empirically, this section does not focus on the absolute differences between the models, but rather on model characteristics and shows the capability of the EIDM to replicate the observed acceleration curves.

Figure 10a) depicts the mean time headway of the simulated and real-world vehicles. The model results generally show good agreement with the observations. With more suitable parameter sets, both the IDM and the Krauss model could replicate the measured headways even more accurately. The advantage of the Extended IDM becomes evident, when we compare the mean speed of the vehicles in Figure 10b). While, in this study, the parameters of the IDM and the Krauss model could not be tuned to reproduce the observed velocity curve, the EIDM reflects the real-world behavior with the chosen parameter set. The velocity curve of the IDM simulation levels off due to its characteristic of never reaching the desired velocity: when increased in an effort to match the curve of the real-world data, the first vehicles drive unrealistically fast. The IIDM resolves this issue, but vehicles still start off fast, reach the saturation speed early and experience sudden acceleration changes when combined with reaction times. The Krauss model can only match the observed speed curve with low acceleration parameters, which stands in contrast to the measured acceleration curves and maximum values.

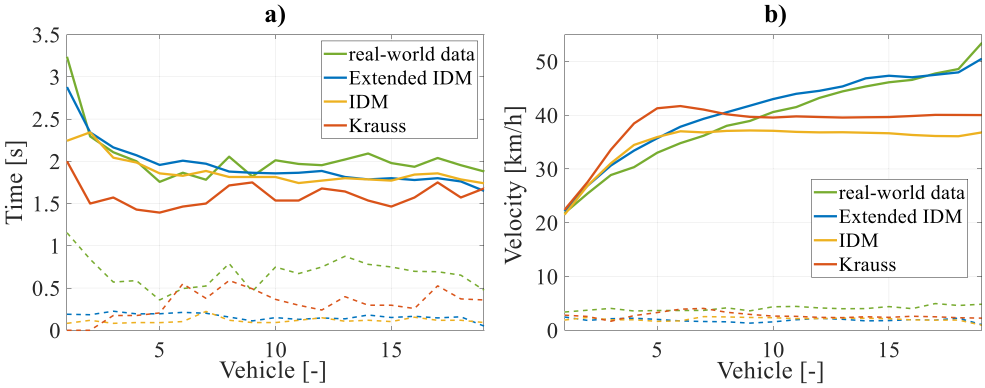


Figure 10: a) Mean time headways between vehicles (solid lines) and the absolute standard deviation of each sample (dashed lines) when passing the reference stop line, **b)** mean speed of vehicles (solid lines) and the absolute standard deviation of each sample (dashed lines) when passing the reference stop line

The combination of IIDM with reaction times and acceleration correction (Equation (19)) produces realistic values with respect to time headways, velocities and accelerations. The acceleration curves of the first 9 vehicles in the different simulations are shown in Figure 11, where each row features the results of the different models. Column a) shows the mean acceleration curves, b) the absolute standard deviation of each sample and c) the Mean Bias Error (MBE) between the model and the real-world measurement. The Mean Absolute Error (MAE) between the mean acceleration curves is calculated seperatly for each vehicle position.

This first analysis of the EIDM shows more accurate acceleration trajectories compared with the IDM and the Krauss model. However, the results of the Krauss model improve significantly when we cut off the first acceleration peak, illustrated in Figure 11a) third row, and delay the drive off to a later point in time.

The EIDM possesses a unique characteristic in that the mean maximal acceleration drops until it reaches a plateau with approximately the fourth vehicle. Still, the observed vehicles reach their maximal acceleration later than those modeled by the EIDM. Consequently, the parameters of the correction term need to be slighty adjusted. Increasing the amount and distribution of the parameter sets can further improve the model.

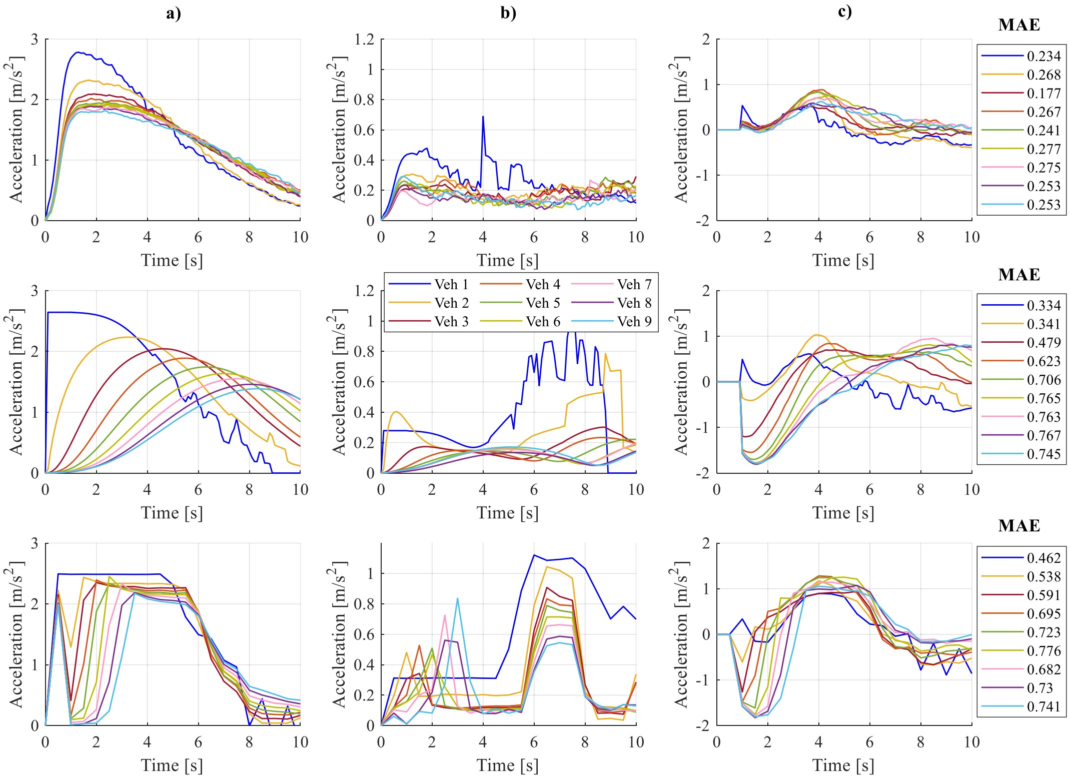


Figure 11: a) Mean acceleration curves of the first 9 vehicles in the queue from the simulation with the EIDM (first row), the IDM (second row) and the Krauss Model (third row), **b)** the corresponding absolute standard deviation of each sample, **c)** the Mean Bias Error between the real-world mean accelerations of the queued vehicles and the simulation results, depicted with their respected Mean Absolute Error values over the first 9 s of drive off (disregarding the first second, as the movement can't be correctly detected there)

In the past, different parameter identification techniques have been introduced to calibrate the IDM using floating car data, but that usually results in low values, which do not agree with the data presented in this study.

Table 4 lists such calibrated parameter sets.

Kovács et al. (2016) calibrate the IDM to obtain accurate flow capacities and time headways at signalized intersections. They use a reliable method to calculate the IDM parameters based on the time headway saturation. On the negative side, the resulting value and the desired time headway are rather low for urban traffic flow (Schulz 2013).

In summary, there are many models that can accurately describe time headways and speeds at intersections (Le Vine et al. 2016; Dey et al. 2013; James A. Bonneson; Li and Chen 2017; Jumsan KIM et al. 2005). But they can generally only calculate the time headway, velocity and acceleration at specific points and do not generate trajectories as car-following models can.

Parameter	EIDM	Kovács et al. (2016)	Kesting and Treiber (2008)	Treiber and Kesting (2013b)	Dallmeyer (2014)	Treiber and Kesting (2013a)
[m/s^2]	2.5-3.5	1.6	1.5-1.6	1.0	1.5	1.4
$\bar{}_0$ [m/s]	13.89	15.28	16.1	15	13.89	16.1
[s]	1.1-1.3	0.86	1.3-1.4	1.0	1.5	1.2
$_0$ [m]	1.5-2.5	2	1-1.6	2.0	2.0	1.5
[m/s^2]	2.5-3.5	5	0.6-0.75	1.5	2.0	0.65
[-]	2	4	-	4	4	-

Table 4: IDM parameters used in literature compared to those in this study

5 Discussion and conclusions

In this paper we extended the IDM in SUMO to replicate human driver behavior. First, previously published IDM enhancements were introduced and discussed, as they have not yet been integrated in SUMO. Additional enhancements of the IDM to account for discontinuities in SUMO's time-discrete simulation were presented, with a special emphasis on the drive off process.

The drive off of the EIDM was validated using a drone dataset. The vehicle trajectories of the drone video were extracted using a Faster R-CNN with Reset101. The specific measurement method was outlined and the accuracy of the method was evaluated. The errors were determined to be negligibly small.

Nevertheless, the car-following model was only evaluated with a small amount of data in a specific environment: Drive off procedures at a saturated junction. The model will still need to prove that it can realistically reproduce other driving situations. In addition, only mean and characteristic values were used to compare the observed data with the simulation. The parameter set in this study is not comprehensive enough to replicate the standard deviations and typical log-normal distributions (Jin et al. 2009) during drive off. A detailed parameter identification for each vehicle could lead to a better agreement of modeled and observed behavior. By introducing the new drive off equation, the parameters identifying the drive off process can be separated from those characterizing vehicle following situations.

Additional variables could further increase the precision and influence of the individual driver by including road slope (Schulz 2013), spatial anticipation (Chen et al. 2009) or using action points dependent on the driving situation. Another factor to be considered are aerodynamics, which limit the maximal acceleration at high speeds and could explain why the IDM-parameter a_{max} is small when calibrating the model using highway traffic data.

6 Future work

Future work includes the integration of SUMO into an Unreal Engine driving simulator environment. Such linking has already been performed using Unity3D (Biurrun-Quel et al. 2017). The ultimate objective is the integration of simulative constructed drive trains (Ebel et al. 2017) into the environment, as done in DYNA4 (Kaths et al. 2019) and by Riegl et al. (2019). Additional research is needed to investigate the positive effects of more accurate acceleration patterns on energy consumption and emission calculations.

The EIDM will benefit from additional development with respect to cooperative lane changes. We plan to integrate the model into the current SUMO version.

7 Acknowledgment

This study was partially funded by the Robert Bosch GmbH. The authors would like to thank Helix4Motion for providing the drone videos and especially the German Aerospace Center and all SUMO contributors for creating such a versatile Open Source mobility platform.

References

Alazzawi, Sabina; Hummel, Mathias; Kordt, Pascal; Sickenberger, Thorsten; Wieseotte, Christian; Wohak, Oliver (2018): Simulating the Impact of Shared, Autonomous Vehicles on Urban Mobility – a Case Study of Milan. In. SUMO 2018- Simulating Autonomous and Intermodal Transport Systems: EasyChair (EPiC Series in Engineering), 94-76.

Alekszejenkó, Levente; Dobrowiecki, Tadeusz P. (2019): SUMO Based Platform for Cooperative Intelligent Automotive Agents. In. SUMO User Conference 2019: EasyChair (EPiC Series in Computing), pp. 107–189.

Bando; Hasebe; Nakayama; Shibata; Sugiyama (1995): Dynamical model of traffic congestion and numerical simulation. In *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics* 51 (2), pp. 1035–1042. DOI: 10.1103/PhysRevE.51.1035.

Biurrun-Quel, Carlos; Serrano-Arriezu, Luis; Olaverri-Monreal, Cristina (2017): Microscopic Driver-Centric Simulator: Linking Unity3D and SUMO. In Álvaro Rocha, Ana Maria Correia, Hojjat Adeli, Luís Paulo Reis, Sandra Costanzo (Eds.): Recent Advances in Information Systems and Technologies, vol. 569. Cham: Springer International Publishing (Advances in Intelligent Systems and Computing), pp. 851–860.

Bochinski, Erik; Eiselein, Volker; Sikora, Thomas (2017): High-Speed tracking-by-detection without using image information. In. 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). Lecce, Italy, 2017: IEEE, pp. 1–6.

Bock, Julian; Krajewski, Robert; Moers, Tobias; Runde, Steffen; Vater, Lennart; Eckstein, Lutz (2019): The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections. Available online at http://arxiv.org/pdf/1911.07602v1.

Chen, Xiqun; Li, Ruimin; Xie, Weijun; Shi, Qixin (2009): Stabilization of traffic flow based on multi-anticipative intelligent driver model. In. 12th International IEEE Conference on Intelligent Transportation Systems (ITSC): IEEE, pp. 1–6.

Coifman, Benjamin; Li, Lizhe (2017): A critical evaluation of the Next Generation Simulation (NGSIM) vehicle trajectory dataset. In *Transportation Research Part B: Methodological* 105, pp. 362–377. DOI: 10.1016/j.trb.2017.09.018.

Dallmeyer, Jörg (2014): Simulation des Straßenverkehrs in der Großstadt. Das Mit- und Gegeneinander verschiedener Verkehrsteilnehmertypen. Teilw. zugl.: Frankfurt, Univ., Diss., 2013 u.d.T.: Akteursorientierte multimodale Straßenverkehrssimulation. Wiesbaden: Springer Vieweg (Research).

Derbel, Oussama; Mourllion, Benjamin; Basset, Michel (2012): Extended safety descriptor measurements for relative safety assessment in mixed road traffic. In. 15th International IEEE Conference on Intelligent Transportation Systems - (ITSC 2012): IEEE, pp. 752–757.

Dey, Partha; Nandal, S.; Kalyan, R. (2013): Queue discharge characteristics at signalised intersections under mixed traffic conditions. In *European Transport - Trasporti Europei*.

Donateo, Teresa; Pacella, Damiano; Laforgia, Domenico (2010): Simulation and Optimization of the Energy Management of ITAN500 in the SUMO Traffic Model Environment. In.

Ebel, André; Baumgartner, Edwin; Orner, Markus; Reuss, Hans-Christian (2017): Bewertung simulativ ausgelegter Antriebsstränge am Stuttgarter Fahrsimulator. In *MTZ Extra* 22 (S1), pp. 40–43. DOI: 10.1007/s41490-017-0004-9.

Ejercito, Paolo M.; Nebrija, Kristine Gayle E.; Feria, Rommel P.; Lara-Figueroa, Ligaya Leah (2017): Traffic simulation software review. In. 8th International Conference on Information, Intelligence, Systems & Applications (IISA): IEEE, pp. 1–4.

Erdağı, İsmet Gökşad; Silgu, Mehmet Ali; Çelikoğlu, Hilmi Berk (2019): Emission Effects of Cooperative Adaptive Cruise Control: A Simulation Case Using SUMO. In. SUMO User Conference 2019: EasyChair (EPiC Series in Computing), 92-82.

Erdmann, Jakob (2014): Lane-changing model in SUMO. In, vol. 24.

Fried, Oliver (2004): Betriebsstrategie für einen Minimalhybrid-Antriebsstrang. Aachen: Shaker (Berichte aus der Fahrzeugtechnik).

Gabb, Michael; Digel, Holger; Muller, Tobias; Henn, Rudiger-Walter (2019): Infrastructure-supported Perception and Track-level Fusion using Edge Computing. In. IEEE Intelligent Vehicles Symposium (IV). Paris, France, 2019: IEEE, pp. 1739–1745.

Gipps, P. G. (1981): A behavioural car-following model for computer simulation. In *Transportation Research Part B: Methodological* 15 (2), pp. 105–111. DOI: 10.1016/0191-2615(81)90037-0.

Grumert, Ellen; Ma, Xiaoliang; Tapani, Andreas (2015): Analysis of a cooperative variable speed limit system using microscopic traffic simulation. In *Transportation Research Part C: Emerging Technologies* 52, pp. 173–186. DOI: 10.1016/j.trc.2014.11.004.

He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2016): Deep Residual Learning for Image Recognition. In. IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, pp. 770–778.

James A. Bonneson: Modeling Queued Driver Behavior at Signalized Junctions. In : Transportation Research Record, vol. 1365, pp. 99–107.

Jiang, R.; Wu, Q.; Zhu, Z. (2001): Full velocity difference model for a car-following theory. In *Physical review. E, Statistical, nonlinear, and soft matter physics* 64 (1 Pt 2), p. 17101. DOI: 10.1103/PhysRevE.64.017101.

Jin, Xuexiang; Zhang, Yi; Wang, Fa; Li, Li; Yao, Danya; Su, Yuelong; Wei, Zheng (2009): Departure headways at signalized intersections: A log-normal distribution model approach. In *Transportation Research Part C: Emerging Technologies* 17 (3), pp. 318–327. DOI: 10.1016/j.trc.2009.01.003.

Jumsan KIM; Zunhwan HWANG; Sungmo RHEE (2005): Vehicle Passing Behavior Through The Stop Line of Signalized Intersection. In *Journal of the Eastern Asia Society for Transportation Studies* 6, pp. 1509–1517. DOI: 10.11175/easts.6.1509.

Kaths, Jakob; Schott, Benedikt; Chucholowski, Frederic (2019): Co-simulation of the virtual vehicle in virtual traffic considering tactical driver decisions. In. SUMO User Conference 2019: EasyChair (EPiC Series in Computing), 21-12.

Kaufmann, Stefan; Kerner, Boris S.; Rehborn, Hubert; Koller, Micha; Klenov, Sergey L. (2018): Aerial observations of moving synchronized flow patterns in over-saturated city traffic. In *Transportation Research Part C: Emerging Technologies* 86, pp. 393–406. DOI: 10.1016/j.trc.2017.11.024.

Kesting, Arne; Treiber, Martin (2008): Calibrating Car-Following Models by Using Trajectory Data. In *Transportation Research Record* 2088 (1), pp. 148–156. DOI: 10.3141/2088-16.

Kesting, Arne; Treiber, Martin; Helbing, Dirk (2010): Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity. In *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 368 (1928), pp. 4585–4605. DOI: 10.1098/rsta.2010.0084.

Kotusevski, G.; Hawick, Ken (2009): A Review of Traffic Simulation Software. In *Res. Lett. Inf. Math. Sci.* 13.

Kovács, Tamás; Bolla, Kálmán; Gil, Rafael Alvarez; Csizmás, Edit; Fábián, Csaba; Kovács, Lóránt et al. (2016): Parameters of the intelligent driver model in signalized intersections. In *Teh. vjesn.* 23 (5). DOI: 10.17559/TV-20140702174255.

Kovvali, Vijay Gopal; Alexiadis, Vassili; Zhang P.E., Lin: Video-Based Vehicle Trajectory Data Collection. In : Transportation Research Board 86th Annual Meeting.

Krajewski, Robert; Bock, Julian; Kloeker, Laurent; Eckstein, Lutz (2018): The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems. In. 21st International Conference on Intelligent Transportation Systems (ITSC): IEEE, pp. 2118–2125.

Krauss, S.; Wagner, P.; Gawron, C. (1997): Metastable states in a microscopic model of traffic flow. In *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics* 55 (5), pp. 5597–5602. DOI: 10.1103/PhysRevE.55.5597.

Le Vine, Scott; Liu, Xiaobo; Zheng, Fangfang; Polak, John (2016): Automated cars: Queue discharge at signalized intersections with 'Assured-Clear-Distance-Ahead' driving strategies. In *Transportation Research Part C: Emerging Technologies* 62, pp. 35–54. DOI: 10.1016/j.trc.2015.11.005.

Li, Li; Chen, Xiqun (2017): Vehicle headway modeling and its inferences in macroscopic/microscopic traffic flow theory: A survey. In *Transportation Research Part C: Emerging Technologies* 76, pp. 170–188. DOI: 10.1016/j.trc.2017.01.007.

Lopez, Pablo Alvarez; Wiessner, Evamarie; Behrisch, Michael; Bieker-Walz, Laura; Erdmann, Jakob; Flotterod, Yun-Pang et al. (2018): Microscopic Traffic Simulation using SUMO. In. 21st International Conference on Intelligent Transportation Systems (ITSC): IEEE, pp. 2575–2582.

Macedo, Jose; Kokkinogenis, Zafeiris; Soares, Guilherme; Perrotta, Deborah; Rossetti, Rosaldo J. F. (2013): A HLA-based multi-resolution approach to simulating electric vehicles in simulink and SUMO. In. 16th International IEEE Conference on Intelligent Transportation Systems - (ITSC 2013): IEEE, pp. 2367–2372.

Milanés, Vicente; Shladover, Steven E. (2014): Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data. In *Transportation Research Part C: Emerging Technologies* 48, pp. 285–300. DOI: 10.1016/j.trc.2014.09.001.

Nagel, Kai; Schreckenberg, Michael (1992): A cellular automaton model for freeway traffic. In *J. Phys. I France* 2 (12), pp. 2221–2229. DOI: 10.1051/jp1:1992277.

Pfeil, Raphael (2019): Methodischer Ansatz zur Optimierung von Energieladestrategien für elektrisch angetriebene Fahrzeuge (Wissenschaftliche Reihe Fahrzeugtechnik Universität Stuttgart). Available online at https://doi.org/10.1007/978-3-658-25863-4.

Queck, Tobias; Schüenemann, Bjöern; Radusch, Ilja (2008): Runtime infrastructure for simulating vehicle-2-x communication scenarios. In Varsha Sadekar, Paolo Santi, Yih-Chun Hu, Martin Mauve (Eds.): Proceedings of the fifth ACM international workshop on VehiculAr Inter-NETworking - VANET '08. the fifth ACM international workshop. San Francisco, California, USA, 15.09.2008 - 15.09.2008. New York, New York, USA: ACM Press, p. 78.

Ren, Shaoqing; He, Kaiming; Girshick, Ross; Sun, Jian (2015): Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Available online at http://arxiv.org/pdf/1506.01497v3.

Richter, Gerald; Grohmann, Lukas; Nitsche, Philippe; Lenz, Gernot (2019): Anticipating Automated Vehicle Presence and the Effects on Interactions with Conventional Traffic and Infrastructure. In. SUMO User Conference 2019: EasyChair (EPiC Series in Computing), 230-215.

Riegl, Peter; Gaull, Andreas; Beitelschmidt, Michael (2019): A tool chain for generating critical traffic situations for testing vehicle safety functions. In. IEEE International Conference on Vehicular Electronics and Safety (ICVES): IEEE, pp. 1–6.

Rumbolz, Philip; Piegssa, Anne; Reuss, Hans-Christian (2010): Messung der Fahrzeug-internen Leistungsflüsse und der diese beeinflussenden Größen im 'real-life' Fahrbetrieb. In *VDI-Berichte 2105*, pp. 175–188.

Saifuzzaman, Mohammad; Zheng, Zuduo (2014): Incorporating human-factors in car-following models: A review of recent developments and research needs. In *Transportation Research Part C: Emerging Technologies* 48, pp. 379–403. DOI: 10.1016/j.trc.2014.09.008.

Schulz, Ralph (2013): Blickverhalten und Orientierung von Kraftfahrern auf Landstraßen. 1. Aufl. Dresden: TU, Lehrstuhl Gestaltung von Straßenverkehrsanlagen (Schriftenreihe des Lehrstuhls Gestaltung von Straßenverkehrsanlagen, H. 10).

Sommer, Christoph; Yao, Zheng; German, Reinhard; Dressler, Falko (2008): On the need for bidirectional coupling of road traffic microsimulation and network simulation. In Minkyong Kim, Cecilia Mascolo, Mirco Musolesi (Eds.): Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models - MobilityModels '08. Proceeding of the 1st ACM SIGMOBILE workshop. Hong Kong, Hong Kong, China, 26.05.2008 - 26.05.2008. New York, New York, USA: ACM Press, p. 41.

Thiemann, Christian; Treiber, Martin; Kesting, Arne (2008): Estimating Acceleration and Lane-Changing Dynamics from Next Generation Simulation Trajectory Data. In *Transportation Research Record* 2088 (1), pp. 90–101. DOI: 10.3141/2088-10.

Treiber; Hennecke; Helbing (2000): Congested traffic states in empirical observations and microscopic simulations. In *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics* 62 (2 Pt A), pp. 1805–1824. DOI: 10.1103/PhysRevE.62.1805.

Treiber, Martin; Helbing, Dirk (2004): Visualisierung der fahrzeugbezogenen und verkehrlichen Dynamik mit und ohne Beeinflussungs-Systemen. In, pp. 323–334.

Treiber, Martin; Kanagaraj, Venkatesan (2015): Comparing numerical integration schemes for time-continuous car-following models. In *Physica A: Statistical Mechanics and its Applications* 419, pp. 183–195. DOI: 10.1016/j.physa.2014.09.061.

Treiber, Martin; Kesting, Arne (2013a): Microscopic Calibration and Validation of Car-Following Models – A Systematic Approach. In *Procedia - Social and Behavioral Sciences* 80, pp. 922–939. DOI: 10.1016/j.sbspro.2013.05.050.

Treiber, Martin; Kesting, Arne (2013b): Traffic flow dynamics. Data, models and simulation. Heidelberg, New York: Springer.

Treiber, Martin; Kesting, Arne (2017): The Intelligent Driver Model with Stochasticity -New Insights Into Traffic Flow Oscillations. In *Transportation Research Procedia* 23, pp. 174–187. DOI: 10.1016/j.trpro.2017.05.011.

Treiber, Martin; Kesting, Arne; Helbing, Dirk (2006): Delays, inaccuracies and anticipation in microscopic traffic models. In *Physica A: Statistical Mechanics and its Applications* 360 (1), pp. 71–88. DOI: 10.1016/j.physa.2005.05.001.

Wagner, C.; Salfeld, M.; Knoll, S.; Reuss, H.-C. (2010): Quantifizierung des Einflusses von ACC auf die CO2-Emissionen im kundenrelevanten Fahrbetrieb. In *Proceedings Stuttgart International Symposium, Nr. 10*.

Wagner, Peter; Lubashevsky, Ihor (2003): Empirical basis for car-following theory development.

Wegener, Axel; Piórkowski, Michał; Raya, Maxim; Hellbrück, Horst; Fischer, Stefan; Hubaux, Jean-Pierre (2008): TraCI: an interface for coupling road traffic and network simulators. In Aftab Ahmad, Arnold Bragg (Eds.): Proceedings of the 11th communications and networking simulation symposium on - CNS '08. the 11th communications and networking simulation symposium. Ottawa, Canada, 14.04.2008 - 17.04.2008. New York, New York, USA: ACM Press, p. 155.

Wiedemann, Rainer (1974): Simulation des Straßenverkehrsflusses. Hochschulschrift, Karlsruhe.

Xiao, Lin; Wang, Meng; Schakel, Wouter; van Arem, Bart (2018): Unravelling effects of cooperative adaptive cruise control deactivation on traffic flow characteristics at merging bottlenecks. In *Transportation Research Part C: Emerging Technologies* 96, pp. 380–397. DOI: 10.1016/j.trc.2018.10.008.

Xiao, Lin; Wang, Meng; van Arem, Bart (2017): Realistic Car-Following Models for Microscopic Simulation of Adaptive and Cooperative Adaptive Cruise Control Vehicles. In *Transportation Research Record* 2623 (1), pp. 1–9. DOI: 10.3141/2623-01.

Zhou, Mofan; Qu, Xiaobo; Jin, Sheng (2016): On the Impact of Cooperative Autonomous Vehicles in Improving Freeway Merging: A Modified Intelligent Driver Model-Based Approach. In *IEEE Trans. Intell. Transport. Syst.*, pp. 1–7. DOI: 10.1109/TITS.2016.2606492.

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.96

Published: 28 Jun. 2022

Estimation of Green House Gas and Contaminant Emissions from Traffic by microsimulation and refined Origin-Destination matrices: a methodological approach

Jorge E. Luzuriaga, Juan A. Moreno, Edgar Lorenzo-Sáez, Santiago Mira Prats, Javier F. Urchueguia, Lenin G. Lemus, J.V. Oliver-Villanueva, and Miguel A. Mateo Pla

ITACA Institute - (http://www.itaca.upv.es),
ICT against Climate Change Research Group (ictvscc.webs.upv.es/en),
Cami de Vera s/n, 46022 València - Spain
Corresponding author: jorluqui@upvnet.upv.es

Keywords: mobility, demand modelling, OD matrices, CO2 emissions, SUMO, Lagrangian algorithm.

Abstract

The high levels of air contamination and presence of different pollutants are a large problem in most of the cities in which road transport is the primary source of emissions. The governments of more than 100 countries are adopting different policies and strategies to help reduce and mitigate their global emissions. In terms of road transport, reductions in emissions could be achieved by replacing conventional vehicle technologies or by chang-ing the travel patterns of individuals using a private vehicle as their primary means of transportation. However, accurately quantifying the emissions related to the urban traffic from multiple possible scenarios is a very complicated task, even when appropriate tools made for this purpose are available. Here we apply a scientifically rigorous protocol to accurately estimate greenhouse and other polluting gases. We describe the methodological steps we followed to analyse the vast quantities of data available from different heteroge-neous sources. This data can aid decision-makers in planning better strategies for urban transportation. We used the origin-destination matrices already available for Valencia city (Spain), as well as historical information for their street induction-loops and the phases and times of their traffic light system as our input data for the traffic model. Rather than a brute-force algorithm, we used a fast-convergence Lagrangian algorithm model which deals with that vast quantities of information. Based on the elements mentioned above together with the statistics about the types of vehicles in the city by simulations the urban mobility city's traffic was reconstructed at different times to quantify the emissions produced with a high spatial and temporal resolution.

1 Introduction

Interactions between the demand and supply of transport determine traffic flows on road networks. Vehicles consume fuel and in turn, the emissions produced by these mobile sources determine the concentration of pollutants in the air. Air and noise pollution have been identified as among the most critical environmental problems present in urban areas. The importance of environmental issues in the quality of life of the population lies in health problems related to pollution. These include physiological and psychological disorders and their severity depends on the levels and extent of exposure.

As measures to mitigate the different environmental problems caused by urban traffic, various strategies have been implemented (usually by city councils), starting by defining the origins of the discomfort the population experiences in relation to these problems. These include modifying vehicle manufacturing and circulation regulations, placing barriers around highways and urban centres, careful new road developments, replacement of the types of vehicles used in fleets, and the creation of new massive parking spaces. Nevertheless, these measures are usually based on decisions not supported by scientific data or technical criteria and so, often end up increasing traffic and causing mobility problems.

In this sense, our work highlights the essential role of multivariate statistics and mathematical models in the analysis of information obtained from large amounts of data. These analyses allow us to define the significance of the observed findings and can highlight elements that may not be self-evident. The knowledge achieved through this type of analysis provides public decision-makers with a robust methodology for quantifying emissions/pollution which can help them to choose appropriate urban mobility plan strategies to mitigate the impact of transport on the environmental quality of cities.

Nonetheless, the available data only indicates the number of detections—i.e., it tells us that several vehicles have passed through a section. However, the vehicle speeds, characteristics, and trajectories remain unknown. Thus, simulations which use extrapolation-like methods are required to try to leverage this data to build a complex picture of the situation and obtain useful information from it. This is not a simple extrapolation in the form of a graphical curve, it is a much more sophisticated extrapolated reconstruction based on a mathematical model.

As a use case for this methodology, we examined the CO_2 emissions produced in the city of Valencia (Spain). Even though data related to urban traffic in this city is available, it cannot be extrapolated with sufficient accuracy to be able to make inferences and estimations based on it with any scientific rigour. Our approach creates a traffic model that uses as the input data (1) the origin–destination (OD) matrices already available for the city, (2) historical data from the induction loop detectors of the city's streets, and (3) the city's traffic light system regulation phases and times. Based on these elements, the mathematical model based in a Lagrangian algorithm deals with that vast quantity of information and results in a fast convergence. We then use the statistics about the types of vehicles in the city and by simulations of urban mobility the city's traffic flow is reconstructed at different times to quantify the emissions produced.

The rest of the paper is organised as follows: a review of relative works is presented in section 2. Material and Methods sections present scenario components and the steps we follow are presented in section 3 and 4 respectively. The reached results are presented in section 5. The paper finalises with the "Further research" and "Conclusions" sections.

2 Related Works

Nowadays, due to climate change and the challenges facing the environment, countries need to find solutions that reduce global warming pollution, reduce the use of fossil fuels and focus on clean energy sources. These solutions could affect mobility in different ways and planners should know whatever those effects are. Below is presented a few works that we can find in the related literature. The quantification of emissions produced by the transport sector that are specified in the Intergovernmental Panel on Climate Change (IPCC) reports could be improved by the developing of sectorial emission models of atmospheric pollutants [13].

Authors in [2] present a comparison of estimated against real truth produced emissions through simulations. This study is based on broadcasted information by equipped vehicles with an special communication devices when they passed over an induction loop in an small scenario composed just by one intersection.

In [14] authors analyse the behaviour of all the streets in the Valencia city to determine based on loads of vehicles on the streets what will be journey time and their occupancy. The mathematical model characterises all the main roads and city streets. The model is used by a centralised server that receives all the requests that arrive from each car to make balanced traffic management and predict future occupancy. In this way, the street/road occupation is already known, and in high occupancy situation, all the new requests will be sent to other areas to avoid traffic congestion problems.

In [4] the total CO_2 emissions generated in the metropolitan area of a city in Taiwan is analysed. Authors made a statistical analysis of the traffic volume in an entire year according to peak traffic hours and the development of several buildings. They obtain a prediction formula used to forecast the development scale of various buildings and the information of the road system and traffic volume is presented as hot maps.

In [9] the acquisition of an OD matrix according to the existing traffic in real–time is proposed. The resulted matrix is configured using a Lagrangian optimization algorithm with restrictions on the components of the initial matrix, using vehicular flow data at specific points in the road network.

However, the emission quantification's that comes from the IPCC reports are merely a table extrapolation that may not be the most appropriate approach for some particular areas where a certain level of accuracy is needed. Regarding to the other works, first [14],[9] the issue of emissions is not considered. Then, [4] where emissions are obtained through a formula or model lack of a simulation part to counteract their results and finally, [2] where communication V2I does not fit the reality at least in the city under study. Thus the problem is in a certain way analysed without approaching it as is presented in the methodology of this document.

3 Material

3.1 Simulator

For the microsimulation we have chosen the Simulation of Urban MObility (SUMO) [12] because it is an open system which allows simulating the dynamics and interactions of almost all the elements that build mobility in a city. In addition, it has a large set of tools for the simulation's scenarios creation moreover of a big a development community behind it [11]. In this work we have used the Eclipse SUMO 1.5.0 versions.

3.2 Traffic Network

The city of Valencia with more than 800,000 inhabitants, currently has a vehicle fleet of approximately 498,000 vehicles and road network of 300 km long. To optimize the general traffic conditions for all the agents involved in the urban traffic of the city (pedestrians, private vehicles, collective transport, police, media, etc.), the Centre of City Traffic Management of Valencia carries out comprehensive traffic management.

The traffic status of the city road network is known in real-time through two information sources: (a) the detectors installed in the traffic lanes and (b) Closed Circuit Television (CCTV) images. Those source devices that communicate with the Center of City Traffic Management through a network of TCP/IP fibre optic communications with redundant gigabit Ethernet rings [7].

3.3 Traffic Demand

Existing traffic detectors along the city provide vehicles intensity information that is recorded during a data integration period (currently every ten minutes). A set of detectors contributes to a measurement point in a given coefficient. A measuring point is associated with an urban road segment. The intensity at a measurement point is measured in a number of vehicles per hour, and it is obtained by a linear combination of the intensities of the detectors that compose it. The urban road segments that have a measurement point are considered as a monitored road segment. Thus, a monitored road segment contains the traffic information associated to a specific point in the urban traffic network. In addition to the traffic detectors, some video detectors are capable of classifying the type of vehicle that travels on the roads (information not considered in this study).

Valencia has about 1318 monitored road segments [10] with information intensity recorded every ten minutes; due to the huge amount of data, for this study, we have considered hourly operating data just for the year 2017.

3.4 Origin-destination matrices

In general, in countries of the European Union to collect information about the exposure to urban traffic mobility of the population, different surveys are often carried out with different periodicity since they constitute an exposure measure that is frequently used to identify the demand for transport and mobility in their cities [3].

In Valencian Community (Spain) in 2016 through a survey [8], the inhabitants have been asked to complete a description of the "travel diaries" in which their routes and trips of the day and weekend before the survey. It may have the disadvantage that the interviewees tend not to register very short journeys by not considering them as trips or simply because the people forget them. Anyway, from the collected information, the origin-destination matrices that characterize the usual mobility patterns of the population have been configured [6]. Assuming for the majority of the people have relatively constant mobility patterns during their daily lives, as well as they are consequently easy to remember. The matrices provide high temporal resolution (daily) information about two dimensions: the intensity of exposure and the means of transport for their displacement.

3.5 Traffic light system regulation

In Valencia there is more than 1,000 intersections regulated by traffic lights their approximate location of a large part of them can be seen in the Figure Fig:trafficLight. The city has a Centralized Traffic Control system that allows traffic lights to be regulated in real-time to adapt them to different traffic conditions. Through this system, it is possible to modify the green time of each access, the traffic light cycle and the synchronization between different crossings to avoid generate queues in the streets, thereby reduce the delay and increase the circulation speed. In addition Valencia has a fixed-time emergency system that would work automatically in case of centralized system failure [7].

However, approaching the reality problem in terms of what is happening in the city is difficult since factors such as: (a) intervention of traffic police officers in access streets nearby to educational or sports centers; (b) traffic operator decisions based on CCTV observations; (c) traffic light activations based on the automatic detection of vehicles on the streets or (d) by pedestrian push-to-walk buttons; as well as (e) automatic self-regulation of traffic lights control

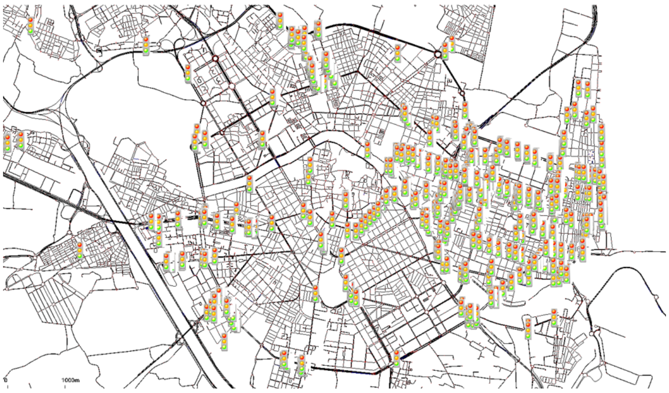


Figure 1: A part of the Traffic Lights definition in Valencia's network.

system should be consider as system inputs for the simulations and they are normally complex to quantify.

3.6 Lagrangian algorithm

A Lagrangian algorithm is a scalar function from which a temporal evolution of a dynamic system can be obtained [1]. The algorithm has a multi-target approach, using as input the initial OD matrix, and the traffic counts. The algorithm incorporates updating processes that control the deviations of the adjusted travel matrix from a previous one. Matrices that after an assignment stage and several repetitions of the loop reach convergence. It reproduces exactly the observed traffic flows, and thus quickly finds a solution to the problem without requiring excessive computational resources [9].

The initial matrix is the O-D matrix obtained from the telephone surveys of households in the Valencian Community, and the vector of variables (Lagrange multipliers) corresponds to the traffic restrictions at a specific time of day based on the meters. Thanks to this algorithm, errors in the estimation of O-D matrices and the reconstruction of observed flows are minimised.

4 Methodology

Urban inventories carried out at any given time allow city decision makers and planners to quantify the magnitude of total emissions between different means of transportation. The inventory is also the starting point for the development of a mitigation strategy. Below, the steps we have followed to define our methodology are described and summarised in the Figure 2.

1. Identification of the starting data
   - Demand identification in urban transport

   With the information provided by the sensorization (magnetic loops on the street of the city of Valencia) corresponding to the year 2017 of all the monitored street segments, a graph shown in Figure 3 is obtained. This, together with the number of journeys characterised in the initial OD matrix, constitutes a measure of exposure used to identify the demand for transport and mobility.

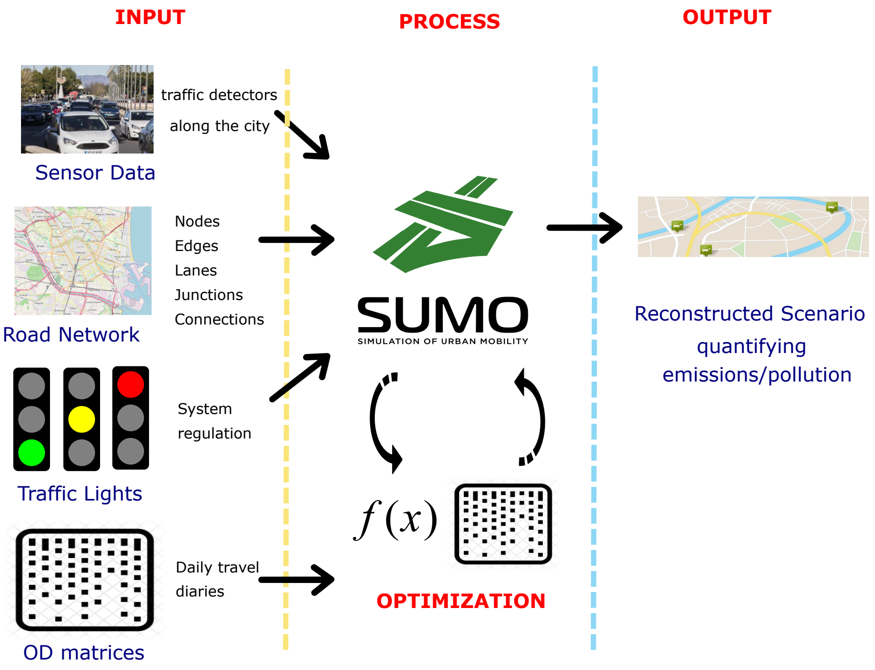


Figure 2: Summary of the followed methodology.

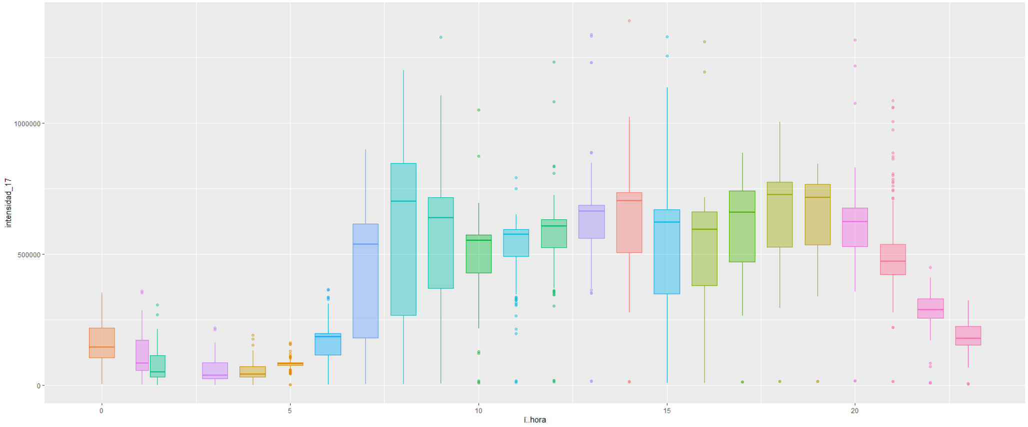


Figure 3: The overall trend of the traffic intensity in Valencian streets for the whole 2017 by daily hours distribution.

- Definition of the vehicle fleet and its characteristics

A customized report is configured from the database consulted in [5] with all these attributes: type of vehicle, municipality of residence, brand, fuel, power, displacement, load, seats, antiquity – technology. By filtering and grouping the available attributes of the studied vehicle fleet, we managed to configure a database composed of just these attributes: vehicle typology, fuel, technological regulations, weighted relative for each of the categorizations. This database will be affected by the corresponding emission factors through the methodology used to obtain the total emissions of each of the pollutants contemplated and the CO_2 equivalent.

2. Design of the road network model

   The network model is built using mainly the open data source offered by OpenStreetMap (OSM). The OSM data set downloaded for the whole metropolitan area of Valencia for the backbone of the model. Due to the city under consideration is where we live and in the case of notice an inconsistency on the data set, it was manually corrected using NETEDIT.

3. Zoning (districts/neighbourhoods) within an aggregate system

   The partition of the study area into a set of pieces (geographical discretization) as known as zones are necessary to characterize the mobility that is currently taking place, quantify it and make forecasts about its possible evolution. We have taken advantage of the maximum pre-existing information. Thus the fundamental administrative division criterion for establishing transport zoning in Valencia has been used here. The whole zonification process in SUMO was made using NETEDIT with the traffic area zone (TAZ). Based on the division by neighbourhoods and districts plus the attached zones that simplify the metropolitan trips of the people, sixty four polygons were created as a traffic area zone.

4. Estimation of the target source matrix

   Once the types of transport have been filtered, and with the zoning defined, a 64x64 table was obtained. The table columns represent the origins “O” (where people live), and the rows represent the destinations “D” of the displacement. The table represents the trips that people make daily.

5. Improvement of the algorithm and the OD matrix

   The proposed algorithm performs a set of operations on each of the elements of the matrix. This approach does not show any inconvenience unless samples with a large number of zones, districts or divisions are treated. A large number of zones increase the dimensionality of the matrices to points where scattered matrices could be found (matrices where a large part of the elements are null or close to zero). An improvement has been implemented in the algorithm to increase its efficiency. To that act just on the necessary elements after an analysis of the different components of the initial matrix.

   One of the elements needed by the OD-matrix refinement program is a three-dimensional probability matrix “p_k”. The refinement matrix shows the probability that a vehicle with a defined origin “i” and destination “j” will circulate through each of the possible ”k” sections existing in the road network. In order to obtain this final improved matrix, various OD matrices are used as iterations that allow us to carry out statistical and probability studies.

6. Mathematical model to refined Origin-Destination matrices

   The model applied is based on an initial OD matrix and measurements of traffic flow at specific points of the road network as in [9]. The desired matrix will generate flows that must have a convergence process towards a scenario that is close to real situations. The convergence process is achieved employing mathematical models based on multivariate statistical methods and controlled dispersion rates that characterise the variability of the data. In this way, we can appreciate the tendency of the obtained values to converge towards real values quantified in situ. On the other hand, these measures help us to determine if our data are far from the expected value. They provide information as to whether the central value "centroid" is adequate to represent the study population. This is useful for comparing distributions and understanding risks in decision making.

Therefore, based on this difference in the number of flows, the OD matrix is changed. If these changes meet certain pre-established convergence conditions, we will consider that they reproduce the current traffic with a certain error that we assume, as long as the convergence requirements are not reached, the iteration process will continue. When this happens, the obtained OD matrix is accepted, and the traffic load is simulated again. The result of the simulation will be a contamination map that will be used for future analysis. A graphical representation of this step can be seen in Figure 4.

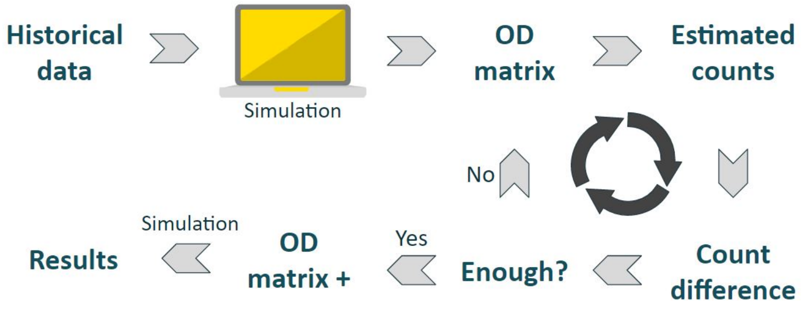


Figure 4: Summary of the optimization process implemented in a mathematical model.

7. Treatment and analysis of results (the geolocation of emissions)

   The results of emissions of the different types of pollutants are shown with (a) the maximum possible granularity and (b) the highest periodicity to show the potential of the developed models with the available data. The geolocation of these emission levels by type of pollutant is a graphic representation that allows visualising and analysing numerical values in a spatial context. For a better analysis of the results, the geolocated quantification can be studied in specific sectors and sections with disaggregation of the different pollutants according to vehicle and fuel typology.

8. Synthesis and information preparation

   From the results obtained with the developed tool, these results could be in one hand a conveniently structured compilation of emission information. On the other hand, the results will be offered as graphics representation that could be displayed in dashboards, always thinking that could be a support tool for decision making.

9. Definition of new scenarios

   With this tool, an infinite number of different scenarios can be considered to carry out simulations and observe their effects on emissions. We mainly propose simulations with most of the traffic light control programs or patterns which the city has (approximately 27 patterns). Where each of the programs is primarily adapted to a wide variety of conditions: weather (sun, wind, rain, cold, heat), time (day, night) and seasonal (weekdays, holidays, holidays) and the possible combinations among them.

5 Results

One of the essential pieces of information to be able to analyse the functioning of a city's transport networks is to obtain the initial origin-destination matrices. These can be used to plan improvements and simulate the mobility dynamics of the city's citizens. In this work, several simulations have been carried out where the number of vehicles inserted corresponds to the number of routes and journeys made by the vehicles represented in the initial origin-destination matrix. These values do not represent the number of physical vehicles in the city, nor the vehicle fleet registered in the city, nor the sum of the detections in an instant of time. Thus, when a vehicle arrives at a destination, it is removed from the simulation. Then, the log records for each of the vehicles inserted in the scenarios proposed in the simulation are analysed. Simulations carried out considering the average daily/hourly data for the whole year of 2017.

In each execution, redefinition of matrices is carried out by adjusting the traffic volumes that pass through the detectors. This data serves as input for a new iterative simulation. Thus with all the different daily hours. Simulating for each hour, and up to ten repetitions of each hour by changing the seed. As the objective of this article is to show the effectiveness of the methodology carried out, we show one of the graphs that will be part of a dashboard for the users who make the decisions. In Figure 5 we can see the neighbourhoods or districts of the city with the average daily CO_2 emissions obtained after the simulation. Where the dark colours have the lowest emissions throughout the year, and the light yellow colours have the highest. With this graph, we confirm an expected result that the neighbourhoods/districts that are next to the large motorways entering and leaving the city are the most affected by CO_2 emissions.

In the same way that CO_2 emissions have been treated, we are able to present the contaminant emissions among which we can highlight Nitrogen Dioxide (NO2) or the atmospheric particulate matter with a diameter of less than 2.5 micrometers (PM2.5) both data available at the end of a SUMO simulation.

6 Further research

As a further research we pretend:

(a) At a strategic level, to contribute to the improvement of scientific and technological knowledge in order to meet the objectives of a low carbon economy, as set by the European Commission for the medium and long term horizon (2030 and 2050 respectively)

(b) At the technological level, to further study and characterise the interrelationship between transport and mobility with the Greenhouse Gases Emissions (GHG) and other pollutants in the urban environment. This will allow us to improve the precision and granularity of the spatial and temporal representation of the emissions.

(c) At the cooperative level, to integrate and combine efforts in research, development and innovation in the fight against and adaptation to climate change in a continuous multisectoral advance towards energy transition.

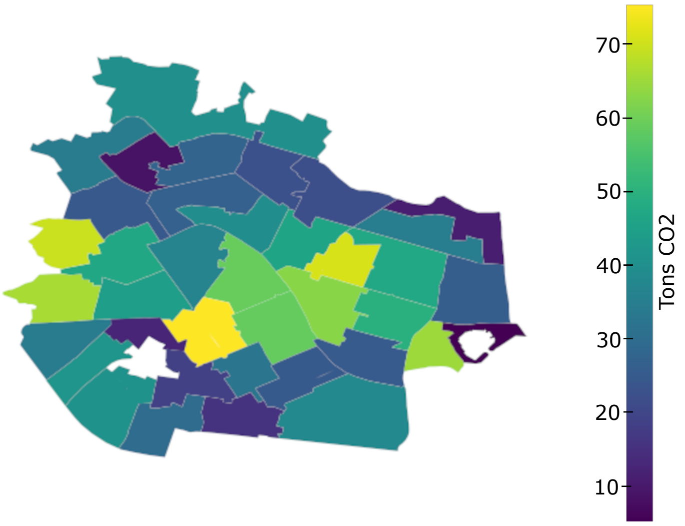


Figure 5: Geolocation of the daily average CO_2 emissions per sensorised square kilometer distributed in the zonal districts in Valencia city.

7 Conclusions

Thanks to the high degree of data digitalisation in Valencia, it is possible to collect data related to traffic in this city in real-time. By inputting this data into different mathematical models information about the environmental impact and influence of traffic on emissions can be obtained. SUMO microsimulation makes it possible to consider different simulation scenarios to analyse how emissions may be altered through time by different traffic mitigation approaches. This type of modelling also allows us to examine the interrelation between factors, variables, sectors, and emissions, and how varying these parameters might change pollutant gas and CO_2 emissions outcomes.

Here we created a mathematical model using the large amount of information available improve the accuracy of emissions predictions. We then compared the results obtained from our models to those from microsimulation with SUMO in a process of convergence and validation. Thus, the accuracy and scientific rigour of this process was greater than simple extrapolation of values from Intergovernmental Panel on Climate Change (IPCC) reports.

Moreover, as a temporary reference product of our methodological proposal, georeferenced emissions allow us to better understand the character and distribution of the carbon footprint and other pollutants in Valencia. In turn, this will generate new challenges and increase opportunities for the Valencian local government to manage these problems. With this information, both the public administration and planners can measure the impact of their mitigation plans and adapt their approaches to climate change in order to reduce the carbon footprint of the transport sector.

8 Acknowledgments

The authors would like to thank the Smart City Office of the Ajuntament de Valencia (www.smartcity.valencia.es), ETRA I+D (https://www.grupoetra.com/), and the financial support of the Innovation Agency of the Valencia Region (AVI) under grant INNEST00/18/005 corresponding to the TRUST2030 project.

References

[1] Michael G.H. Bell. The estimation of origin-destination matrices by constrained generalised least squares. *Transportation Research Part B*, 25(1):13–22, 1991.

[2] Laura Bieker, Daniel Krajzewicz, Andreas Leich, and Martin Dippold. Local Emissions Monitoring using vehicular communication. In *SUMO 2016 – Traffic, Mobility, and Logistics*, pages 165–172, 2016.

[3] Robin A. Blanchard, Anita M. Myers, and Michelle M. Porter. Correspondence between self-reported and objective measures of driving exposure and patterns in older drivers. *Accident Analysis and Prevention*, 42(2):523–529, 2010.

[4] Chou-tsang Chang. Estimation of Carbon Dioxide Emissions Generated by Building and Traffic in Taichung City. 2018.

[5] Dirección General de Tráfico. Portal estadístico de vehículos y conductores.

[6] Dirección General de Tráfico. Movilidad: ciudadano&vehículo. *Tráfico y Seguridad Vial*, (248):60–61, 2018.

[7] Ayuntamiento de Valencia. Centro de Gestión de Tráfico de Valencia, last accesed: 16th Jan, 2020.

[8] Conselleria d'Habitatge Obres Públiques i Vertebració del Territori. Pla Mobilitat Metropolità València. Technical report, Generalitat Valenciana, Valencia, 2018.

[9] Javier Doblas and Francisco G. Benitez. An approach to estimating and updating origin-destination matrices based upon traffic counts preserving the prior structure of a survey matrix. *Transportation Research Part B: Methodological*, 39(7):565–591, 2005.

[10] Grupo ETRA. Barcelona Traffic Regulation Systems, last accesed: 16th Jan, 2020.

[11] Eclipse Foundation. Simulation of Urban MObility, 2020. `https://www.eclipse.org/sumo/`, Last checked on 2020/09/14.

[12] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[13] Sims R., R. Schaeffer, F. Creutzig, X. Cruz-Núñez, M. D'Agosto, D. Dimitriu, M. J. Figueroa Meza, L. Fulton, S. Kobayashi, O. Lah, A. McKinnon, P. Newman, M. Ouyang, J. J. Schauer, D. Sperling and G. Tiwari. Transport. In: Climate Change 2014: Mitigation of Climate Change. Contribution of Working Group III. Technical report, 2014.

[14] Jorge Luis Zambrano-Martinez, Carlos T. Calafate, David Soler, Lenin Guillermo Lemus-Zúñiga, Juan Carlos Cano, Pietro Manzoni, and Thierry Gayraud. A centralized route-management solution for autonomous vehicles in urban areas. *Electronics (Switzerland)*, 8(7):1–21, 2019.

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.99

Published: 28 Jun. 2022

SAGA: An Activity-based Multi-modal Mobility Scenario Generator for SUMO

Lara CODECÁ[1], Jakob ERDMANN[2], Vinny CAHILL[1], and Jérôme HÄRRI[3]
[1] Trinity College Dublin - School of Computer Science and Statistics
Dublin, Dublin 2, Ireland
Lara.Codeca@tcd.ie, Vinny.Cahill@tcd.ie
[2] German Aerospace Center - Institute of Transportation Systems
Rutherfordstr. 2, 12489 Berlin, Germany
Jakob.Erdmann@dlr.de
[3] EURECOM - Communication Systems Department
06904 Sophia-Antipolis, France
Jerome.Haerri@eurecom.fr

Abstract

In this paper, we define a workflow and a toolchain to support fast mobility scenario prototyping based on open data and open-source software. SAGA is an activity-based multi-modal mobility scenario generator for the Simulation of Urban MObiltiy (SUMO). Starting from an OpenStreetMap (OSM) file, SAGA extracts the data required to build a multi-modal scenario, and in a step-by-step fashion, generates the configurations needed to execute it, including the intermediate steps required to refine the scenario with additional data, allowing the iterative improvement of realism and representativeness.

The workflow implemented, extended, and automated by SAGA was developed while hand-crafting the Monaco SUMO Traffic (MoST) Scenario. Based on the fast prototyping capabilities added by SAGA, the creation of a multi-modal mobility scenario is readily achievable, and the incremental process to fine-tune it is supported by a workflow instead of being solely based on expert knowledge and experience.

Based on previous experience, the generation of the first working prototype of a city-scale multi-modal mobility scenario may take months of work and expert knowledge. SAGA automatically generates such a prototype, and all the intermediate configuration files are made available for further iterative improvements.

1 Introduction

Over the last decade, the focus on sustainable development has been prioritized by international leaders[1]. Sustainable development is an umbrella term that covers multiple areas, such as responsible consumption and production, climate action, and sustainable cities and communities. Moreover, with the increasing volume of transportation and mobility information available in cities, both the research and industry communities are promoting smart cities, smart mobility, and Intelligent Transportation Systems (ITS) as one means of achieving sustainable development. These innovations have brought to light new problems that require solving [1, 2, 3]. Although the range of issues afflicting smart mobility is quite broad, the problem-solving methodology is straightforward. Investigators find a problem that needs to be solved, and when related to large and/or complex systems, simulation tools are the first option to study potential solutions. More precisely, independently of the problem at hand, the steps to solve it are: (i) identify the problem, (ii) identify the models required to represent it,

[1] United Nations Sustainable Development Goals: https://www.un.org/sustainabledevelopment/sustainable-development-goals/ Access: March, 2022

(iii) build a simulation scenario representative of the problem, and (iv) analyze, implement and validate the solution in the simulation. With more and more cities and industries interested in solving smart mobility problems tailored to their own specific needs, overcoming the difficulty of building one or more scenarios in a fast and efficient way is becoming more pressing than ever, given that the generation of representative mobility scenarios has been an ad-hoc process that requires a large amount of high-quality data, expert knowledge, and hand-tuning.

In this paper, we define a workflow and a toolchain to support fast mobility scenario prototyping based on open data and open-source software, the use of which facilitate addressing the sustainable development challenge. The publication of data in open, free, and reusable formats promotes innovation and transparency. The choice of using open-source software enables collaborative and public development, allowing diverse perspectives (beyond those of a single community) to be leveraged. OpenStreetMap (OSM) [4] is a crowd-sourced and user-generated collaborative project to create a free editable map of the world. All the geographic information is volunteered, and the aggregated geodata is the primary output of the project. Simulation of Urban MObiltiy (SUMO) [5] is an open-source, highly portable, microscopic, and continuous road traffic simulator designed to handle large road networks. With a very active development team and an engaged user community, SUMO represents a viable choice as an open-source simulator, with an ever-expanding toolset capable of handling open data formats such as OSM.

Section 3 presents in detail the requirements for and issues encountered in building representative multi-modal mobility simulation scenarios. This is based on the observation that people tend to move around during the day based on the location of their activities (e.g., work, sport, school). Hence, a reasonable representation of the mobility in a city can be the set of journeys made by the population, based on their undertaking generic sequences of activities, during the day. Much research has been done on activity-based mobility modeling [6, 7], and it has been shown that disaggregated activity-based models better represent individual decision-making [8], improving the realism of the generated mobility compared to traditional aggregated demand models.

Section 4 presents SUMO Activity GenerAtion (SAGA), a user-defined activity-based multi-modal mobility scenario generator for SUMO. A simulation scenario capable of handling activity-based mobility requires detailed information on the environment (e.g., buildings, PoIs), as well as the transportation infrastructure. The framework proposed in this paper extracts the additional environmental information available automatically, generates all the configuration files required by SUMO, and fills the missing information with sensible default values, ready to be enriched with external data, if available. Starting from an OSM file, SAGA extracts the data required to build a multi-modal scenario, and in a step-by-step fashion, generates all the additional configurations needed to compose the scenario (e.g., parking areas, buildings, Points of Interest (PoIs)), providing the scaffolding required to iteratively refine the scenario with additional external data. Figure 1 presents an overview of the SAGA operating model. On one side, we have the problem (application or optimization) that requires study, and on the other side, we have SUMO, a powerful microscopic mobility simulator, which can be quite cumbersome to configure. SAGA positions itself in between. Starting from an OSM file, it extracts the data required and produces all the SUMO configuration files needed to run a mobility simulation. Additionally, it provides sensible default values for the missing or incomplete information needed for the simulation, filling the gaps while providing the structure required to extend the scenario with additional external data. Implemented in Python 3.7, it primarily uses the configuration tools provided by SUMO, extending them when required. SAGA provides both the application to generate the complete multi-modal mobility scenario and the isolated applications that can be used to polish and enhance each step with additional data.

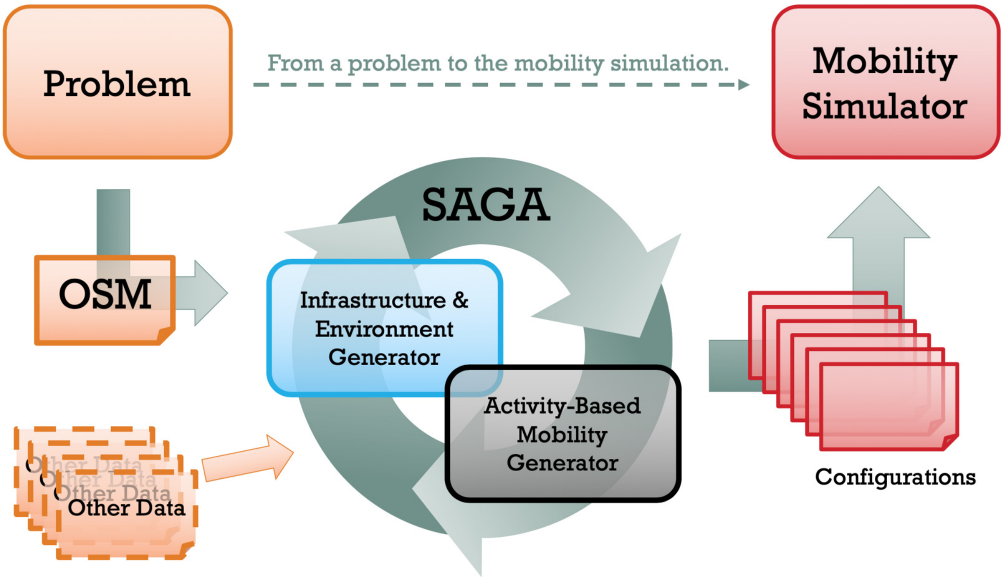

Problem
From a problem to the mobility simulation.
Mobility Simulator
SAGA
OSM
Infrastructure & Environment Generator
Activity-Based Mobility Generator
Other Data
Configurations

Figure 1: Overview of SAGA s interaction model.

The workflow implemented by SAGA was developed while hand-crafting the Monaco SUMO Traffic (MoST) Scenario [9] (presented in Section 5) and automated to achieve efficiency in generating mobility scenarios of various sizes. Other tools and models available focus on mobility generation and optimization, leaving out the configuration of the underlying transportation infrastructure and data extraction. SAGA provides the tools to do both infrastructure and mobility generation, focusing on fully configurable isolated steps, meant to increase the usability with additional sources of data that may be available, enabling fast prototyping of multi-modal mobility scenarios facilitating planning and reuse.

The structure of the paper is the following: Section 2 positions SAGA s scope and features in comparison to related work. Multi-modal mobility modeling, and its requirements are presented in Section 3. Section 4 is a detailed explanation of SAGA s workflow and algorithms. Significant use-cases are presented in Section 5. A final consideration of SAGA and its development is discussed in Section 6.

2 Related Work

In this section, we position SAGA relative to related work. The literature describes many mobility simulators, ranging from open source to commercial. Among the open-source software, we can find TRANSSIM [10], MATSim [11], SimMobility [12], and SUMO [5]. Because of our focus on open-source, we do not consider commercial mobility simulators such as PVT VISSIM[2], Paramics[3], and Citilabs s Cube[4] further here.

[2] VISSIM: http://vision-traffic.ptvgroup.com/en-us/training-support/support/ptv-vissim/ Access: March, 2022

[3] Paramics: https://www.paramics.co.uk/en/ Access: March, 2022

[4] Cube: https://www.citilabs.com/software/ Access: March, 2022

TRANSSIM, MATSim, and SimMobility address a specific traffic optimization problem: given a population generate an activity-based traffic assignment and iterate the generation until an equilibrium (e.g., Wardrop, Nash) is reached. Although the simulators mentioned above are efficient and powerful in their own right, their specificity makes them less general-purpose, flexible, and interactive (or scriptable) than SUMO and its Traffic Control Interface (TraCI).

SUMOPy[5] implements an activity-based iterative optimization capability similar to TRANSSIM, and MATSim, and it is provided as a contributed tool for SUMO. The authors present the framework in [13], where they explain how the use of SUMO is suited to evaluate multi-modal travel plans and how the iterative optimization implemented by SUMOPy changes the trip plans based on the travel times experienced after each simulation run, showing that the algorithm converges and an equilibrium is reached. Although the activity-based mobility generation provided by SUMOPy is similar to the one provided by SAGA, SUMOPy s main objective is to solve the traffic assignment problem, to provide solutions similar to TRANSSIM or MATSim, and not to automate the scenario generation, making it not fully customizable and general-purpose.

Considering frameworks for large scale transportation simulations, we can find a flexible automated mobility-on-demand modeling and simulation framework (AMoD), developed within SimMobility [14], and CityMoS [15], formerly known as SEMSim [16] a distributed architecture for multi-scale traffic simulations. Although these frameworks are very extensive, they assume that a scenario exists, and it is already adequately implemented.

Considering Figure 1, state-of-the-art simulation frameworks focus on mobility simulators and mobility generation, leaving the cumbersome and complex task of generating the transportation infrastructure model to the expert users. SAGA s role is to support and automate the complete mobility scenario generation, from the transportation infrastructure to the activity-based mobility plans. The predecessors of SAGA are ACTIVITYGEN[6] and OSMWebWizard[7], tools already available in the SUMO toolbox. OSMWebWizard is conceptually similar to SAGA because it provides the scaffolding required to generate a simple mobility scenario based on random trip definitions instead of OD-matrices or activity-based trip plans. SAGA builds upon the tools used by OSMWebWizard, extracting the additional information required to generate representative activity-based mobility plans. ACTIVITYGEN produces trip-based mobility meant to represent the concatenation of trips required to satisfy the sequence of activities in a personal plan. Used together with OSMWebWizard, it may produce results similar to SAGA after a cumbersome manual customization process, although the actual journey plan (and information connected to its sequence of activities) is not preserved. Unfortunately, the project is discontinued and, unlike SAGA, not able to leverage the state of the art multi-modal mobility models and infrastructure implemented in SUMO.

To the best of our knowledge, SAGA is the first fully customized activity-based multi-modal mobility scenario generator that explicitly focuses on the transportation infrastructure definition and extraction of additional environmental information. Its goal is to automate fast mobility scenario prototyping and to provide both the conceptual workflow and the scaffolding (in terms of detailed configurations) required to enhance and tune the intended mobility scenario. SAGA is freely available under the EPLv2 license on GitHub[8], and it is included in the contributed tools since SUMO v1.3.0.

[5] SUMO Wiki: SUMOPy https://sumo.dlr.de/docs/Contributed/SUMOPy.html Access: March, 2022 SUMO

[6] Wiki: ACTIVITYGEN https://sumo.dlr.de/docs/ACTIVITYGEN.html Access: March, 2022

[7] SUMO Wiki: OSMWebWizard http://sumo.sourceforge.net/userdoc/Tutorials/OSMWebWizard.html Access: March, 2022

[8] SAGA on GitHub: https://github.com/lcodeca/SUMOActivityGen Access: March, 2022

3 Modelling Smart Mobility

As mentioned in the introduction, most of the latest research on mobility modelling is pursuing activity-based mobility. Nonetheless, the latest research is not addressing the requirements to consistently build a simulation scenario able to leverage activity-based mobility generation, and the configuration of transportation infrastructure and environmental information are left to expert knowledge and personal experience.

In order to create realistic mobility patterns, the population is defined through individual people with daily routines. Each routine is defined as an activity chain, a sequence of generic activities used to model travelers behaviours (such as work, school, and sport). The mobility plans generated by SAGA are in the form of a set of personal journey plans composed of multiple trips (with an associated transportation mode) used to connect the location of the activities in the given chain.

In the scope of this paper, a mobility simulation is composed of three main interacting components: (i) the vehicular mobility models, and the definitions of both (ii) transportation infrastructure and (iii) the trip plans associated with a given traffic demand (with the associated activity chains). Although the vehicular mobility models described in the literature can be further improved and calibrated, our focus is to improve and automate the generation of the transportation infrastructure and the trip planning associated with the given traffic demand and respective activity chains.

3.1 Activity-based Mobility Plans

Over the years, it has been shown that disaggregated activity-based models better represent individual decision-making [8], improving the realism of the generated mobility compared to traditional aggregated demand models.

There are multiple ways to generate these type of mobility traces, with increasing levels of complexity and flexibility. Thinking about our daily routine, we can construct a complete daily plan built from simple trips consistently aggregated. For example, a person may drive their car from home to school, leave the kids, drive to work, and look for a parking spot. Over the lunch break, the person can walk to the bus stop, take the bus to a stop closer to the desired restaurant, and walk to it. Similar steps can be used on the way back to the office, and, eventually, home. The complete plan is composed of multiple activities, each of them associated with location, starting time, and duration. The complexity of the back-and-forth trip linking each activity may vary, depending on the requirements and the trip feasibility. A multi-modal scenario generated using SAGA is suited to model the complex activity-based journey previously described.

Although the chains of activities may be implied by the use of Origin-Destination (OD)-matrices built from surveys and counters, when used to generate the traffic assignment, the sequences of activities are not usually explicit. The resulting mobility definition obtained from complete journey plans preserves important information on internal consistency; information that can be crucial during the decision-making associated with specific applications and optimizations. The mobility produced by SAGA is in the format of personal journeys composed but multiple trips, where the connection between each activity and the trip (with the mode of transport) is explicit. This plan definition leverages the latest multi-modal capabilities provided by SUMO.

3.2 Components of a Mobility Simulation

The transportation infrastructure represents the backbone of the simulated environment. It consists mainly of roads and intersections, but the presence of additional environmental information is crucial to building a representative version of the real world.

Table 1 shows a non-exhaustive list of important features. The first column names the feature. A tick in the second column means that the feature is explicitly defined and modeled in SUMO. A tick in the third column means that, although it is not explicitly modeled in SUMO, it is still possible to build it using other features. The fourth and last column, if ticked, means that SAGA can leverage or improve the feature. The reason why SAGA does not leverage containers is that it is targeting mobility for people and not for goods. Similarly, the meaning of traffic signs and traffic detectors is embedded in the infrastructure and does not add additional information for activities. In SUMO, parking maneuvers can be partially modeled through delays and obstruction to traffic. This feature, if configured, will change the resulting trip time for SAGA, but it will not otherwise affect the decision making. Thanks to SUMO's rapid development cycles, features such as taxis and bicycle stands, shared rides, charging stations, and consistent inter-modal mobility are planned to be supported by SAGA as soon as the underlying models reach stability.

Given that traffic composition has a significant impact on the resulting mobility, the simulation should be based on multi-modal mobility models (different types of vehicles interact with each other) capable of inter-modal trips (multiple modes of transport used consistently during the same journey). Nonetheless, the only inter-modal trip capabilities implemented by SAGA are based on the public transportation infrastructure. Therefore, SAGA is capable of generating a journey plan that uses both buses and trains, but it is unable to generate a plan involving both bicycles and taxis for different legs of the journey. The final mobility traces are generated starting from an OD matrix, usually containing the origin, destination, mode of transport, and the number of vehicles. Additionally, SAGA supports the creation of generic parametrized chains of activities that can be tuned with additional information, if available. The accuracy of all the previously mentioned information, the aggregation, and the mobility generation methodology determine the degree of representativeness of the resulting mobility scenario.

3.3 Automation Process

In principle, the generation of a general-purpose large-scale mobility simulation tailored to study smart mobility should be an automatable process. The automation of the scenario generation implies a sequence of actions that, starting from a dataset, delivers the mobility simulation scenario without user intervention. In reality, we must take into account the heterogeneity of the datasets required to build it, their accuracy, the limitations of the models involved, and the complexity of the interactions between the models.

In this paper, we present a framework able to provide a scaffolding infrastructure for the automation process based on the SUMO mobility simulator. A user that wants to use SAGA to generate a mobility scenario automatically can run the main application with only the OSM dataset of the area of interest as input. The output provided by SAGA is the multi-modal mobility scenario of the given area, and its representativeness depends on the quality of the OSM data. Given that building a mobility scenario is an iterative process, based on SAGA's configurable stages, the iterations required to obtain representative infrastructure and mobility can be scripted and individually tuned. As previously discussed in Section 2, independently of the mobility simulator chosen, the automatic generation of mobility simulations is a problem

Table 1: List of features crucial for the activity-based mobility generation.

	Explicitly defined	Modellable with SUMO	Used by SAGA
Streets	✓	✓	✓
Railways	✓	✓	✓
Waterways	✓	✓	✓
Bridges	✗	✓	✓
Over/Under passes	✗	✓	✓
Stairs	✗	✗	✗
People	✓	✓	✓
Vehicles	✓	✓	✓
Containers	✓	✓	✗
Intersections	✓	✓	✓
Traffic signs	✗	✓	✗
Traffic lights	✓	✓	✓
Pedestrian areas	✓	✓	✓
Pedestrian crossings	✓	✓	✓
Intersection permissions	✓	✓	✓
Lane permissions	✓	✓	✓
On-street parking	✓	✓	✓
Parking areas	✓	✓	✓
Multi-layer parking areas	✗	✓	✓
Parking maneuvers	✗	✗	✗
Land use	✗	✓	✓
Buildings	✓	✓	✓
Points of interest	✓	✓	✓
Public transports stops	✓	✓	✓
Public transports schedule	✗	✓	✓
Taxi stands	✗	✓	✗
Bicycle stands	✗	✓	✗
Detectors	✓	✓	✗
Charging stations	✓	✓	✗
Activities	✗	✓	✓
Activity chains	✗	✓	✓
Rerouting	✓	✓	✓
On-demand rides	✗	✓	✓
Shared rides	✓	✓	✗
Multi-modal trips	✓	✓	✓
Inter-modal trips	✓	✓	✗

not yet solved, and there are not many open-source tools capable of enabling this automated generation in a fully configurable manner. Taking into account that complete datasets are hard to obtain, SAGA's goal is to structure and support the process of fixing and filling in the inconsistent information with sensible default values. Finally, considering that mobility simulators implement complex parametrized models (improved regularly based on the latest research studies), and that new multi-modal features are implemented frequently, a tool such as SAGA is required to take into account all the dependencies and to facilitate and improve the resulting simulations.

4 Mobility Scenario Generation Framework

SAGA is an activity-based multi-modal mobility scenario generator for SUMO. Starting from an OSM file, SAGA extracts data required to build a multi-modal scenario, and in a step-by-step fashion, generates the configurations (e.g., parking areas, buildings, generic traffic demands and activities) needed to compose a simulation scenario, while providing the scaffolding required to iteratively refine it with additional data to improve realism.

The OSM data format allows the definition of transportation and environmental features required to generate a mobility scenario. Unfortunately, the completeness of this information is not consistent across the dataset. SAGA extracts these data and computes sensible default values for the missing information, providing not only a mobility simulation scenario, but also the intermediate configurations required to enable the incorporation of additional sources of data, if available.

SAGA is freely available under the EPLv2 license on GitHub[9], and it is included in the contributed tools since SUMO v1.3.0.

4.1 Requirements

With the final goal in mind of implementing, studying, and evaluating various mobility applications and optimizations, the scenario generation process needs to be as straightforward as possible for the user to minimize overhead. Although the generation process still requires supervision, the use of a reliable automation framework provides a consistent workflow for speeding up the iterative refinement of the scenario.

A mobility scenario generation framework needs to address both the static infrastructure and mobility traces. Without a consistent and detailed transportation infrastructure, it is not possible to generate reasonable mobility traces. In addition, the information initially available may not be complete enough for the scenario to be representative of the problem at hand, but the guided process can be used to integrate missing data from other sources. Specific requirements identified are the following: (i) the scenario generator must be fully configurable and general-purpose, (ii) the workflow must be straightforward and able to keep track of all the interactions between features, and (iii) all the steps must be independent of each other to allow iterative refinement and data integration of incomplete or missing information.

The above mentioned requirements are tightly coupled with the capabilities and requirements of the mobility simulator. SAGA uses SUMO and its toolchain, but it implements additional components to extract and configure the required environmental data automatically, enabling the straightforward inclusion of the specific scenario components required by the activity-based mobility generation.

4.2 Top-down Overview

Taking into account that the scenarios generated using SAGA are expected to be used to measure and evaluate the impact of the candidate applications and optimizations on mobility features, we analyze the requirements of the mobility components first and then explain the information required to obtain them.

[9] SAGA on GitHub: https://github.com/lcodeca/SUMOActivityGen

Activity-Based Mobility Generation Our primary assumption is that only partial information on the population, the activities, and the traffic demand (including the OD-matrix) is available for the mobility scenario generation. The missing data required to build the mobility scenario is based on parametrized numerical estimations and approximations. The automation of the activity-based mobility generation then requires a generic definition for the activities, the composition of the chains with their associated probabilities, and the transport modes linked with each chain. Each activity requires a location in order to generate the associated trip, and the exact sequence of activities influence their expected locations (detailed in Section 4.3.2). The locations are based on the Traffic Assignment Zone (TAZ) definition (configured as detailed in Section 4.3.1). Each trip linking activities has an associated transport mode, and depending on the configuration, additional requirements can be applied. For example, on-demand mobility services, such as taxis, do not require parking at the destination, unlike personal vehicles.

Transportation Infrastructure Generation The generation of detailed and consistent transportation infrastructure is required to populate the scenario with representative mobility. Aiming to generate a multi-modal mobility model, the primary feature to extract is the road topology, with all the data associated with the streets' and lanes' permissions, and the intersections' signaling and geometry. Additional transportation and environmental information on the public transportation infrastructure is needed. With public transportation, we mean any mode of transport with a schedule, a route, and predefined stops. Based on the city in question, it may be composed of buses, trains, metros, trams, and ferries. Finally, it is necessary to extract and generate the location and capacity of both on-street and off-street parking areas.

Additional Environmental Features Generation Transportation infrastructure aside, additional environmental information is required to generate activity-based mobility. Data on building size and location, the associated land use, and location and type of PoIs can be used to complement incomplete or missing information on the traffic demand. Reasonably, locations with large commercial buildings attract more people than residential areas with smaller buildings. Origins and destinations in the OD-matrix are based on the relative weight of the TAZ. In case external information is not available on the shape and location of TAZ, administrative boundaries are straightforward to find in open datasets, and they can be used instead.

4.3 Workflow and Capabilities

In this section, we discuss in detail the framework and the associated workflow available to guide the mobility scenario generation process and to keep track of all the requirements.

4.3.1 Iterative Scenario Generation Process

Starting from an OSM file, SAGA generates a running multi-modal mobility scenario, including the configuration files used by SAGA's components, which can be enriched to tune both the transportation infrastructure and mobility traces. The representativeness of the generated scenario depends entirely upon the quality of the information available in the OSM file. Figure 2 presents all the isolated steps in the workflow used by SAGA to generate the scenario. It heavily uses SUMO tools, and it provides additional tools only when necessary, in an ongoing effort to improve the SUMO toolbox, when possible.

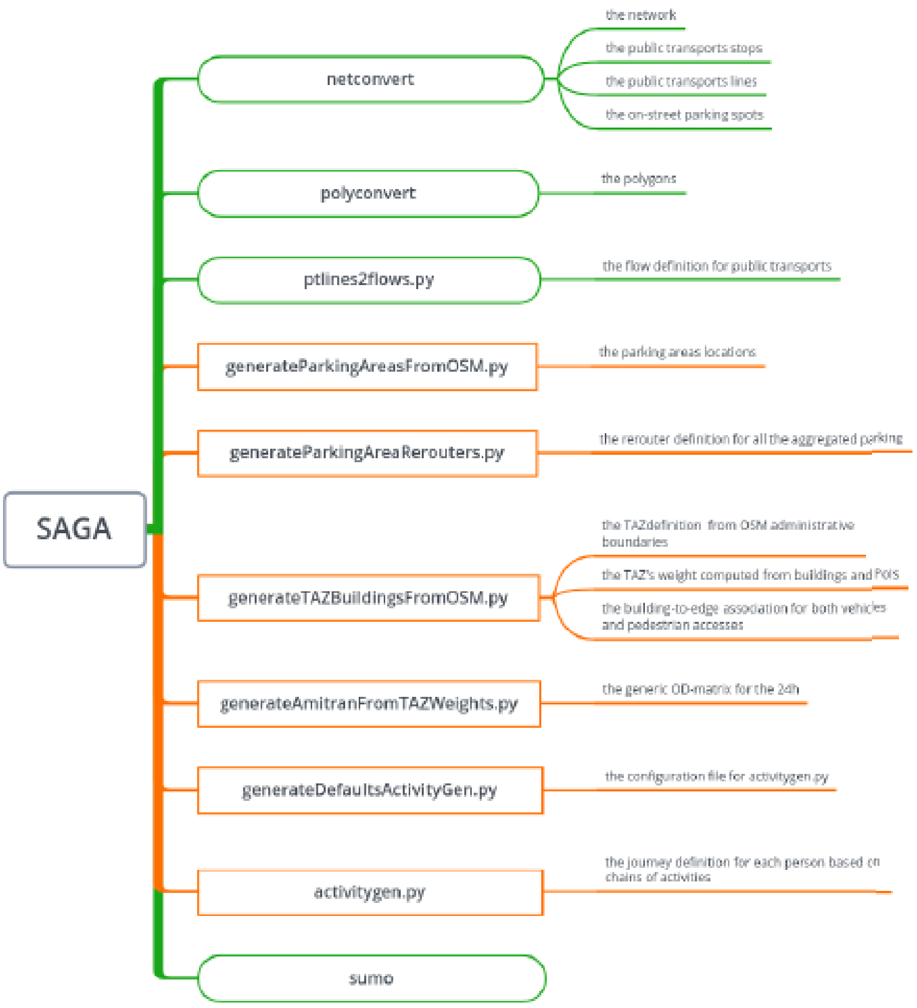

SAGA
netconvert
the network
the public transports stops
the public transports lines
the on-street parking spots
polyconvert
the polygons
ptlines2flows.py
the flow definition for public transports
generateParkingAreasFromOSM.py
the parking areas locations
generateParkingAreaRerouters.py
the rerouter definition for all the aggregated parking
generateTAZBuildingsFromOSM.py
the TAZdefinition from OSM administrative boundaries
the TAZ's weight computed from buildings and PoIs
the building-to-edge association for both vehicles and pedestrian accesses
generateAmitranFromTAZWeights.py
the generic OD-matrix for the 24h
generateDefaultsActivityGen.py
the configuration file for activitygen.py
activitygen.py
the journey definition for each person based on chains of activities
sumo

Figure 2: Overview of the scenario generation process. The rounded boxes in green are the tools provided by SUMO. The squared boxes in orange are the tools provided by SAGA. On the right of each box is listed the generated output.

1. The OSM file is processed using `netconvert`[10] to obtain (i) the network definition, (ii) the public transportation stops, (iii) the public transportation lines, and (iv) the on-street parking spots.
2. The OSM file and the newly generated network definition is processed using `polyconvert`[11] to extract all the environmental geometrical features available (e.g., buildings, PoIs, and

[10] SUMO Wiki: NETCONVERT https://sumo.dlr.de/docs/NETCONVERT.html Access: March, 2022 SUMO
[11] Wiki: POLYCONVERT https://sumo.dlr.de/docs/POLYCONVERT.html Access: March, 2022

administrative boundaries). These features, although cosmetic for SUMO, can be accessed at simulation time using TraCI, enriching the capabilities of the resulting mobility scenario.

3. Based on the network file and the public transport data extracted from OSM, we use `ptlines2flows.py`[12] to generate the public transportation timetable with the relevant flow of vehicles. More precisely, each line (e.g., bus routes, trains, or other) is defined as a series of vehicles (flow) with a fixed route and scheduled stops, that repeats over time.

4. Starting from the OSM file and the network definition, additional off-street parking areas[13] are extracted. In case the parking capacity is not defined in OSM, a parametrized value (adjustable afterward) is used. The previously extracted definition for the on-street parking is then merged with the newly generated parking area information, creating a complete and consistent definition for all parking.

5. For each parking area (on-street or otherwise), SUMO recommends defining the possible parking alternatives to be used by SUMO to reroute traffic in case a parking area is full. The definition of the alternatives is done using `generateParkingAreaRerouters.py`[14].

6. The extraction of both TAZs and buildings is performed by SAGA:

   (a) Using the OSM file, the administrative boundaries[15] are extracted and are used to generate the TAZ definition for SUMO (as a list of streets).

   (b) The TAZ are weighted based on the number of buildings, PoIs, and infrastructure available in the area. The weight associated with a TAZ is an estimation of its potential attractiveness, and is computed based on e.g., the number of streets, buildings, and PoIs defined in the TAZ, divided by the area of the TAZ itself. Unless otherwise specified, it is assumed that the more infrastructure and PoIs are available in a given geographical area, the higher is the probability of having traffic going through it.

   (c) Each building is associated with one or more TAZs due to possible overlapping administrative boundaries. Two possible accesses to each building are computed based on the closest street that allows vehicles, and the closest path that allows pedestrians. The two accesses may be the same. For each access, the weighted probability of selecting it is computed (as an origin or destination) within the TAZ. This probability is the area[16] of the building relative to the area of the TAZ, following the principle that (unless otherwise specified) larger buildings attract more people.

7. A generic OD-matrix based on 24 hours of mobility is generated based on the weighted TAZ. This OD-matrix is defined in the commonly used Amitran[17] format.

[12] SUMO Wiki: ptlines2flows.py https://sumo.dlr.de/docs/Tutorials/PT_from_OpenStreetMap.html Access: March, 2022

[13] SUMO Wiki: Parking areas https://sumo.dlr.de/docs/Simulation/ParkingArea.html Access: March, 2022

[14] SUMO Wiki: Parking Area Rerouters https://sumo.dlr.de/docs/Simulation/Rerouter.html#rerouting_to_an_alternative_parking_area Access: March, 2022

[15] OpenStreetMap Wiki: administrative boundaries https://wiki.openstreetmap.org/wiki/Tag:boundary=administrative Access: March, 2022

[16] The use of the building's area instead of its volume generates inaccurate weights in case of geographical areas with skyscrapers, and both large and small buildings. Although OSM allows the definition of the volume, the information is missing from the majority of the buildings. On the other hand, the building's area is always computable.

[17] SUMO Wiki: Amitran https://sumo.dlr.de/docs/Demand/Importing_O/D_Matrices.html#the_amitran_format Access: March, 2022

8. Based on all the previously extracted data, a default configuration for the activity-based mobility generation is created as explained in Section 4.3.2.

9. The final activity-based mobility composed of personal journey plans is generated using the algorithm presented in Section 4.3.2.

10. The scenario is run using a default SUMO configuration file.

All the steps described above are isolated applications with precise input and output files. Each configuration can be modified manually to add specific information available for the chosen scenario or to refine it. Once one configuration is changed, all the following steps should be re-launched to maintain consistency.

4.3.2 Activity-Based Mobility Generation in Detail

The activity-based mobility generation is parametrized through a configuration file and fully scriptable. The complete mobility can be generated at once, or single slices[18] can be individually customized and tuned, leaving SUMO to merge them afterward during the simulation. The mobility generator interacts with SUMO using the TraCI Python APIs[19]. In the following paragraph, the required parameters and their implications are discussed in detail.

Input Figure 3 shows the input required by the mobility generator as described in the following sections:

The SUMO files required are (i) the network definition, (ii) a default definition of the vehicles available, (iii) the definition of the parking areas, and (iv) the configuration file with the public transportation available (in terms of stops and flows). These files are used to configure a basic SUMO simulation to be queried during the mobility generation.

The transportation modes behavior is customized by specifying (i) if some parking areas cannot be used, (ii) which types of vehicle are allowed to use the parking spots, and (iii) if the parameter associated with the mode of transport defined in the traffic demand should be used as a weight (discussed in more detail afterwards) or the probability of selecting each given mode.

The population is customized in terms of (i) the number of people (hence the number of individual plans) and (ii) the TAZ definition (with its associated streets, weights, and buildings). The TAZ can be aggregated and renamed for more straightforward use. When the aggregated name is used in the traffic demand definition, one of the individual TAZ is selected using a uniform probability distribution.

Three categories of activities can be defined: *Home*, *Primary*, and *Secondary*. All the parameters required for each of them are defined by separate Gaussian distributions with given mean and standard deviation. Home activity and Secondary activities have only duration. The Home activity is unique and used as place-holder for the origin defined in the traffic demand. Primary activities have both start time and duration. In any given chain, even if multiple Primary activities are defined, their location is unique and associated with the destination set in the traffic demand.

[18] A slice of mobility is one single entry of the OD-matrix.

[19] SUMO Wiki: TraCI Python APIs https://sumo.dlr.de/docs/TraCI/Interfacing_TraCI_from_Python.html Access: March, 2022

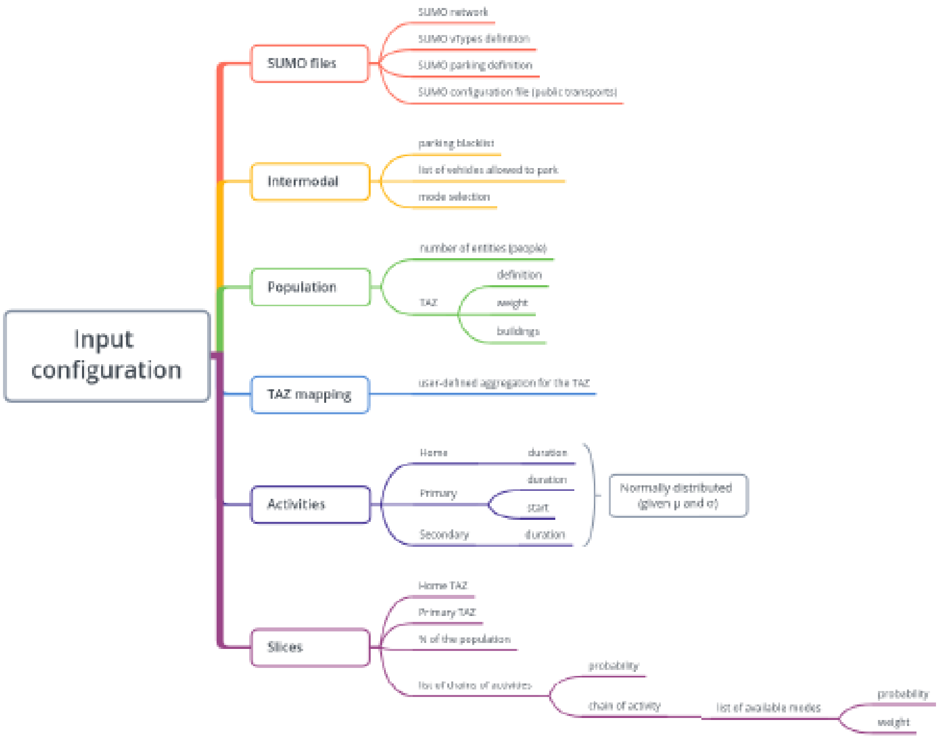


Figure 3: Input required by the activity-based mobility generator.

- The traffic demand is defined in slices. Each of them requires the percentage of the population associated with it, the TAZ associated with the Home activity (used as origin), the TAZ associated with the Primary activity (used as primary destination), and the distribution of chains of activities. Each activity chain in the distribution must start and end with the Home activity and needs to contain at least one Primary activity. Secondary activities can be arbitrarily inserted in the chain, but it is not possible to have two Secondary activities one after the other. Each chain must define the list of modes available, and each available mode must be associated with the probability of choosing it or its cost (defined in the Generation paragraph summarized in Figure 5).

Secondary Activity Location Figure 4 Figure shows the decision-making process used to select the locations for secondary activities. Starting from the assumption that only partial information is available for the traffic demand, locations of origin and destination are known, but the location of all secondary activities is unknown. We need a generative model for the missing locations. Based on [17] and without further insight on the specific area, it is reasonable to assume that people tend to optimize their routine, and secondary activities tend to be close to home, on the way to the primary activity (work, for example) or close by it.

Hence, the position of the Secondary (S) activity in the chain determines its location relative to the Home (H) and the Primary (P) activity. For example, in the chain $H \rightarrow P \rightarrow H \rightarrow$

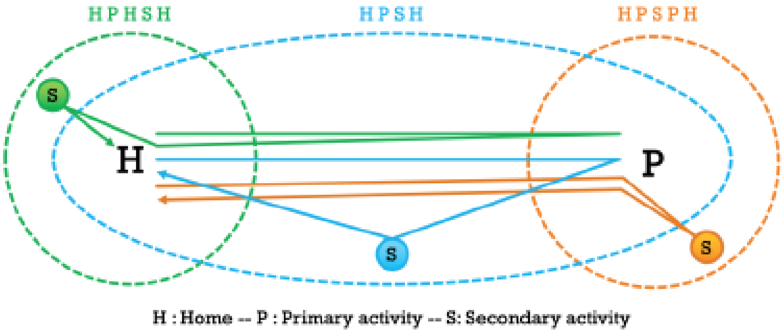


Figure 4: Diagram showing how the location for the Secondary activities is chosen.

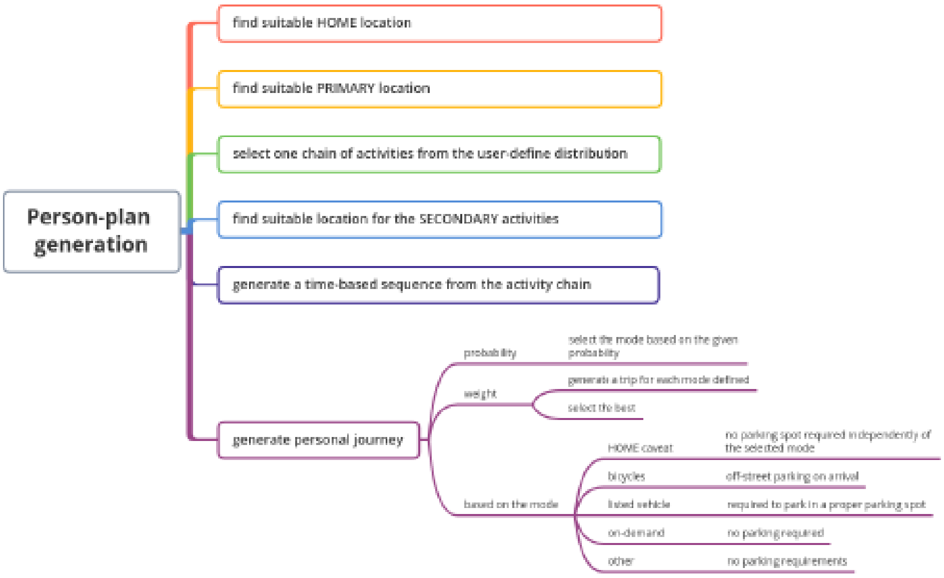

Person-plan generation
find suitable HOME location
find suitable PRIMARY location
select one chain of activities from the user-define distribution
find suitable location for the SECONDARY activities
generate a time-based sequence from the activity chain
generate personal journey
probability
select the mode based on the given probability
weight
generate a trip for each mode defined
select the best
based on the mode
HOME caveat
no parking spot required independently of the selected mode
bicycles
off-street parking on arrival
listed vehicle
required to park in a proper parking spot
on-demand
no parking required
other
no parking requirements

Figure 5: Overview of the implementation choices made during the activity-based mobility generation.

$S \rightarrow H$ (in green), the location of S is chosen to be within the area of the circle centered at H with a parametrized radius. Similarly, in the chain $H \rightarrow P \rightarrow S \rightarrow P \rightarrow H$ (in orange), the location for S is within the parametrized radius of the circle centered at P. Finally, in the chain $H \rightarrow P \rightarrow S \rightarrow H$ (in blue), the location of S is chosen within the area covered by an ellipse with focuses in the locations of H and P respectively. Following the rule, the position of S can be computed for all the possible sequences starting and ending in H, with at least one P, and without two consecutive Ss.

Generation Figure 5 summarizes the algorithm used to generate the trip plan for one person.

1. The Home location is computed based on the given TAZ. The exact location is selected by choosing a building based on its weighted probability, and the street accesses associated with it are saved.

2. The location for the Primary activity is computed following the same principle as the Home location. All the Primary activities in a single chain share this location.

3. The chain of activities associated with the plan is chosen from the distribution defined in the given configuration.

4. Following the sequence of activities, the areas of interest associated to each Secondary activity are computed. The process required to compute the actual location is similar to the one described for the other activities, but in this case, the area of interest, instead of being a TAZ, follows the definition presented in Figure 4.

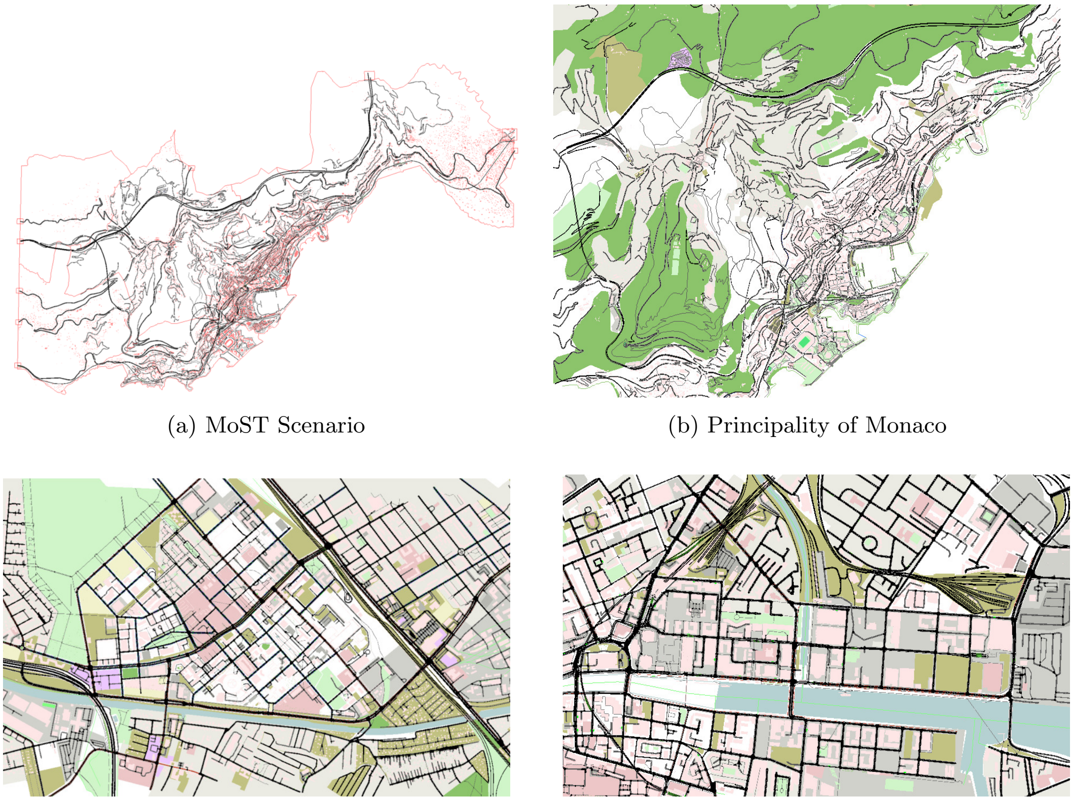
5. An estimation of the time-based sequence of locations is generated using starting times and durations retrieved from the generic activities, where the Estimated Travel Time (ETT) used for the transit journeys is computed by SUMO based on a generic vehicle.

6. Based on the available transportation modes defined for the selected activity chain, the complete plan associated with a person is generated by composing consecutive trips.

    - The trip itself is generated using the TraCI API `findIntermodalRoute`[20] provided by SUMO.
    - In case the parameter associated with the mode is its probability, a single transportation mode is chosen. In case the trip is impossible with the given mode, the journey is discarded.
    - If the parameter associated with the mode is its weight, a trip is generated for all the given modes, their cost is multiplied by the given weight, and finally, the best mode (minimum cost) is selected for the trip. The default value defined in SUMO for the cost of a trip is the ETT. In order to model different user preferences associated with each mode of transport, the cost is multiplied by the given weight, skewing the selection of the best mode, if required.
    - Independently of the transport mode chosen, there is no requirement for parking associated with the Home activity. The assumption is that people have the means to park at home.
    - Each mode presents different parking behaviours:
        - Bicycles are parked off-street on arrival.
        - The vehicles listed in the configuration file that require parking will look for the parking area closest to the destination, leaving SUMO at simulation time to find an alternative parking area in case the given one is full.
        - On-demand vehicles have no parking requirements.
        - Other vehicles, such as emergency, have no parking requirements unless explicitly inserted in the configuration file.

The algorithm described above is used to compute every journey plan for each person in the given population.

[20] SUMO Wiki: findIntermodalRoute API https://sumo.dlr.de/docs/TraCI/Simulation_Value_Retrieval.html#command_0x87_find_intermodal_route Access: March, 2022



(a) MoST Scenario

(b) Principality of Monaco

(c) Adlershof (DE)

(d) Docklands (IE)

Figure 6: Examples of SAGA usage.

Remarks The workflow presented above is independent of the size of the scenario in question. The size of the scenario and the quality of the available information impact the effort required to tune and generate a representative mobility scenario. Usually, the larger the scenario, the more heterogeneous are the datasets that need to be integrated, increasing the complexity of the problem. Nonetheless, the actual workflow does not change, and given the isolation of each step, they can be swapped with ad-hoc extractors and aggregators able to handle the specific set of information that needs to be processed.

5 Use Case: Fast Prototyping

With SAGA, it is possible to generate the first prototype of a mobility scenario automatically, where its quality depends on the input data. It can be used to identify missing or problematic datasets, speeding up the feasibility study and planning required to build a representative multi-modal mobility simulation of a location, being a neighborhood or an entire city.

The workflow presented in this paper was developed while hand-crafting the MoST Scenario [9] (Figure 6a). The MoST Scenario covers an area of approximately 70 km^2 that includes three logical areas and 20 TAZ. It provides the locations of PoIs, more than 100 parking areas, the shape and location for the buildings, and the elevation of buildings and streets. Buses and trains comprise the public transport network, with more than 150 stops and more than 20 routes. Based on previous experience and expert knowledge, to achieve the first working prototype of MoST Scenario, it took approximately 960 hours (one person working for six months). Only afterward, the iterative process of hand-tuning the multi-modal scenario could begin.

Figure 6b shows the initial scenario obtained from OSM for the city-state of Monaco and the surrounding French region. This prototype was built starting from the initial OSM dataset used for MoST Scenario but using SAGA. The automated generation of the working prototype required approximately four hours of computation on a standard computer (processor: Intel$^{®}$ Core^{TM} i7-9750H CPU @ 2.60GHz x 12, RAM: 16 GB). The prototype presents plenty of problems, mostly linked to the lack of precise environmental data for the region and the complex 3D geometry of the infrastructure (due to the presence of mountains and sea, the streets are layered with bridges and tunnels). Nonetheless, it shows how a large-scale scenario can be rapidly generated, and the intermediate configuration files provided by SAGA can be modified step-by-step to achieve a properly hand-tuned scenario representative of the mobility in the area.

Tuning a Multi-modal Scenario Once the prototype is built, by following the workflow presented in Figure 2 (and using the intermediate configurations generated by SAGA) it is possible to find most of the issues that require investigating. The Table 2 lists the steps required to tune a multi-modal scenario. The first column names the tool and discusses the generic issues that can be found in the intermediate outputs, and the second column presents the solution used during the hand-tuning of MoST Scenario.

Fast Mobility Scenario Prototyping Figures 6c and 6d show two mobility scenarios we generated directly from OSM without changing the default parameters. Although these scenarios cannot be considered representative of the real area, due to the use of generic traffic demand, they present an adequate starting point for a feasibility study to achieve a representative multi-modal mobility simulation.

The first one (Figure 6c) is based on Adlershof, a locality south of Berlin in Germany. The initial dataset retrieved from OSM is exceptionally detailed and almost ready to be used, providing an excellent example of a best-case setting for SAGA. We use this OSM dataset to verify SAGA s ability to extract detailed information correctly. Nonetheless, we are aware of multiple issues: (i) broken geometry for multi-modal intersections, (ii) on-street and off-street parking area duplication, (iii) the administrative boundaries are incomplete (the area is quite small). We have not further investigated the existence of external sources of data to tune this scenario.

The second one (Figure 6d) is based on an area called Docklands in Dublin (Ireland). We use this OSM dataset to verify SAGA s ability to build sensible default values when correct and detailed information is not available. This example of left-hand driving presents plenty of problems. Nonetheless, thanks to the fast prototyping enabled by SAGA, the critical issues with intersections, interconnections for the multi-modal transportation, and missing information on the tram and the bus stops have been straightforward to identify. The administrative boundaries (and the associated TAZs) are detailed, and the geometry of both census and election districts are available. Information on parking areas is incomplete and we found duplication and inconsistencies. We started to investigate the presence of additional information to supplement SAGA s configuration files, and for the moment we are looking into the Building City Dashboard

Table 2: Discussion on the intermediate output from SAGA and the hand-tuning required by MoST Scenario as a practical example.

SAGA: Generic Issues	**MoST Scenario**
`netconvert`: Check the network for issues with intersections, public transportation stops and route definitions, location of the on-street parking areas.	Manually changed the geometry of all the roundabouts, updated missing routes and stops for the buses, discarded the on-street parking areas due to incomplete information.
`polyconvert`: Check the polygons for issues with overlapping buildings and missing administrative boundaries.	Manually moved the overlapping buildings, but the administrative boundaries were properly tagged.
`ptlines2flows.py`: Check the default parameters used to generate the schedule and flows for the public transportation.	Manually configured the schedule based on the one posted on the official website.
`generateParkingAreasFromOSM.py`: Check the locations of the parking areas and their capacity.	Manually removed duplicate parking areas, and capacity updated based on external online sources.
`generateTAZBuildingsFromOSM.py`: Check the TAZ extracted from the administrative boundaries. Check both the access to the buildings and their weight.	No additional information on the TAZ was available. Manually moved the access to some buildings based on personal knowledge of the area, but no additional information from the census was available.
`generateAmitranFromTAZWeights.py`: Tune the generic OD-matrix with traffic demand data.	Traffic demand was not available. The OD matrix was hand-tuned with statistical data on average traffic in the area.
`generateDefaultsActivityGen.py`: Tune the default activities and the activity chain distributions.	No additional data based on periodic reports provided by Monaco, nor ad-hoc surveys were available on the activities, the chains, and their probability distributions.

project[21], and more precisely, Dublin Dashboard[22].

Remark The MoST Scenario is available to the community under GPLv3 license, and it can be downloaded from GitHub [23]. This scenario is packaged with SAGA as an example of a hand-tuned scenario based on the work flow previously described, and it can be used as an example to improve the configuration of the activity generation based on a more complex environment.

[21] Building City Dashboard https://dashboards.maynoothuniversity.ie/ Access: March, 2022 Dublin

[22] Dashboard http://www.dublindashboard.ie/ Access: March, 2022

[23] MoST Scenario on GitHub https://github.com/lcodeca/MoSTScenario

6 Conclusion and Future Work

In this paper, we present the SAGA framework, a fully configurable multi-modal scenario generator with mobility based on user-defined activity chains for SUMO. SAGA distinguishes itself from the other mobility generators because it explicitly focuses not only on the activity-based mobility generation but also on the extraction and configuration of the transportation infrastructure and the environmental information required to generate the mobility. The main use-case considered is fast scenario prototyping, where the workflow and toolset associated with SAGA support the feasibility study and planning required to generate a representative mobility scenario from scratch. More precisely, starting from OSM, SAGA is able to automate (i) the extraction of additional infrastructure and environmental features (e.g., parking areas, buildings, and PoIs), (ii) the extraction of generic TAZs based on the administrative boundaries, (iii) the generation of a default configuration for the traffic demand, and (iv) based on default activity chains, the generation of activity-based mobility plans for people. SAGA supports multiple travel modes (walking, bicycles, public transport, on-demand, and user-defined vehicles), and each mode has parametrized parking requirements. Finally, the user-defined chain of activities is parametrized to be flexible and able to represent generic daily routines. SAGA is freely available under the EPLv2 license on GitHub `https://github.com/lcodeca/SUMOActivityGen`, and it is included in the contributed tools since SUMO v1.3.0.

The future of SAGA is focused on improving the automation, and on the implementation of inter-modal mobility based on activity chains. The next step is to automate the extraction of additional infrastructure features, such as bicycle and taxi stands, going in the direction of a definition of a more complex inter-modal mobility hub. Additionally, the SUMO developers are adding additional features for shared rides that we intend to incorporate. Finally, it would be interesting to expand the parametrization of the activities, for example, by adding categories that can be directly associated with building and land use, increasing the level of tuning available.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 713567. ENABLE is funded under Science Foundation Ireland (16/SP/3804) and is co-funded under the European Regional Development Fund.

This work was partially funded by the French Government (National Research Agency, ANR) through the “Investments for the Future”, ref. #ANR-11-LABX-0031-01. EURECOM acknowledges the support of its industrial members, namely BMW Group, IABG, Monaco Telecom, Orange, SAP, ST Microelectronics and Symantec.

References

[1] Li, Yanying and Voege, Tom. Mobility as a service (MaaS): challenges of Implementation and Policy Required. *Journal of Transportation Technologies*, 7(02):95–106, 2017.

[2] Silva, Bhagya Nathali and Khan, Murad and Han, Kijun. Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities. *Sustainable Cities and Society*, 38:697–713, 2018.

[3] Pangbourne, Kate and Stead, Dominic and Mladenovic, Milos and Milakis, Dimitris. The case of mobility as a service: A critical reflection on challenges for urban transport and mobility

governance. In *Governance of the smart mobility transition*, pages 33 48. Emerald Publishing Limited, 2018.

[4] Mooney, Peter and Minghini, Marco and others. A review of OpenStreetMap data. 2017.

[5] Pablo Alvarez Lopez and Michael Behrisch and Laura Bieker-Walz and Jakob Erdmann and Yun-Pang Flotterod and Robert Hilbrich and Leonhard Lucken and Johannes Rummel and Peter Wagner and Evamarie Wießner. Microscopic Traffic Simulation using SUMO. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[6] Toader, Bogdan and Cantelmo, Guido and Popescu, Mioara and Viti, Francesco. Using Passive Data Collection Methods to Learn Complex Mobility Patterns: An Exploratory Analysis. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 993 998. IEEE, 2018.

[7] S. Yeung, H. M. A. Aziz, and S. Madria. Activity-Based Shared Mobility Model for Smart Transportation. In *2019 20th IEEE International Conference on Mobile Data Management (MDM)*, pages 599 604, June 2019.

[8] Bowman, John L and Ben-Akiva, Moshe E. Activity-based disaggregate travel demand model system with activity schedules. *Transportation research part a: policy and practice*, 35(1):1 28, 2001.

[9] Lara Codeca and Jerome Harri. Monaco SUMO Traffic (MoST) Scenario: A 3D Mobility Scenario for Cooperative ITS. In *SUMO 2018, SUMO User Conference, Simulating Autonomous and Intermodal Transport Systems*, Berlin, GERMANY, 05 2018.

[10] Saxena, Pratiksha and Choudhary, Abhinav and Kumar, Sanchit and Singh, Satyavan. Simulation Tool for Transportation Problem: TRANSSIM. In *Problem Solving and Uncertainty Modeling through Optimization and Soft Computing Applications*, pages 111 130. IGI Global, 2016.

[11] Horni, Andreas and Nagel, Kai and Axhausen, Kay W. *The multi-agent transport simulation MATSim*. Ubiquity Press London, 2016.

[12] Adnan, Muhammad and Pereira, Francisco C and Azevedo, Carlos Miguel Lima and Basak, Kakali and Lovric, Milan and Raveau, Sebastian and Zhu, Yi and Ferreira, Joseph and Zegras, Christopher and Ben-Akiva, Moshe. SimMobility: A multi-scale integrated agent-based simulation platform. In *95th Annual Meeting of the Transportation Research Board Forthcoming in Transportation Research Record*, 2016.

[13] Schweizer, Joerg and Rupi, Federico and Filippi, Francesco and Poliziani, Cristian. Generating activity based, multi-modal travel demand for SUMO. *EPiC Series in Engineering*, 2:118 133, 2018.

[14] Basu, Rounaq and Araldo, Andrea and Akkinepally, Arun Prakash and Nahmias Biran, Bat Hen and Basak, Kalaki and Seshadri, Ravi and Deshmukh, Neeraj and Kumar, Nishant and Azevedo, Carlos Lima and Ben-Akiva, Moshe. Automated mobility-on-demand vs. mass transit: a multi-modal activity-driven agent-based simulation approach. *Transportation Research Record*, 2672(8):608 618, 2018.

[15] Zehe, Daniel and Nair, Suraj and Knoll, Alois and Eckhoff, David. Towards CityMoS: A Coupled City-Scale Mobility Simulation Framework. *5th GI/ITG KuVS Fachgesprach Inter-Vehicle Communication*, 2017:03, 2017.

[16] Xu, Yadong and Aydt, Heiko and Lees, Michael. SEMSim: A distributed architecture for multi-scale traffic simulation. In *Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*, pages 178 180. IEEE Computer Society, 2012.

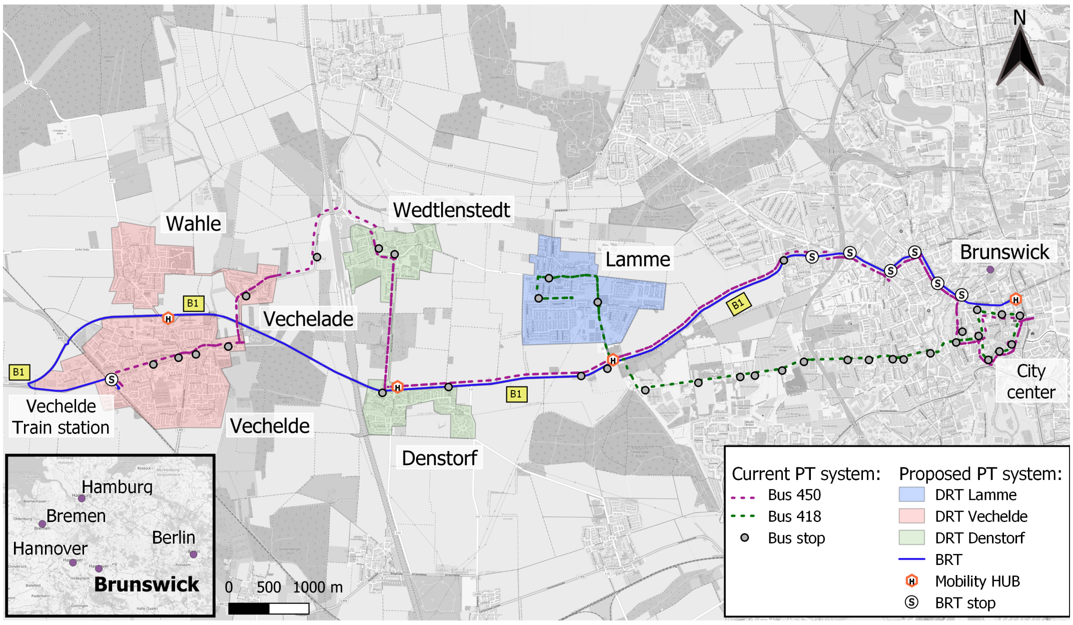
[17] Cantelmo, Guido and Viti, Francesco. Incorporating activity duration and scheduling utility into equilibrium-based Dynamic Traffic Assignment. *Transportation Research Part B: Methodological*, 126:365 390, 2019.

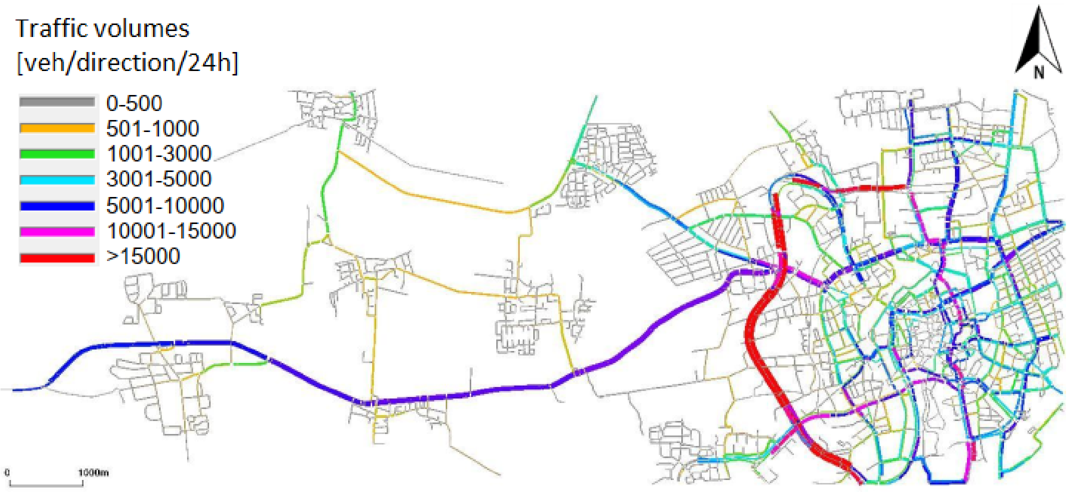
SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.101

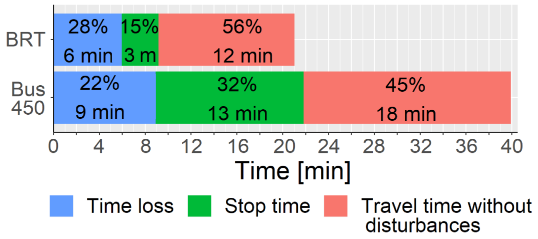
Published: 28 Jun. 2022

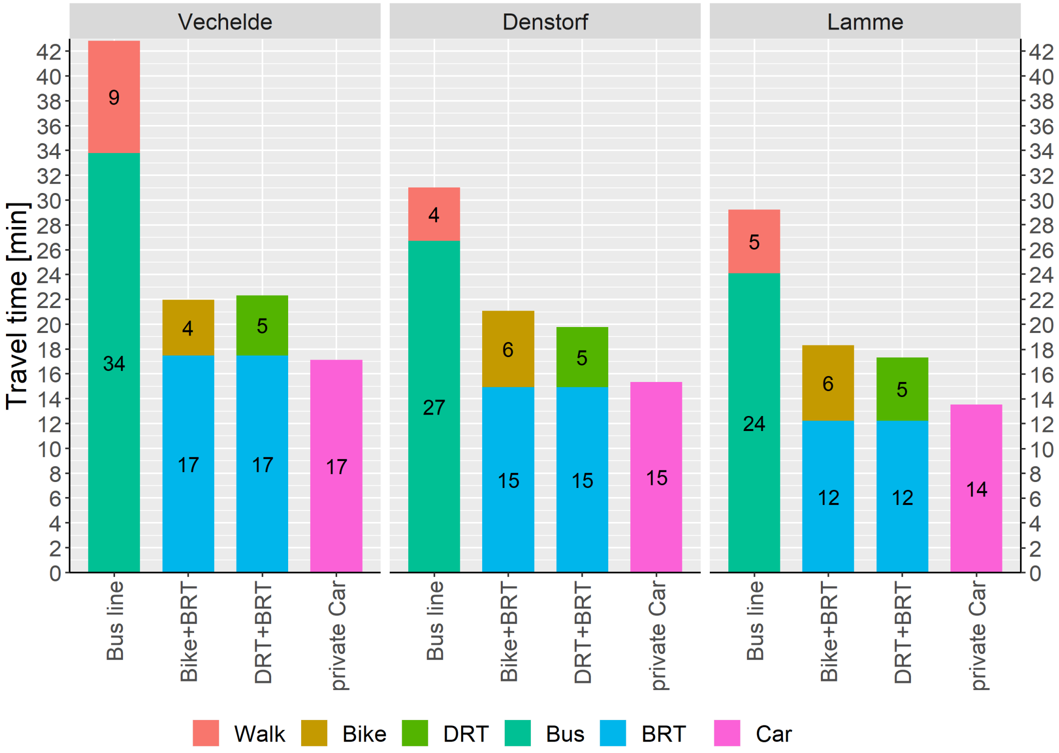
maria.armellini@dlr.de

laura.bieker@dlr.de


N
Wahle
Wedtlenstedt
Lamme
Brunswick
Vechelade
B1
Vechelde
Train station
Vechelde
Denstorf
City
center
Hamburg
Bremen
Berlin
Hannover
Brunswick
0 500 1000 m
Current PT system:
Bus 450
Bus 418
Bus stop
Proposed PT system:
DRT Lamme
DRT Vechelde
DRT Denstorf
BRT
Mobility HUB
BRT stop


Traffic volumes
[veh/direction/24h]
0-500
501-1000
1001-3000
3001-5000
5001-10000
10001-15000
>15000
N
0
1000m


BRT
28%
6 min
15%
3 m
56%
12 min
Bus 450
22%
9 min
32%
13 min
45%
18 min
0
4
8
12
16
20
24
28
32
36
40
Time [min]
Time loss
Stop time
Travel time without disturbances


Vechelde
Denstorf
Lamme
Travel time [min]
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40
42
9
34
4
17
5
17
17
4
27
6
15
5
15
15
5
24
6
12
5
12
14
Bus line
Bike+BRT
DRT+BRT
private Car
Walk
Bike
DRT
Bus
BRT
Car

-

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.102

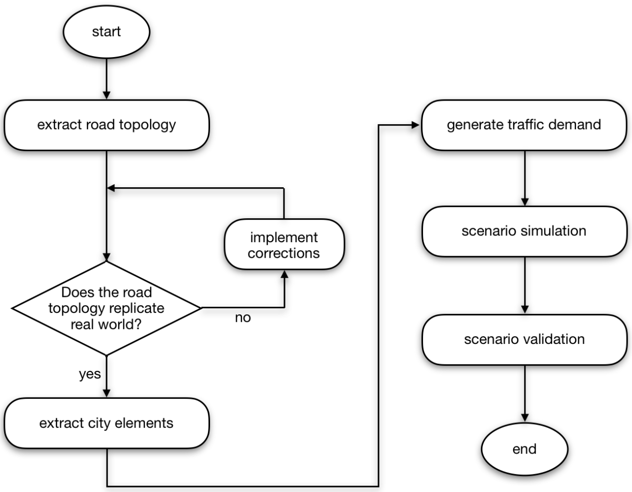
Published 28 Jun. 2022

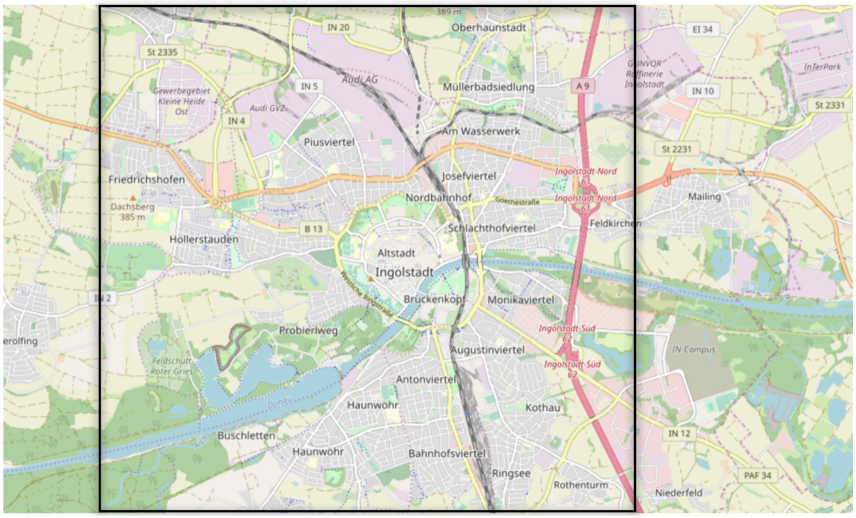
silascorreia.lobo@carissma.eu

stefan.neumeier@thi.de

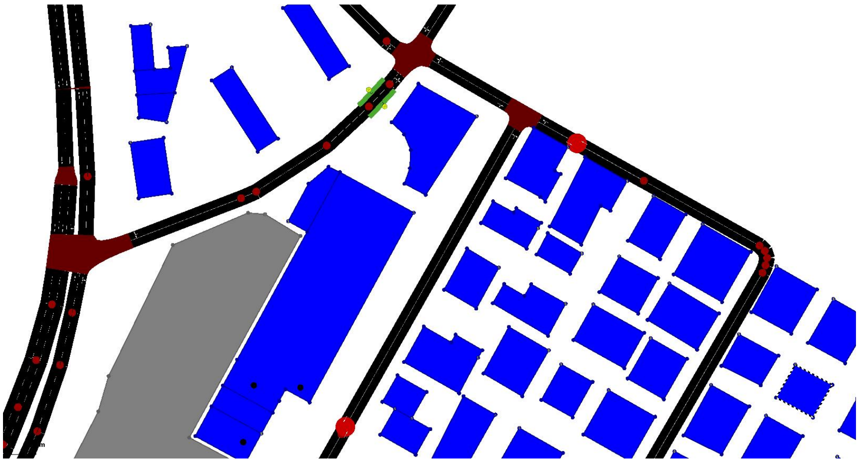
evelio@eletrica.ufpr.br

christian.facchi@thi.de

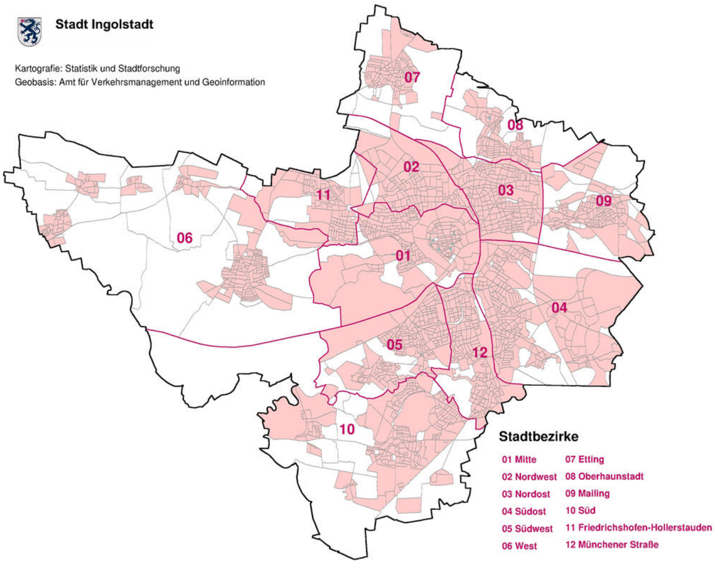

start
extract road topology
implement corrections
Does the road topology replicate real world?
no
yes
extract city elements
generate traffic demand
scenario simulation
scenario validation
end

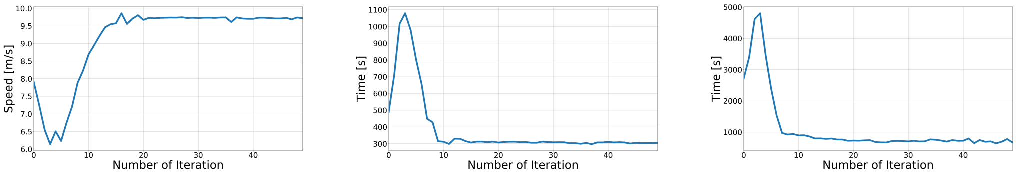

Oberhaunstadt
Müllerbadsiedlung
Am Wasserwerk
Audi AG
Audi GVZ
Gewerbegebiet Kleine Heide Ost
Piusviertel
Friedrichshofen
Dachsberg 385 m
Hollerstauden
Josefviertel
Nordbahnhof
Schlachthofviertel
Feldkirchen
Altstadt
Ingolstadt
Brückenkopf
Monikaviertel
Probierlweg
Augustinviertel
Antonviertel
Haunwöhr
Kothau
Buschletten
Bahnhofsviertel
Ringsee
Rothenturm
Niederfeld
Mailing
IN-Campus
InTerPark
GUNVOR Raffinerie Ingolstadt
Ingolstadt-Nord
Ingolstadt-Süd
Feldschutt Roter Gries
Donau
A 9
B 13
IN 20
IN 5
IN 4
IN 2
IN 10
IN 12
St 2335
St 2231
St 2331
EI 34
PAF 34
erolfing

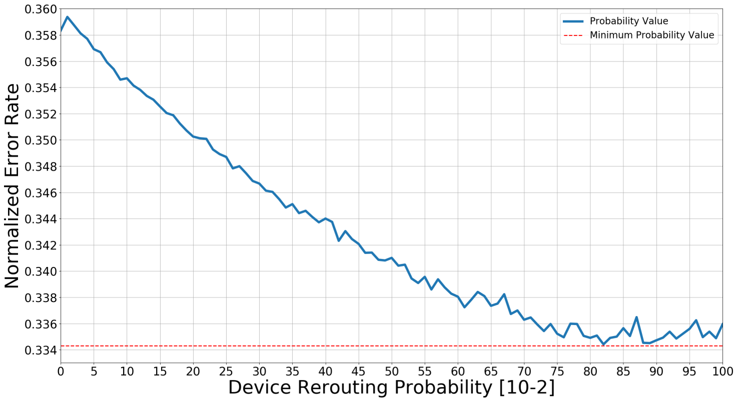
Roßmühlstraße
Schloßla
kplatz

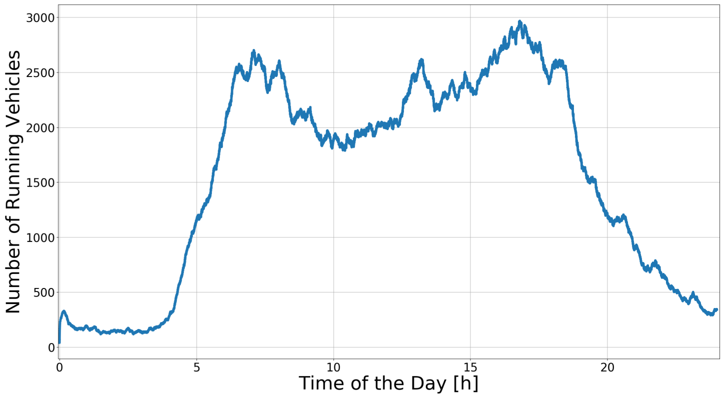


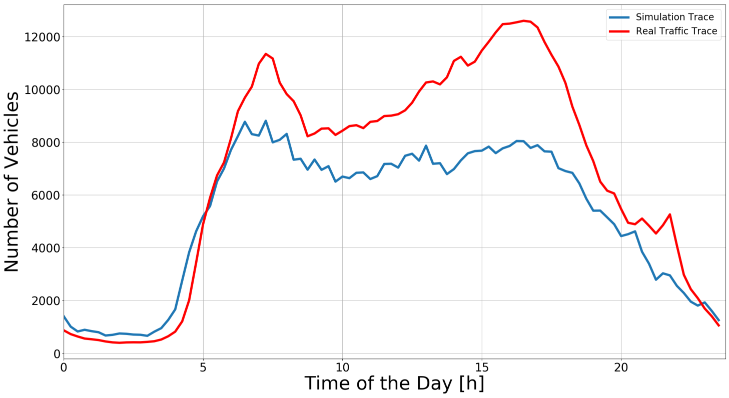
https://sumo.dlr.de/userdoc/Demand/Activity-based_Demand_Generation.html

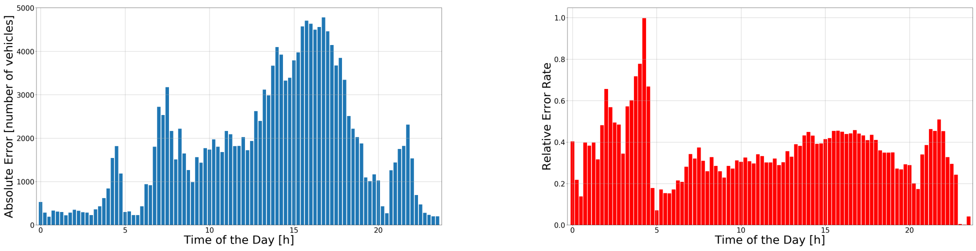

Stadt Ingolstadt
Kartografie: Statistik und Stadtforschung
Geobasis: Amt für Verkehrsmanagement und Geoinformation
07
08
02
03
09
11
06
01
04
05
12
10
Stadtbezirke
01 Mitte
02 Nordwest
03 Nordost
04 Südost
05 Südwest
06 West
07 Etting
08 Oberhaunstadt
09 Mailing
10 Süd
11 Friedrichshofen-Hollerstauden
12 Münchener Straße

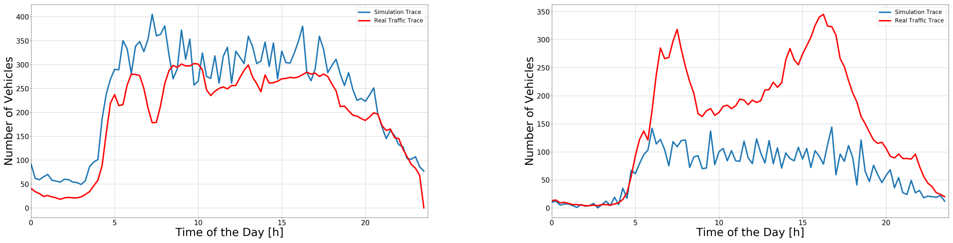

Speed [m/s]
10.0
9.5
9.0
8.5
8.0
7.5
7.0
6.5
6.0
0
10
20
30
40
Number of Iteration
Time [s]
1100
1000
900
800
700
600
500
400
300
0
10
20
30
40
Number of Iteration
Time [s]
5000
4000
3000
2000
1000
0
10
20
30
40
Number of Iteration


Normalized Error Rate
0.360
0.358
0.356
0.354
0.352
0.350
0.348
0.346
0.344
0.342
0.340
0.338
0.336
0.334
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
Device Rerouting Probability [10-2]
Probability Value
Minimum Probability Value


Number of Running Vehicles
0
500
1000
1500
2000
2500
3000
0
5
10
15
20
Time of the Day [h]


Simulation Trace
Real Traffic Trace
Number of Vehicles
0
2000
4000
6000
8000
10000
12000
0
5
10
15
20
Time of the Day [h]


Absolute Error [number of vehicles]
0
1000
2000
3000
4000
5000
Time of the Day [h]
0
5
10
15
20
Relative Error Rate
0.0
0.2
0.4
0.6
0.8
1.0
Time of the Day [h]
0
5
10
15
20


Number of Vehicles
0
50
100
150
200
250
300
350
400
Simulation Trace
Real Traffic Trace
Time of the Day [h]
0
5
10
15
20
Number of Vehicles
0
50
100
150
200
250
300
350
Simulation Trace
Real Traffic Trace
Time of the Day [h]
0
5
10
15
20

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.103

Published: 18 Jun. 2022

Pre-study and insights to a sequential MATSim-SUMO tool-coupling to deduce 24h driving profiles for SAEVs

Henriette Triebke, Markus Kromer , and Peter Vortisch

[illegible]

henriette.triebke@de.bosch.com

[illegible]

peter.vortisch@kit.edu

Abstract

New mobility concepts such as shared, autonomous, electric vehicle (SAEV) fleets raise questions to the vehicles' technical design. Compared to privately owned human driven cars, SAEVs are expected to exhibit different load profiles that entail the need for newly dimensioned powertrain and battery components. Since vehicle architecture is very sensitive to operating characteristics, detailed SAEV driving cycles are crucial for requirement engineering. As real world measurements reach their limit with new mobility concepts, this contribution seeks to evaluate three different traffic simulation approaches in their ability to model detailed SAEV driving profiles. (i) The mesoscopic traffic simulation framework MATSim is analyzed as it is predestined for large-scale fleet simulation and allows the tracking of individual vehicles. (ii) To improve driving dynamics, MATSim's simplified velocity profiles are enhanced with real-world driving cycles. (iii) A sequential tool-coupling of MATSim with the microscopic traffic simulation tool SUMO is pursued. All three approaches are compared and evaluated by means of a comprehensive test case study. The simulation results are compared in terms of driving dynamics and energy related key performance indicators (KPI) and then benchmarked against real driving cycles. The sequential tool-coupling approach shows the greatest potential to generate reliable SAEV driving profiles.

1 Introduction

SAEV load profiles and technical requirements are expected to differ fundamentally from conventional private cars. While the latter feature (a) small daily mileages, (b) long times of non-use, (c) high driving ranges and (d) have access to a dense refueling infrastructure, SAEV operating characteristics are rather opposite when used for urban passenger transport. Higher daily mileages and shorter (battery-limited) driving ranges entail the need for frequent recharging. This, however, is counteracted economically by the request for little idling times and technically by long charging durations within a comparatively thin network of charging stations. As the complexity of vehicle development increases, detailed SAEV driving profiles become more and more important for virtual prototype testing. For this purpose, they need to meet the following **key requirements**: **(KR1)** The profiles need to mirror the vehicles' movement throughout entire metropolitan areas for 24 hours accounting for all range and charging constraints as well as for different routing, dispatching and pricing strategies. **(KR2)** They need to provide information on the vehicles' states such as *idling*, *relocating*, *charging* or *occupied* to enable optimal climate control or battery preconditioning. **(KR3)** The driving cycles must be accurate enough to derive reasonable velocity profiles that reflect for autonomous driving, road congestion and diverse transport infrastructures. **(KR4)** Depending on the road network's topography or the driving cycle's purpose, further time-series such as altitude or occupancy profiles are also of interest. To the authors' best knowledge, the problem of deriving representative SAEV driving profiles that meet all above stated requirements has not been tackled by the scientific community yet. There are many publications that deal with conventional driving cycle generation, the modeling of autonomous driving behavior or large-scale SA(E)V fleet simulation. However, no holistic approach is known that combines all three areas.

For the automotive industry **driving cycles** play a major role in state-of-the-art emission modeling, performance prediction and virtual prototype testing. Driving cycles most commonly designate second-by-second time-velocity profiles and can be distinguished in *modal* and *transient* cycles. Modal cycles are highly simplified and consist of different idling, straight acceleration and steady speed phases. They often feature unrealistic dynamics in the transition zones [1, 9, 24]. Transient cycles in contrast, reflect real-life driving behavior under on-road conditions [12]. A common technique to derive new driving cycles comprises four steps: **route choice, data collection, data clustering and cycle generation** [1, 40, 43]. Route choice involves selecting the route on which data are to be collected. The driving data are gathered by means of *on-board measurement*, *GPS-tracking* and/or *chase car method*. As stated in [43], on-road measurements reflect the selected route most accurately but feature a strong bias due to unusual congestion pattern which entails the need for repetitive measurements. The chase car method is less cost-intensive and involves randomly following target vehicles by imitating their driving behavior. This approach, however, comes at the price of route choice. The collected profiles are often decomposed into micro trips[1] which are clustered according to traffic condition, vehicle type or other KPI. Common trip clustering techniques are *k-means cluster algorithms* [15, 41] or hybrid approaches of *k-means* and *support vector machine (SVM)* clustering [43]. Despite their validity, cluster methods often require large computational resources [1]. The final cycle is typically constructed from a pool of available micro trips [1, 40, 43]. The idea of the **micro-trip-based methodology** is to find those micro trips that reflect the diversity of real world driving well enough but in a more compressed manner to be practical and cost effective [1, 40]. Generally, the micro trips are selected by algorithms based on predefined performance measures. Alternatively, *Monte Carlo engines* serve to generate multiple candidate cycles by randomly picking several micro trips and determining their KPI. The best fit in performance is then finally chosen.

Another statistical approach consists in using real world driving databases to generate synthetic driving cycles by means of **Markov chain processes**. As done in [18, 35, 36], the measured velocity profiles serve to construct a transition probability matrix of a Markov chain. At this, each matrix element corresponds to a certain state (denoted by current velocity and acceleration) and within each state, the transition probabilities to jump from one state to another are stored. Yet another data-driven approach of driving cycle deduction is referred to as **route information mapping**. A new concept of defining automotive driving cycles is introduced in [12] by stressing the need to incorporate external conditions such as weather, traffic and terrain data. This is also done in [16] by joining data on slope, road curvature and speed

[1] A micro trip denotes a trip between two idling phases.

limit with traffic information and driver models to form a control problem that is numerically solved to generate velocity profiles. However, prerequisites for such data-driven approaches are (a) large databases of GPS-tracked driving cycles, (b) detailed maps and/or (c) access to traffic information.

As all previous methods rely on measured or historical data, they are not suited to deduce driving cycles for future autonomous vehicles. **Autonomous driving behavior** is often approached by applying **filter or smoothing techniques** on human driven profiles [2, 19, 28]. In this context, the smoothing approach is justified by *kinetosis prevention*[2] on the one hand and by the vehicle's improved perception on the other hand. Advanced sensors and car2x-communication will enable autonomous vehicles (AVs) to respond more smoothly to ambient traffic conditions. However, smoothing techniques tend to annihilate idling times and cannot reflect for platooning effects or connected driving in a methodologically sound manner.

As conventional approaches to deduce representative driving cycles reach their limit with new mobility concepts, **microscopic traffic simulation** became increasingly popular in this regard. Microscopic frameworks have been used for cost-optimized driving cycle deduction [1] and to assess the impact of automated driving on fuel consumption [10, 21, 37]. In [21], the capability of VISSIM[3] to model real world driving cycles is evaluated. Compared to human-driven cycles, the simulated profiles fit well in aerodynamic speed but poor in acceleration: human drivers tend to have higher acceleration rates at lower speeds and the simulation neglects stochastic oscillations around the target velocity. Similar conclusions are drawn in [1] which combines microscopic traffic simulation and micro-trip-based methods to deduce representative driving cycles. According to the authors, default parameters from micro-simulation produce **unrealistic driving behavior**: simulated velocity profiles are too aggressive as their gradients are often set to the vehicle's maximum capability. This is also evidenced in [37] by emphasizing that the driving cycles' quality is directly tied to a well calibrated traffic model. Due to the same reason, the relevance of microscopic traffic models for estimating the impact of traffic strategies on fuel consumption is questioned in [10]. The authors pinpoint the fact that **microscopic traffic simulation models have a validation problem** when driving dynamics are concerned: even though they produce detailed velocity profiles, microscopic traffic models are usually designed to meet macroscopic objectives such as signal timing or transportation planning. Consequently they are calibrated by traffic flow parameters like speed, density or queue length rather than instantaneous speed and acceleration [10, 37]. Thus, speed profiles are often too simplified and therefore might not be applicable for environmental studies or requirement engineering. However, even though microscopic traffic simulation tools have weaknesses in capturing human driving behavior, they are likely to cope well enough with fully automated driving as fewer stochastic terms are involved.

There is plenty of literature dealing with the **acceptance, simulation and impact of autonomous vehicle fleets**. For one thing, AV fleets are expected to improve network capacity due to connected driving and improved safety [34]. Then again, AVs may also increase traffic volumes due to induced travel demand arising from improved travel comfort, additional empty rides and smaller vessel sizes in contrast to public transport means [22]. Due to their disruptive character, AV fleet simulations have been analyzed from many different perspectives. In this context, especially the mesoscopic *Multi-Agent Transport Simulation framework*[4] (MATSim) [20]

[2] To ensure the passenger's well being, the lateral and longitudinal acceleration is limited.
[3] https://www.ptvgroup.com/de/loesungen/produkte/ptv-vissim/
[4] https://www.matsim.org/

is well established. In [4] and [14], for example, the city-wide replacement of private cars with shared autonomous vehicle (SAV) fleets is simulated for Berlin and Austin. Both studies conclude that each SAV could potentially replace ten privately owned cars. Further contributions evaluate the impact of different SAV pricing schemes on mode choice [23, 27] or deal with SAV electrification and its implication for charging infrastructure planning [5, 8, 29, 42]. The influence of routing and dispatching algorithms on taxi services are extensively discussed in [7, 31, 32]. However, even though MATSim has its strong points in large-scale fleet simulation, mesoscopic traffic simulation tools generally lack the necessary level of detail to simulate reasonable dynamics of individual vehicles [38].

To conclude, there are numerous publications dedicated to partial solutions but as those approaches are often too narrow in their objective, they either lose viability or lack feasibility in a broader context. This contribution seeks to elaborate an overall concept to deduce **representative 24h SAEV driving cycles that meet all above stated key requirements**. To this end, three different traffic simulation approaches are evaluated and discussed. To gain deeper insights in terms of large-scale feasibility, the methods are applied to a set of test cases. To reduce modeling effort, several simplifications are made: the pre-study has no fleet character yet, nor does it reflect for autonomous driving behavior. These limits, however, do not affect this study's validity: The main objective at this stage is to quantify the approaches' suitability by means of different evaluation criteria, such as (a) their ability to model detailed driving dynamics, (b) their capability to simulate large-scale areas and (c) the approaches' feasibility in terms of data availability and automation capacity **(KR5)**.

2 Methodological approach

This section serves to outline each of the three simulation approaches in more detail as their understanding is essential for the test case analysis in Section 3.

2.1 MATSim's capabilities and limits in drive cycle deduction

MATSim is a open-source framework for large-scale, agent-based traffic simulation. Its traffic assignment relies upon a co-evolutionary algorithm where so-called agents optimize their daily activity schedules in an iterative fashion by varying their initial departure time, transport mode or route choice to maximize their personal benefit. At this, they compete with other agents for space-time resources in the transportation network until a quasi equilibrium state is reached[5]. MATSim allows the deduction of vehicle trajectories and status profiles by design. Every action an agent performs – such as entering or leaving a certain road segment (*link*) – is recorded. Based on this information, daily status and speed profiles can easily be derived as exemplarily shown in Figure 1. However, as MATSim uses a simplified queue model to approximate traffic dynamics, the framework does not provide any reasonable information on a vehicle's position on a link itself. Only average link-speeds can be extracted. The queue model further leads to limitations in congestion modeling [3] as the tool's primary purpose is to simulate large scenarios in decent time which requires simplifications in traffic and driving dynamics. As the understanding of those shortcomings is essential for this work, a brief recap of MATSim's traffic dynamics is given next. MATSim relies on the discrete cell transmission model (CTM) [11] and the queuing model described in [17]. In the CTM, the length of the homogeneous network cells

[5] For more information on the user equilibrium, replanning process or plan scoring please refer to [20].

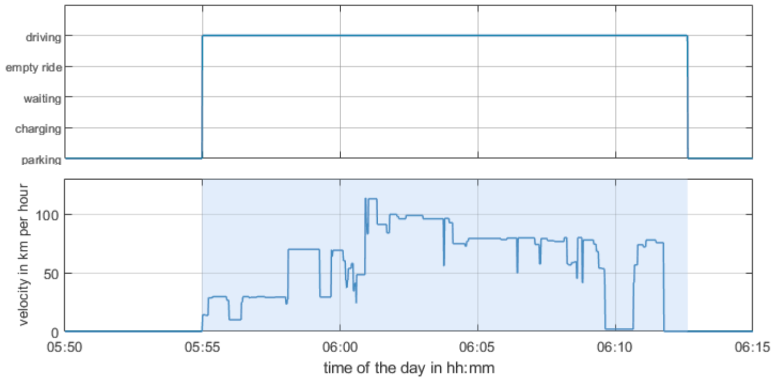


Figure 1: Status (top panel) and average link speed profile (bottom) for a chosen MATSim car

is defined by the distance a vehicle travels in the time step at free-flow velocity. As defined in Equation 1, the number of vehicles in a cell depends on the number of cars in that cell at the previous time step and the difference of inflowing and outflowing vehicles.

(1)

Here, the vehicles' movement into the cell is limited by three restrictions as depicted in Equation 2, where the *flow capacity* represents the maximum number of vehicles allowed to enter a cell and the *storage capacity* the cell's capacity to store vehicles.

min with (2)

With the improved queue model by [17], the road network is represented by so-called *links* of different length instead of homogeneous cells. Additionally, *priority queues* are introduced in MATSim that sort vehicles on a link according to their order of entrance or earliest exit time.

Under certain conditions MATSim's queue model leads to false congestion patterns and therefore misleading vehicle dynamics especially on short links or in sample runs[6]. The flow capacity basically acts like a batch system: A flow capacity of 600 cars/h means that only every sixth second a vehicle is allowed to leave a link. Otherwise the exit is blocked. Consequently, newly arriving vehicles queue up on the link and wait for their turn to leave which sometimes leads to unrealistic long passing times. Consider, for example, two subsequent vehicles on a 15 m link: even with a free flow velocity of 50 km/h the rear car would need at least 6 s to pass 15 m as the exit is blocked this long by the first vehicle. The *stucktime parameter*[7] complicates this even further as it temporarily allows a car surplus on a link: 10 % sample runs reveal vehicle

[6] Sample runs increase computational performance, as only a subset of agents is simulated. In a 10 % run for example, each simulated vehicle gets the weight of ten and therefore occupies a net-space of 75 m on the network (the default vehicle length in a 100 % sample is 7.5 m) [20]. To preserve traffic dynamics, the flow and storage capacities are adjusted accordingly and multiplied by a factor [illegible] .

[7] To counteract gridlocks, the [illegible] has been introduced to bypass the storage capacity constraint in case the first vehicle in the queue is stuck too long. In doing so, a minimal flow even under very congested traffic conditions is maintained [20].

queues of 300 m length on a single link 10 m long. At this, the second vehicle needs at least 1 min to pass the link, the third a minimum of 2 min and the third even 3 min[8]. Technically, even four vehicles of weight 100 (which sum up to a queue of 3 km) can be enforced to stand on a single short link without throwing an error. It has to be stressed at this point, that under those circumstances the queues do *not* line up on upstream links, which hinders MATSim to model spatial congestion patterns in detail (even though they might be correct on a pure temporal level as the flow capacity has it's methodical legitimacy).
To conclude, short links act as temporary vehicle sinks, storing too many vehicles which otherwise would have spilled back in upstream links. Consequently, the average link-speed profiles are faulty under congested traffic conditions as they often show average link speeds near zero on short links but nearly free flow velocities on links prior to those error-prone short links.

2.2 MATSim drive cycle enrichment with real-world driving profiles

To improve driving dynamics, MATSim's average link speed profiles are enhanced with synthetic and real-world driving cycles. For this, five different driving cycles are chosen that mirror a wide range of driving maneuvers and road types. All together, they account for a total driving time of 228 min. The cycles' normalized velocity and acceleration distributions are given in

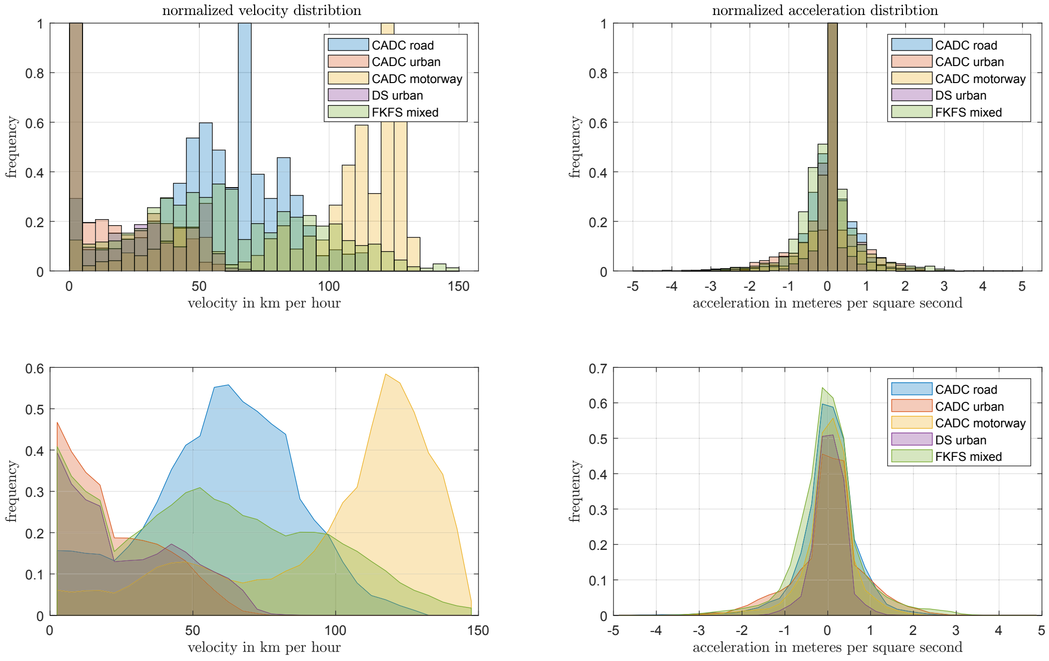


Figure 2: Normalized velocity (left) and acceleration distribution (right) of the considered drive cycles. All distributions are normalized to their local maximum and flattened by moving average.

[8] Given a nominal flow capacity of 600 cars/h which corresponds to 60 cars/h in a 10% sample run.
[9] Common Artemis Driving Cycles (CADC): https://dieselnet.com/standards/cycles/artemis.php
[10] The *DS urban* is a RB-internal cycle through Stuttgart city used for load collective deduction.
[11] The FKFS cycle was conceived by the *Research Institute of Automotive Engineering and Vehicle Engines Stuttgart* as representative driving cycle for the Stuttgart region.

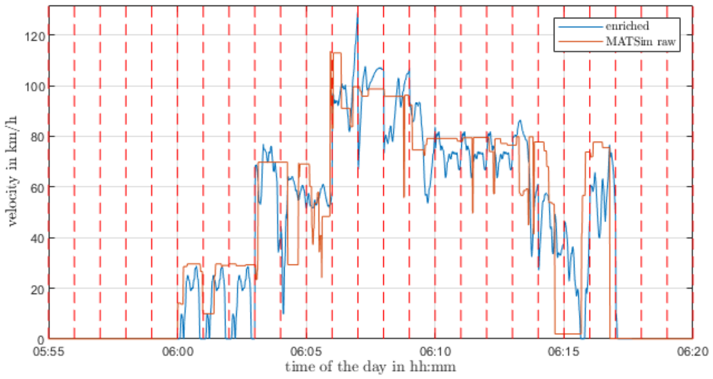


Figure 3: Exemplary representation of an enriched velocity profile without improvement measures. The red dashed lines indicate the transition of two consecutive 1 min-segments.

approach, an algorithm goes through all MATSim segments and identifies the CADC/DS/FKFS segment with the lowest discrepancy in average speed without bothering about unrealistic driving dynamics in the transition zones. In case the maximum speed limit of the MATSim segment is lower than the corresponding tabulated one, the segment with the next best fit in average speed is chosen. This prevents congested motorway cycles from being mixed into urban MATSim profiles. Figure 3 displays the outcome of this approach. At this, the orange and blue line represent MATSim's simplified and enriched profile respectively. As expected, the latter looks more realistic, but still features unrealistic acceleration rates between consecutive segments that require further improvement: (i) As discussed in Section 2.1, MATSim often features velocities near zero on short links. As those are hard to match with real driving cycles, the average speed of those 1 min-segments is set to zero if ¯ km/h. (ii) To make up for the lost distance, the chosen synthetic driving cycles are allowed to exceed MATSim's speed limit by 20 %. This is further justified by the fact that real world drivers tend to overspeed as well. (iii) Moreover, acceleration rates in the transition zones are limited to realistic values. If the acceleration exceeds m/s , the identified CADC/DS/FKFS segment is discarded and a better one is iteratively chosen. The so generated profile is considered acceptable if the daily traveled distance of both profiles $_{\text{raw}}$ and $_{\text{enr}}$ have a relative error of less than 5 %. The relative error $^{\text{day, rel}}_{\text{veh}}$ is calculated as follows

$$ {}^{\text{day, rel}}_{\text{veh}} \qquad \frac{{}_{\text{enr, veh}} \qquad\qquad {}_{\text{raw, veh}}}{{}_{\text{raw, veh}}} \tag{3}$$

where is the vehicle's identification number, the maximum of 1440 1 min-segments per day and the total of 60 s per minute.

2.3 Microscopic drive cycles from sequential tool-coupling

Another approach to enhance MATSim's speed profiles consists in subjecting the simulated vehicle trajectories to an additional microscopic traffic simulation. In this context, *Simulation* *of Urban Mobility* (SUMO)[30] constitutes a rather natural choice as it is the most popular open-source microscopic traffic simulation framework[12]. SUMO is well established in the fields of traffic management, traffic light evaluation and (in recent years) the simulation of vehicular communications. It provides many interfaces that allow external applications to interfere online with the traffic simulation. In this work, the *Traffic Control Interface* (TraCI) is used to retrieve and instantaneously manipulate object attributes.

Network generation To build a SUMO network based on an existing MATSim model, the geographical area of interest is independently imported from *OpenStreetMap* with SUMO *NETCONVERT*[13]. Network differences in MATSim and SUMO are exemplarily depicted in Figure 4 for the *Bergheimer Steige* in Stuttgart. In MATSim, networks can be imported via the *OsmNetworkReader* with varying degree of resolution, e.g. rather simple networks with reduced number of links (4b) or more complex ones which account more accurately for curved road shapes (4c)[14]. In general, it can be noted that MATSim paths (regardless of their import resolution) already account for corrective measures for road geometry and altitude differences. Consequently, the path lengths fit rather well in direct comparison with *GoogleMaps*. SUMO networks in contrast, feature the most sophisticated network design but additional length gains by altitude differences are not projected to the 2-dimensional network by default. In our work, those data are loaded from an additional elevation model.

Travel demand transfer MATSim-SUMO The travel demand in our SUMO simulation comes entirely from MATSim. For that purpose, all MATSim links bordering a chosen test case are identified. Next, all vehicles passing those links are recorded during MATSim simulation with (i) *vehicleID*, (ii) vehicle route and (iii) time of test case entrance and exit. In case of a MATSim sample run, the travel demand in SUMO is upscaled accordingly by injecting *cloned* vehicles. To prevent severe gridlocks in SUMO, a random time offset (sampled from a Gaussian distribution) is added to the network entering time of the cloned cars. Having all departure

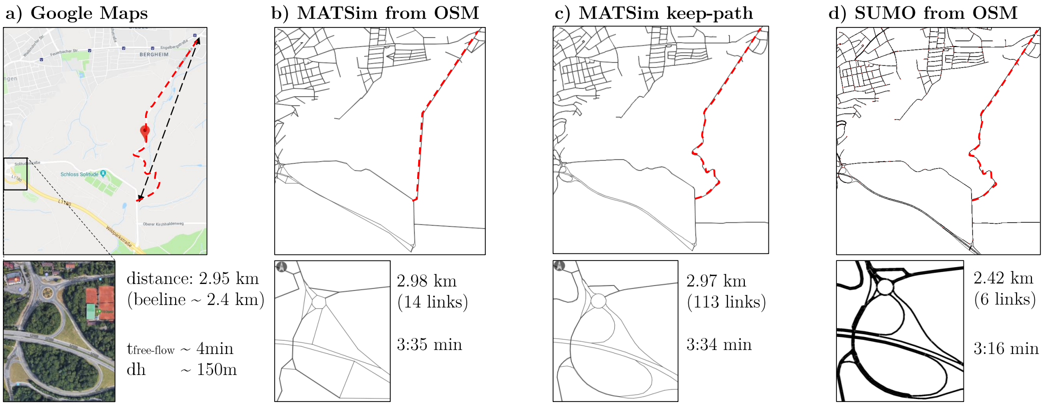


Figure 4: Network differences based on the Bergheimer Steige test case as defined in Section 3.

[2] https://sumo.dlr.de/docs/index.html

[3] In principle, SUMO networks can also be imported from MATSim. This proceeding, however, proved not beneficial for our purpose as MATSim discards some network information which is required in SUMO.

[4] This network, however, behaved poorly in our simulation, as it has too many short links where the artefacts discussed in Section 2.1 occur.

times settled, the *linkIDs* from MATSim are translated into corresponding *edgeIDs* in SUMO to write the final trips-file for SUMO simulation. Finally, all *vehicle trips* in SUMO are converted in *vehicle routes* by *DUAROUTER*. Due to the small test case sizes in this work (see Sect. 3), the SUMO routes match with those in MATSim.

Microscopic traffic dynamics Traffic dynamics in SUMO are realized by car-following models (such as *Krauss* [26] or *Intelligent Driver Model (IDM)* [39]) and lane-change models (such as *LC2013* [13]). In this contribution we use the default Krauss-model according to which the vehicles drive as fast as possible while maintaining a perfect safety distance to the leading car. The safe speed is computed as follows [25]:

$$_{\text{safe}} \qquad \frac{\qquad\qquad}{\bar{\ }} \tag{4}$$

where represents the speed of the leading vehicle, the gap to the leader, the reaction time, the maximum deceleration of the follower and $\bar{\ }$ the mean velocity of following and leading vehicle. As $_{\text{safe}}$ may exceed the legal speed limit of the road or surpass the vehicle's capability, the actual targeted velocity is limited to the minimum of those three. On top of that, a *driver imperfection* has been introduced in SUMO that causes random deceleration to model speed fluctuations that lead to spontaneous jams at high traffic densities. Furthermore, each vehicle draws an individually chosen from a normal distribution to represent a wider variety of human driving styles, e.g. drivers that notoriously stay above or below the legal speed limit. Figure 5 displays an exemplary velocity profile extracted from SUMO simulation by also providing information on the current speed limit and the vehicle's elevation profile.

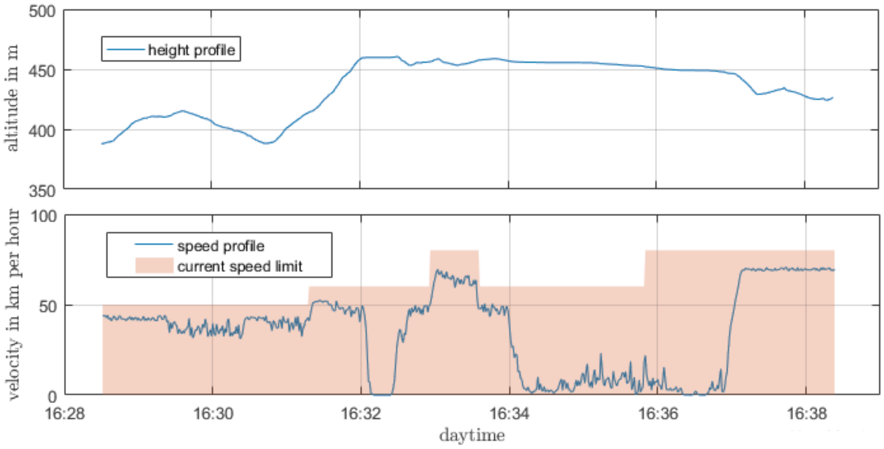


Figure 5: Elevation (top panel) and velocity profile (bottom) for an exemplary vehicle.

3 Test case analysis

This section evaluates all approaches elaborated in Section 2 in their ability to deduce reasonable velocity profiles. The test case analysis relies on an existing MATSim model for the Stuttgart region. Following an approach similar to [6], the MATSim model has been built (by RB) on the basis of a *mobiTopp* [33] travel demand model for the Stuttgart region provided by the *Verband Region Stuttgart* as part of a research collaboration. In total three different test cases were identified that differ in road type, network topology and right of way rules: (i) The *Bergheimer Steige* features no crossroads but sharp turns and road gradients up to 15 %. This test case seeks to analyze to what extent slope and curves influence vehicle speed in simulation. (ii) The *Motorway A8* (Kreuz Stuttgart to AS Stuttgart Möhringen) allows the analysis of traffic dynamics on motorways. (iii) The *Kräherwald* test case (leading from Kräherwald/junction Zeppelinstraße to University of Stuttgart) is of mixed inner-city and highway character and is part of the FKFS cycle as illustrated in Figure 6 (center and right panel).

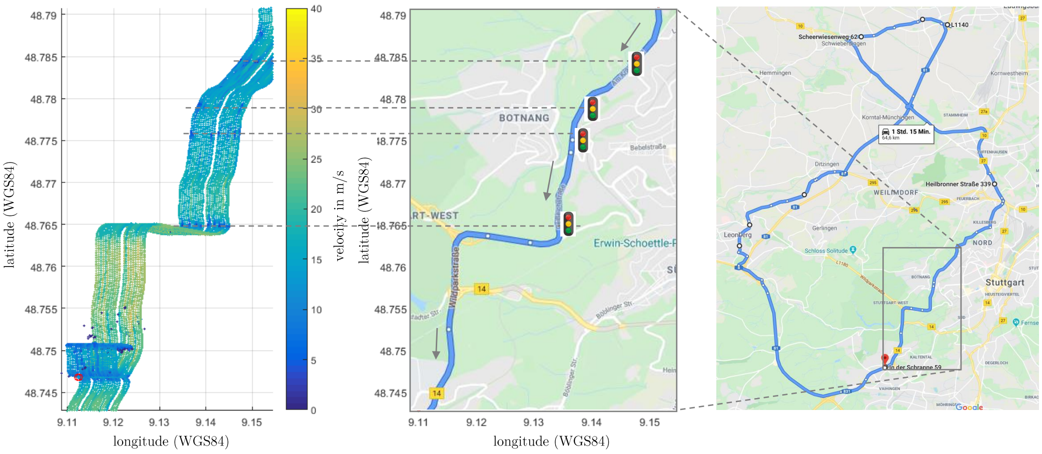


Figure 6: The right plot illustrates the FKFS circuit. Here, the grey rectangle borders the actual *Kräherwald* test case whose zoomed trajectory is given at center. To the left, the spatial velocity profiles of 22 measured FKFS cycles are provided for the *Kräherwald* test case.

All test cases are simulated in MATSim and SUMO for one day. For each test case, driving cycles are deduced by means of (a) pure MATSim simulation, (b) enhanced MATSim simulation with real driving cycles and (c) sequential MATSim-SUMO tool-coupling. The driving cycles are then compared based on **aggregated dynamics** such as velocity and acceleration distributions, overall traveled time and distance as well as average congestion ratio and energy consumption (Sec. 3.1). As further evaluation criteria serve the accuracy of **time- and space-dependent velocity profiles** (Sec. 3.1 and 3.2) as well as the simulation approaches' **large-scale feasibility** and **automation capability** (Sec. 3.3). Energy related KPI are derived from vehicle simulation in *GT-Suite*[15]. For this purpose, the following vehicle specifications have been used: vehicle mass (including battery and powertrain components) kg, constant tire rolling resistance $_{\mathrm{R}}$, vehicle front area $_{\mathrm{f}}$ m , vehicle air resistance coefficient $_{\mathrm{W}}$, road friction coefficient $_{\mathrm{fric}}$, battery capacity $_{\mathrm{bat}}$ kWh and engine power $_{\mathrm{eng}}$ kW. Driving dynamics and energy related KPI are additionally compared to 22 measured and GPS-tracked FKFS cycles for the *Kräherwald* test case[16]. To ensure comparability, the measured velocity profiles are equally passed to GT-Suite simulation.

[15] https://www.gtisoft.com/gt-suite/gt-suite-overview/

[16] All data were gathered by the [illegible] based on a contract research [illegible] with the [illegible]

3.1 KPI comparison

The following assessments refer to Table 1 which summarizes for each test case and simulation approach the most important aggregated KPI. For each test case, only a sample of simulated vehicles have been tracked microscopically. The exact numbers of tracked and simulated vehicles are indicated within the table as well.

Aggregated vehicle dynamics and energy related KPI In general, the average **traveled distance** of all tracked vehicles is similar in all simulation scenarios. Differences mainly arise due to different network designs and import functionalities. Every time a road attribute changes in OSM both MATSim and SUMO create a new link/edge. In contrast to SUMO, MATSim links are represented by straight lines only. In case this straight line deviates strongly from the actual road shape, MATSim inserts artificial nodes to preserve the network geometry. By consequence, one SUMO edge often represents several MATSim links which leads to longer SUMO distances especially in small test cases like ours. The calculated distances of the enriched scenario are purely artificial as they do not correspond to the actual target trajectories. Nevertheless, they are reasonable enough considered the little effort it took to implement the enrichment procedure. Solely the *Motorway A8* test case reveals discrepancies in traveled distance higher than the desired 5 % error margin. This however, is not the fault of the enhancement method itself. Those imperfections are caused by an insufficient number of available fast-driving 1 min-segments in Section 2.2 which also lead to low average velocities and energy consumptions. The validity of the enrichment procedure is therefore directly tied to a wide range of underlying measured driving cycles.

The **average travel time, velocity and energy consumption** are strongly congestion dependent. As the approaches base on different traffic dynamics (queue vs. car-following model) and network attributes (node vs. signaled intersection), the same ego-vehicle is differently delayed throughout the network which leads to different traffic conditions. Naturally, this affects average travel time, velocity and energy consumption. The inconsistencies in congestion modeling

Table 1: Aggregated KPI comparsion for all three test case
(no. of tracked/simulated ego-vehicles)

	KPI	**MATSim**	**enriched**	**SUMO**
Bergheimer Steige (359/7170)	average traveled distance in km	2.9	2.8	3.1
	average traveled time in min	4.3	4.8	4.6
	average congestion rate	0.85	0.72	0.84
	average velocity in km/h	43	36	42
	average energy consumption in kWh/100km	9.2	11.6	16
Motorway A8 (385/379591)	average traveled distance in km	6	5	6.8
	average traveled time in min	3.7	4.1	3.7
	average congestion rate	0.88	0.55	0.97
	average velocity in km/h	92	59	101
	average energy consumption in kWh/100km	23.8	15.8	24
Kräherwald (444/110779)	average traveled distance in km	5.5	5.4	5.7
	average traveled time in min	7.8	8.2	6.6
	average congestion rate	0.82	0.76	0.85
	average velocity in km/h	52	48	53
	average energy consumption in kWh/100km	11.8	12.4	15.6

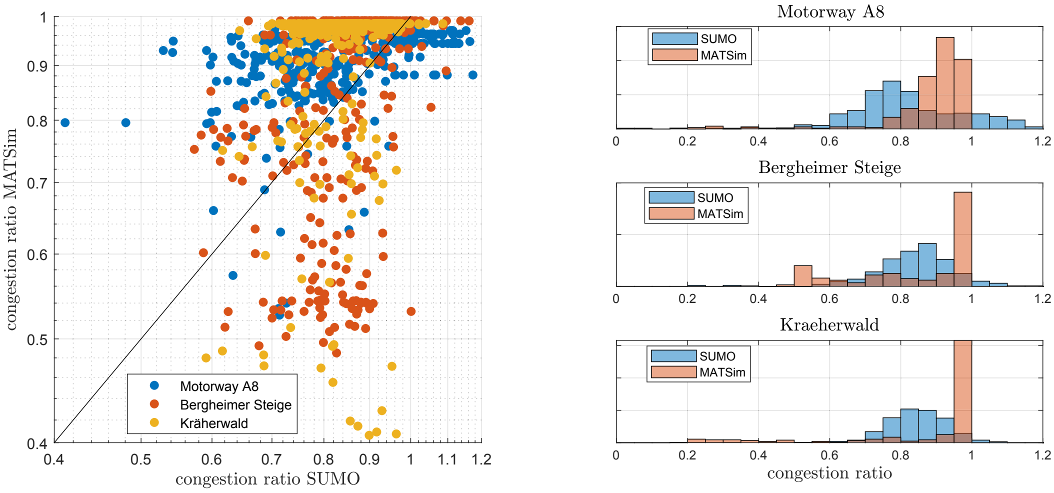


Figure 7: Inconsistencies in congestion modeling in MATSim and SUMO for all three test cases illustrated as log-log plot where each dot represents a tracked vehicle (left) and congestion ratio histograms (right). A congestion ratio of *one* corresponds to free-flow driving conditions.

are further illustrated in Figure 7 on the left, where the congestion ratio for both MATSim and SUMO are compared. At this, each dot represents a tracked vehicle. The congestion ratio is defined as the ratio of actual travel time and the free-flow travel time simulated in MATSim. A congestion ratio of *one* corresponds to free-flow driving conditions, whereas a ratio near *zero* signifies a blocked road[17]. A perfect match would theoretically result in a diagonal line. As depicted in Figure 7 this is seldom the case and needs to be investigated further. The histograms on the right show that the traffic conditions in MATSim are often too optimistic (presumably on links where the spatial queue did not propagate due to the artefacts discussed in Section 2.1) or way too pessimistic (presumably on short links).

When comparing the **speed and acceleration distributions** of all simulations, considerable differences in all approaches become apparent. Figure 8 displays the normalized velocity and acceleration histograms of all 359 tracked vehicles for the *Bergheimer Steige* test case. As expected, pure MATSim simulation exhibits unrealistic driving dynamics as it only accounts for average link speeds with no oscillations around the target velocity. Consequently the acceleration rate is predominately zero. In between two links however, the acceleration may jump from zero to an value predefined by the next link's speed limit. The enriched profiles feature more realistic driving dynamics, but as will be shown in Section 3.2, they are only as good as MATSim's capability to model spatial congestion patterns (which is limited at the moment). SUMO, in contrast, features more bell-shaped distributions (around local maxima) which, however, have not been validated yet. In the enriched MATSim and SUMO simulation the maximal acceleration is limited by design to m/s absolute. However, compared to real-world driving, the acceleration rates in SUMO are distributed too perfectly as equally stated in [1, 21, 37].

[17] As SUMO allows overspeeding (here: up to 20 %) and as SUMO link lengths do not match those of MATSim perfectly, congestion ratios greater *one* may result.

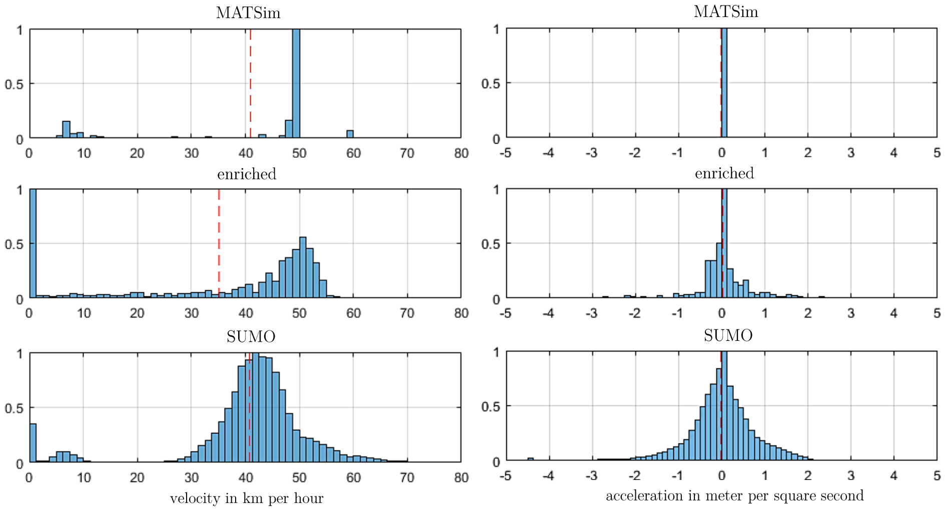


Figure 8: Normalized velocity (left) and acceleration distributions (right) for 359 tracked vehicles within the Bergheimer Steige test case with a temporal resolution of d s. The red dashed lines represent the mean values.

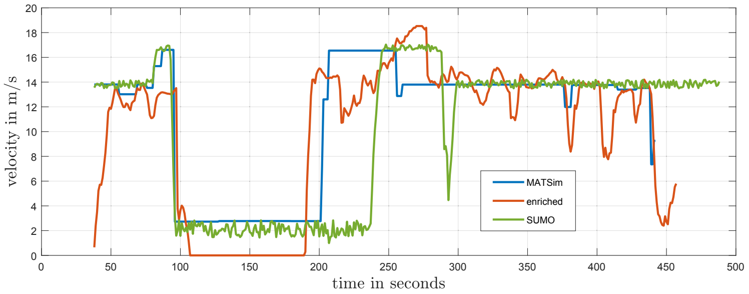


Figure 9: Time-dependent speed profile of a chosen vehicle of the Bergheimer Steige test case as simulated by the different simulation approaches.

Time-dependent speed profiles In Figure 9 the time-dependent velocity profiles for the same vehicle are shown. Even though the starting times are identical for all scenarios for the chosen vehicle[18], the car is differently delayed due to discrepancies in traffic conditions, network distances, traffic signals and right-of-way rules. Whereas MATSim's velocity profile is rather steplike due to the average link speed, SUMO shows strong oscillations around the target velocity (possibly arising from the driver imperfection). However, compared to real world driving, SUMO's oscillation amplitude seems too homogeneous and the frequency too high-frequent. This may be solved by a better parametrized car-following model, but as our approach aims at autonomous driving (AD) applications in future no further effort was put into this task.

[18] This might not always be the case. If strong congestion occurs on a vehicle's departure link, the moment of network entering can be delayed artificially.

3.2 Comparison against GPS-tracked FKFS cycles

In this section, the simulated driving cycles for the *Kräherwald* test case are compared against 22 measured FKFS cycles to assess the quality of the simulated results. To do so, only the part of the FKFS cycle is considered that overlaps with the *Kräherwald* test case as displayed in the right panel of Figure 6.

Space-dependent speed profiles All simulated (in SUMO only) and measured driving cycles of the *Kräherwald* test case are spatially compared in Figure 10 top panels. The bottom panels provide additional information on the vehicles' minimum, mean and maximum velocity at each location of the test case. As indicated in Figure 6, the trajectory undergoes first four successive traffic lights, becomes then a west-heading highway and is finally merged into another arterial road before turning abruptly south. Those characteristic become clearly visible in the both data sets in the form of sudden drops in velocity. In contrast to the FKFS data (that unfortunately reflect free-flow driving conditions only), the SUMO simulation on the right side exhibits some congestion during the day which leads to longer waiting queues in front of the traffic signals and especially when both arterial roads meet. Moreover, in real life locals tend to anticipate upcoming speed limit changes and adjust their velocity accordingly before the actual traffic sign occurs . This is especially true when the speed limit rises. In our simulation, however, the rise and fall of the speed limit is rather step-like. In the context of autonomous driving this simplification is not necessarily disadvantageous as future AVs might adapt to speed limits in a similar manner.

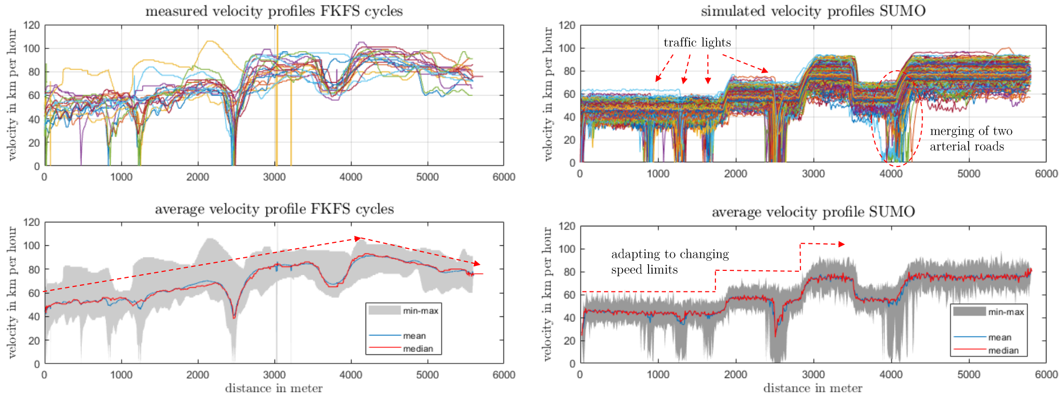


Figure 10: Stacked velocity-space profiles for 22 measured FKFS cycles (left) and 444 simulated SUMO vehicles (right) for the *Kräherwald* test case.

Aggregated vehicle dynamics For further plausibility checks, only those vehicles from the traffic simulation are benchmarked with FKFS data that exhibit similar traffic conditions. Unfortunately all measured cycles feature free-flow driving conditions, consequently no conclusions to the partly or fully congested state can be drawn. Table 2 summarizes selected **aggregated KPI** for a chosen, simulated vehicle and compares them with three different FKFS vehicles. Generally, all listed KPI match rather well for the non congested state regardless of the driving cycle deduction approach. A slightly different picture emerges when regarding the **velocity distribution** under free-flow driving conditions. As evidenced in Figure 11 on the

Table 2: FKFS benchmarking for a chosen simulated vehicle for the non-congested state.

KPI	**simulated cycles via**			**FKFS cycles**		
for the non-congested state	MATSim	enriched	SUMO	car 1	car 2	car 3
distance in m	5489	5460	5698	5951	5955	5942
average velocity in m/s	16.4	15.2	15.6	15.0	14.5	17.2
travel time in min	5.6	6.0	6.1	6.6	6.9	5.8
energy consumption in kWh/100km	20.1	18.3	18.8	18.8	22.0	19.6
congestion ratio	0.95	0.88	0.9	0.87	0.84	0.99

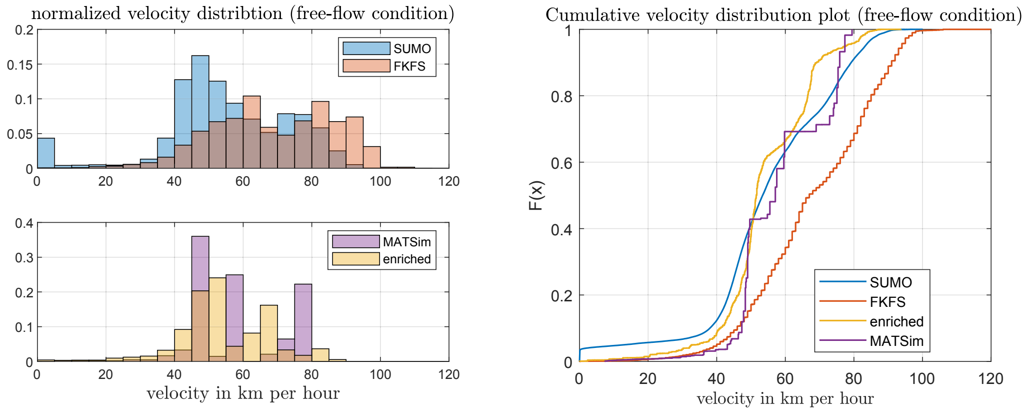


Figure 11: Probability density (left) and cumulative velocity distribution function (right).

left side, real world drivers (represented by the FKFS cycles) tend to drive faster than those simulated in SUMO. Whereas the SUMO simulation exhibits velocity peaks around 45 and 75 km/h, the measured data reach their local maxima around 62 and 82 km/h. Beyond that, the SUMO simulation features many velocities near zero which are not present in the measured data. Simulated vehicles obviously stand a higher chance to hit at least one of the four traffic lights. This also shows as an offset in the cumulative velocity distribution in the right panel of Figure 11: Whereas the graph gradients of SUMO and FKFS match rather well, SUMO's cumulative velocity distribution is shifted considerably more to lower velocities due to the traffic light downtimes. As further expected, MATSim's velocity distribution correlates poorly with the corresponding FKFS data due to the simplified queuing model. The enrichment technique compensates some of those shortcomings, but follows MATSim's trend still too closely. Using a larger sample of measured driving cycles for the enrichment, will likely lead to more realistic velocity distributions.

At this point, however, it has to be emphasized that the simulated driving cycles cannot be validated with the measured FKFS cycles for two reasons: (i) The 22 measured drive cycles are statistically not significant enough to represent the driving behavior of the *Kräherwald* test case during one day. (ii) To validate single profiles, the ego-vehicle's exact environment (e.g ambient traffic and traffic signals) needs to be modeled as encountered during measurement campaign. Unfortunately, neither MATSim nor SUMO are capable to model surrounding vehicles in such a manner. Furthermore, radar and LIDAR data are required to collect necessary data.

3.3 Discussion and implications for final concept choice

This section summarizes all quantitative results of the preceding sections, places them into the context of the key requirements postulated in Section 1 and complements them with qualitative remarks on the approaches' large-scale feasibility and automation capability.

With respect to city-wide SAEV fleet simulation (**KR1**), MATSim has advantage over SUMO in scalability and computational performance on the one hand and existing fleet simulation functionalities on the other hand. The enrichment and tool-coupling approach also benefit from MATSim's capabilities in this regard, as the modeled fleet constraints as well as the impact of different dispatching and routing algorithms do equally reflect in those solutions. Regarding individual vehicle states (**KR2**), MATSim and SUMO prove equally capable. Provided minute-wise drive cycle enhancement, the enrichment procedure should fare well in this regard as well since downtime phases are not altered considerably. Larger enrichment segments, however, increase the chance of annihilating idling periods or inserting additional ones.
The approaches' capability to derive reasonable velocity profiles (**KR3**) has extensively been analyzed in the previous section. Given similar traffic conditions, aggregated trip statistics (e.g. average velocity, traveled distance and time) are well captured by each approach. However, as highlighted in Subsection 3.1, even for a given ego-vehicle the traffic conditions differ considerably between the different approaches due differences in traffic dynamics and network interpretation. A central task in future work therefore relates to the model calibration in terms of (real-world-observed) congestion patterns. Unfortunately, MATSim (and therefore the enrichment approach as well) has some shortcomings in spatial congestion modeling. Another deficit of MATSim is its incapability to model realistic velocity and acceleration profiles due to its simplified queuing model. A satisfying solution that solely relies on MATSim without further enhancement is therefore not conceivable. The velocity profiles obtained from the enrichment procedure closely resemble real world measurements. However, it is not straight forward to transfer this approach to autonomous driving applications, since it depends on measurements as input data. A major drawback for the enrichment approach is therefore its missing sensibility to different driving

Table 3: Approaches' suitability to model detailed SAEV driving profiles
(suitable, o limited suitability, - not suitable, * no statement possible)

	key requirements	pure MATSim	enriched MATSim	pure SUMO	MATSim-SUMO tool-coupling
KR1:	large-scale, multi-modal SAEV fleet				
	simulation with sensitivity to:			-	
	- *range & charging constraints*			o	
	- *dispatching, routing & pricing strategies*			o	
KR2:	vehicle states				
KR3:	realistic velocity profiles with sensitivity to:				
	- *human/ autonomous driving*	-/-	/o	o/*	o/*
	- *congestion rates*	o	o		
	- *transport infrastructures*	o	-		
KR4:	further time series such as				
	- *height/ occupancy profiles*	o/	-/o	o/*	o/
KR5:	feasibility in terms of:				
	- *data availability*		o	o	o
	- *robustness against critical error*			-	-
	- *automation capability*			o	o
	- *computational resources*			o	

styles or platooning effects. SUMO in contrast, enables the deduction of detailed drive cycles whose drive dynamics prove too artificial to reflect for human driving, but may be reasonable enough for autonomous driving. In contrast to MATSim, SUMO provides many features to tweak driving dynamics in a methodological manner. Another strong point of SUMO is that the simulated vehicles react sensitive to diverse transport infrastructures and are able to mimic different driving maneuvers such as stop&go-patterns or zip merging. Unfortunately, SUMO does not account for reduced velocities in narrow curves. Nanoscopic traffic simulation tools such as *CarMaker*[19] would be required to address these kind of topics. The same applies for road gradients: road slope can technically be modeled in each simulation scenario **(KR4)** but requires access to accurate height data. These data, however, relate to the Earth's surface only and consequently produce invalid results for road tunnels. And even with slope modeled, the latter has so far no impact on the vehicles' driving behavior. Slope only influences energy consumption in a subsequent vehicle simulation. Nonetheless it has to be emphasized at this point, that numerous car-following models exist for SUMO. Some may address those issues already. At this point, those options have not been adequately tested nor investigated yet.

Apart from those quantitative KPI, all simulation approaches differ considerably in practical feasibility and automation capability **(KR5)**. With regard to the key requirements KR1-KR4, the MATSim-SUMO tool-coupling approach seems to be the most promising solution to deduce representative SAEV drive cycles as summarized in Table 3. However, its automation capability remains questionable due to the high effort in setting-up the network. SUMO networks are very detailed and therefore require additional data which OSM does not provide, e.g. detailed elevation information, traffic light positions and control. SUMO's autogenerated networks are sometimes misleading as the underlying OSM attributes are non-existing or error-prone and/or the data are too complex to be interpreted correctly by the default import functionalities. This is shown by (a) faulty turning lanes, (b) poorly guessed traffic light positions, (c) poorly joined complex junctions and (d) uncoordinated traffic light initialization. Manual editing represents a most time consuming task. A further serious drawback for all SUMO related approaches is their proneness to artificial deadlocks. Those gridlocks are created for example by two impeding cars, where the left likes to turn right and vice versa. Those gridlocks do not naturally resolve in SUMO, but can only be counteracted by enabling further options such as *time to teleport* or *ignoring junction blockers*. However, those options do not help if the ego-vehicle selected for drive cycle derivation is affected, as this vehicle then cannot complete its daily trajectory. MATSim in contrast, encounters no data-availability or automation problems due to its simplified network representation. Taken all pros and cons into consideration, the MATSim-SUMO tool-coupling seems most promising despite its automation challenges.

4 Conclusion

This contribution presents different approaches to simulate 24h driving cycles for SAEVs. The approaches are evaluated for a set of test cases. From this, a sequential tool-coupling of meso- and microscopic traffic simulation was found to be most promising with respect to the key requirements defined in Section 1. SAEV driving profiles are derived as follows: Depending on different fleet configurations and pricing concepts, SAEV fleets are implemented and simulated in MATSim on a large-scale, multi-modal network. Based on the simulation results, all SAEV trajectories are analyzed with respect to their daily use patterns, such as driven distance,

[9] https://ipg-automotive.com/de/produkte-services/simulation-software/carmaker/

operating time or number of served trips. Next, representative fleet vehicles are automatically identified and post-processed to be simulated in SUMO. To this end, the time-dependent travel demand of all roads in close proximity to the actual target trajectory is recorded in MATSim and transfered to the SUMO model. To reduce network setup effort, only the trajectories of the chosen vehicles (and their close neighborhood) are modeled in SUMO. Besides, each vehicle tagged as SAEV in MATSim simulation is featured with autonomous driving characteristics in SUMO. The ego-vehicle's speed profile is then derived from SUMO simulation.

At present, this tool-coupling approach works for test cases only as the procedure involves manual network matching and cleaning efforts. Its application to city-wide scenarios necessitates tool-chain automation which, however, constitutes a most challenging task. Further research is therefore required to implement the tool-chain in such a way that – starting from an existing, calibrated MATSim model – the SUMO model is setup, simulated and evaluated without further human intervention. To this end, the following aspects are addressed in future work:
(a) Dealing with inconsistencies in MATSim and SUMO. A sequential tool-coupling requires aligning both frameworks in (i) network representation, (ii) route choice, (iii) traffic dynamics on a macroscopic level and (iv) traffic performance. Otherwise, the travel demand transfer from MATSim to SUMO leads to severe gridlocks in the more congestion-prone microscopic traffic simulation and SAEVs cannot serve their appointed customers in time. Consequently, the frameworks' discrepancies need to be analyzed in more detail to derive alignment measures.
(b) Automated network modeling in SUMO. To solve the bottleneck of tool-chain automation, methods and algorithms need to be elaborated to solve network cleaning, traffic light location and control issues in an automated fashion. As time-dependent traffic volumes on all intersection are known from MATSim simulation, approaches are elaborated that (i) detect and eliminate artificial bottlenecks in the SUMO network that fail to handle the appointed traffic flow and (ii) mirror the decision makings of an actual traffic planner to initialize traffic lights.
(c) Automated travel demand transfer. Another obstacle for tool-chain automation represents the travel demand transfer from MATSim to SUMO simulation. This issue is solved by a robust network matching concept with dynamic meso-micro borders.

5 Acknowledgments

This work was supported by the Reasearch Programme on Automation and Connectivity in Road Transport of the German Federal Ministry of Transport and Digital Infrastructure (funding number 16AVF2147B).

References

[1] G. Amirjamshidi and M. J. Roorda. Development of simulated driving cycles for light, medium, and heavy duty trucks: Case of the toronto waterfront area. *Transportation Research Part D: Transport and Environment*, 34:255 – 266, 2015.

[2] A. Anastassov, D. Jang, and G. Giurgiu. Driving speed profiles for autonomous vehicles. In *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, jun 2017.

[3] K. W. Axhausen, A. Horni, and H. J. Herrmann. Final report: The risk for a gridlock and the macroscopic fundamental diagram. Technical report, 2015.

[4] J. Bischoff and M. Maciejewski. Simulation of city-wide replacement of private cars with autonomous taxis in berlin. *Procedia Computer Science*, 83:237–244, 12 2016.

[5] J. Bischoff, F. J. Marquez-Fernandez, G. Domingues-Olavarria, M. Maciejewski, and K. Nagel. Impacts of vehicle fleet electrification in Sweden - a simulation-based assessment of long-distance trips. Technical report, MT.ITS, 2019.

[6] L. Briem, N. Mallig, and P. Vortisch. Creating an integrated agent-based travel demand model by combining mobiTopp and MATSim. *Procedia Computer Science*, 151:776–781, Jan. 2019.

[7] R. Chen and M. W. Levin. Dynamic user equilibrium of mobility-on-demand system with linear programming rebalancing strategy. *Transportation Research Record: Journal of the Transportation Research Board*, 2673(1):447–459, jan 2019.

[8] T. D. Chen, K. M. Kockelman, and J. P. Hanna. Operations of a shared, autonomous, electric vehicle fleet: Implications of vehicle & charging infrastructure decisions. *Transportation Research Part A: Policy and Practice*, 94:243 – 254, 2016.

[9] S. Chugh, P. Kumar, M. Muralidharan, M. K. B, M. Sithananthan, A. Gupta, B. Basu, and R. K. Malhotra. Development of delhi driving cycle: A tool for realistic assessment of exhaust emissions from passenger cars in delhi. In *SAE Technical Paper Series*. SAE International, apr 2012.

[10] T. V. da Rocha, A. Can, C. Parzani, B. Jeanneret, R. Trigui, and L. Leclercq. Are vehicle trajectories simulated by dynamic traffic models relevant for estimating fuel consumption? *Transportation Research Part D: Transport and Environment*, 24:17–26, oct 2013.

[11] C. F. Daganzo. The cell transmission model: a dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B*, Vol. 28B(No. 4):pp. 269–287, 1994.

[12] K. P. Divakarla, A. Emadi, and S. N. Razavi. Journey mapping—a new approach for defining automotive drive cycles. *IEEE Transactions on Industry Applications*, 52(6):5121–5129, nov 2016.

[13] J. Erdmann. SUMO's lane-changing model. In *Modeling Mobility with Open Data*, pages 105–123. Springer International Publishing, 2015.

[14] D. J. Fagnant and K. M. Kockelman. Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in austin, texas. *Transportation*, 45(1):143–158, Jan 2018.

[15] A. Fotouhi and M. Montazeri-Gh. Tehran driving cycle development using the k-means clustering method. *Scientia Iranica*, 20(2):286 – 293, 2013.

[16] Fraunhofer Institute for Industrial Mathematics (ITWM). Virtual measurement campaign. product flyer, accessed: april 2020.

[17] C. Gawron. An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model. *International Journal of Modern Physics C*, Vol. 09(No. 03):pp. 393–407, 1998.

[18] Q. Gong, S. Midlam-Mohler, V. Marano, and G. Rizzoni. An iterative markov chain approach for generating vehicle driving cycles. *SAE International Journal of Engines*, 4(1):1035–1045, apr 2011.

[19] T. Holdstock and M. Bryant. Electric drivetrain architecture optimisation for autonomous vehicles based on representative cycles. In *17th Int. CTI Symp. Automotive Transmissions, Berlin*, 2018.

[20] A. Horni, K. Nagel, and K. W. Axhausen, editors. *The multi-agent transport simulation MATSim.* Ubiquity Press, London, 2016.

[21] Y. Hou, E. Wood, E. Burton, and J. Gonder. Suitability of synthetic driving profiles from traffic micro-simulation for real-world energy analysis: Preprint. *National Renewable Energy Laboratory (NREL)*, 10 2015.

[22] S. Hörl, F. Becker, T. J. P. Dubernet, and K. W. Axhausen. Induzierter Verkehr durch autonome Fahrzeuge. Eine Abschätzung. Technical report, SNF and ETH Zürich, 2019.

[23] I. Kaddoura, G. Leich, and K. Nagel. The impact of pricing and service area design on the modal shift towards demand responsive transit. Submitted to the 9th Int. Workshop on Agent-based Mobility, Traffic and Transportation Models (ABMTRANS), Warsaw, Poland, Apr. 2020.

[24] S. H. Kamble, T. V. Mathew, and G. Sharma. Development of real-world driving cycle: Case study of pune, india. *Transportation Research Part D: Transport and Environment*, 14(2):132–140, 2009.

[25] S. Krauss. Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics. Technical report, 1998. Dissertation. Mathematisch-Naturwissenschaftliche Fakultät, Universität Köln and German Aerospace Center (DLR). LIDO-Berichtsjahr 1999.

[26] S. Krauss, P. Wagner, and C. Gawron. Metastable states in a microscopic model of traffic flow. *Physical Review E*, 5:5597–5602, 1997. LIDO-Berichtsjahr 1997,.

[27] J. Liu, K. Kockelman, P. Bösch, and F. Ciari. Tracking a system of shared autonomous vehicles across the austin, texas network using agent-based simulation. *Transportation*, 08 2017.

[28] J. Liu, K. Kockelman, and A. Nichols. Anticipating the emissions impacts of smoother driving by connected and autonomous vehicles, using the moves model. 01 2017.

[29] B. Loeb, K. M. Kockelman, and J. Liu. Shared autonomous electric vehicle (SAEV) operations across the Austin, Texas network with charging infrastructure decisions. *Transportation Research Part C: Emerging Technologies*, 89:222 – 233, 2018.

[30] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. Microscopic traffic simulation using SUMO. In *The 21st IEEE IC on Intelligent Transportation Systems*, pages 2575–2582. IEEE, Nov. 2018.

[31] M. Maciejewski. Benchmarking minimum passenger waiting time in online taxi dispatching with exact offline optimization methods, 2014.

[32] M. Maciejewski, J. Bischoff, and K. Nagel. An assignment-based approach to efficient real-time city-scale taxi dispatching. *IEEE Intelligent Systems*, 31(1):68–77, jan 2016.

[33] N. Mallig, M. Kagerbauer, and P. Vortisch. mobiTopp – a modular agent-based travel demand modelling framework. *Procedia Computer Science*, 19:854 – 859, 2013. 4th IC on Ambient Systems, Networks and Technologies (ANT) , 3rd IC on Sustainable Energy Information Technology (SEIT).

[34] A. Moreno, A. Michalski, C. Llorca, and R. Moeckel. Shared autonomous vehicles effect on vehicle-km traveled and average trip duration. *Journal of Advanced Transportation*, 2018, 2018.

[35] P. Nyberg, E. Frisk, and L. Nielsen. Using real-world driving databases to generate driving cycles with equivalence properties. *IEEE Transactions on Vehicular Technology*, 65(6):4095–4105, 2016.

[36] S. Shi, N. Lin, Y. Zhang, J. Cheng, C. Huang, L. Liu, and B. Lu. Research on Markov property analysis of driving cycles and its application. *Transportation Research Part D: Transport and Environment*, 47:171–181, aug 2016.

[37] G. Song, L. Yu, and Y. Zhang. Applicability of traffic microsimulation models in vehicle emissions estimates. *Transportation Research Record: Journal of the Transportation Research Board*, 2270(1):132–141, jan 2012.

[38] I. H. Tchappi, V. C. Kamla, S. Galland, and J. C. Kamgang. Towards an multilevel agent-based model for traffic simulation. *Procedia Computer Science*, 109:887 – 892, 2017. 8th IC on Ambient Systems, Networks and Technologies (ANT), 7th IC on Sustainable Energy Information Technology (SEIT).

[39] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62:1805–1824, 02 2000.

[40] Z. Xiao, Z. Dui-Jia, and S. Jun-Min. A synthesis of methodologies and practices for developing driving cycles. *Energy Procedia*, 16:1868–1873, 2012.

[41] P. Yuhui, Z. Yuan, and Y. Huibao. Development of a representative driving cycle for urban buses based on the k-means cluster method. *Cluster Computing*, 22(S3):6871–6880, jan 2018.

[42] H. Zhang, C. J. R. Sheppard, T. E. Lipman, and S. J. Moura. Joint fleet sizing and charging system planning for autonomous electric vehicles. 2018.

[43] X. Zhao, Q. Yu, J. Ma, Y. Wu, M. Yu, and Y. Ye. Development of a representative EV urban driving cycle based on a k-means and SVM hybrid clustering algorithm. *Journal of Advanced Transportation*, 2018:1–18, nov 2018.

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.104
Published: 28 Jun. 2022

Action-points in human driving and in SUMO

Peter Wagner[1][https://orcid.org/0000-0001-9097-8026], Jakob Erdmann[1][https://orcid.org/0000-0002-4195-4535], and Ronald Nippold[1][https://orcid.org/0000-0003-4837-8021]

[1]Institute of Transportation Systems, DLR, Berlin, Germany

Abstract: When following a vehicle, drivers change their acceleration at so called action-points (AP), and keep it constant in between them. By investigating a large data-set of car-following data, the state- and time-distributions of the APs is analyzed. In the state-space spanned by speed-difference and distance to the lead vehicle, this distribution of APs is mostly proportional to the distribution of all data-points, with small deviations from this. Therefore, the APs are not concentrated around certain thresholds as is claimed by psycho-physical car-following models. Instead, small distances indicate a slightly higher probability of finding an AP than is the case for large distances. A SUMO simulation with SUMO's implementation of the Wiedemann model confirms this view: the AP's of the Wiedemann model follow a completely different distribution than the empirical ones.
Keywords: Car following, action points, driver modelling

Introduction

Car-following (CF) models are around since the early 1950ies [1], and they had been developed ever since. They had their first hype in the early 1960ies [2], [3] where especially models that use a description based on differential equations (ODE – ordinary differential equations, DDE – delayed differential equations, or even stochastic SDE – stochastic differential equations) have been used. Also during this time, the first action-point (AP) based models had been introduced [4], [5]. Already this early work assumed that the APs are related to perception thresholds of the human driver, with the idea that by crossing such a threshold, an appropriate action is triggered. E.g., very small speed differences to the vehicle in front are impossible to recognize by a human driver, once this speed difference crosses a critical value (which depends on distance , with larger distances making it more difficult to recognize a certain speed-difference). The AP-models have then been introduced in a much more refined form in the psycho-physical models [6], [7] that are being used in microscopic software packages (e.g. VISSIM). The reaction is then measurable by a fast change in the acceleration , where it is assumed that acceleration is constant when no AP is active.

When looking at car-following trajectory data such as those collected by Naturalistic Driving Studies (NDS) [8], [9], [10], [11], [12], [13], where data are typically resolved with a time-step size of , and from the fact that vehicles are heavy objects it is clear that acceleration cannot truly jump. For driver assistant systems, as well as in recommendations about comfort-able rides in public transit vehicles, typically there is a limit in the jerk of the vehicle. The jerk is the time-derivative of the acceleration, i.e. , and this is typically limited to values smaller than . This is also true for the data-set used in this work. However, when analyzing these data it is not too difficult to find times where acceleration changes fast, and times where it is constant. So, it seems that human drivers do have the habit of ”do nothing” as long as possible, and then change to what is needed to avoid a collision, or to avoid falling back to far behind.

The second important point is that the perception thresholds are most likely not precise curves in -space. They have to be understood in a probabilistic sense, where the probability to issue such an AP changes with the distance to this line, and the actual line e.g. marking the point where with certainty such an AP is issued.

Surprisingly, there is little work found by the author related to the details of these thresholds, and they are often ignored by most of the scientific literature on car-following. So, one may wonder, whether they are important at all. Not to be mis-understood: there is not the shadow of a doubt that human perception is limited, despite the fact, that the visual system of humans is capable of truly astonishingly feats. And therefore, perception threshold do exist. But it is quite another question whether they are needed to describe car-following behavior, and to what extend the behavior of human drivers is constrained by them. The driving style chosen by them might avoid them completely, and therefore be ignorant of these thresholds.

Let us note in passing, that the AP’s also occur in the other output of a human driver, that is the steering. To the knowledge of these authors, this is not a very well researched area (see [14] for an example).

The data-analysis

The data used here are from the German project simTD [10]. This project was not a NDS in the strict sense, its goal was to look into car-to-car communication. For this purpose, about 100 cars had been instrumented with communication devices, and these devices collected anything that could be collected from the CAN-bus of the vehicles, together with the proper geo-location and GPS-time. So, some of the vehicles recorded the distance and speed-difference to the lead vehicle, some of them recorded the gas-pedal (throttle) and brake-pedal usage, and all of them recorded the speeds , accelerations (lateral, as well as longitudinal), and GPS positions as well as the speeds from the GPS, together with a measure of the GPS error. The recording was asynchronous and in different time granularity, it ranged from for the GPS readings, most variables had , to even shorter intervals for the acceleration data ( ). With the exception of the GPS-data, these data have been enforced for the following analysis to a common time-step size of . If more than one value appeared in such a time-interval, only the average had been recorded together with a number telling how many values had been averaged.

The data had been collected in four months from September 2012 to December 2012 by about 1,000 volunteers who drove these vehicles around according to a certain protocol that was invented to maximize what can be learned from car-to-car communication. So, although the drivers where aware that they had been recorded, it was not done to look at their driving behavior.

To identify APs in the data set, several approaches had been tested (using the acceleration , the throttle position , or the speeds). It turns out, that the speed data of each vehicle yield the clearest signal to find the APs, a result that has also been reported in [15]. Identification of APs has been done, then, by applying the Ramer-Douglas-Peucker (RDP) [16], [17] and the Visvalingam-Whyatt (VW) [18] algorithm to the time-series . These algorithms have each one parameter that determines the degree of simplication to the time-series, and these have been chosen by try-and-error – there are no objective criteria that can be used here. (These algorithms have been invented originally to simplify geographical objects; there the goal is to eliminate detail but still keep the visual appearance e.g. of a coast line intact and recognizable.) The analysis has been done with the Python library simplification [19]. The rest of this analysis has been done with R [20], and the front-end RStudio [21].

The Figure 1 shows an example of the effect of the two simplification methods on these data. It could also been seen, that the VW algorithm seems to do a slightly better job, but this of course depends on the parameters chosen.

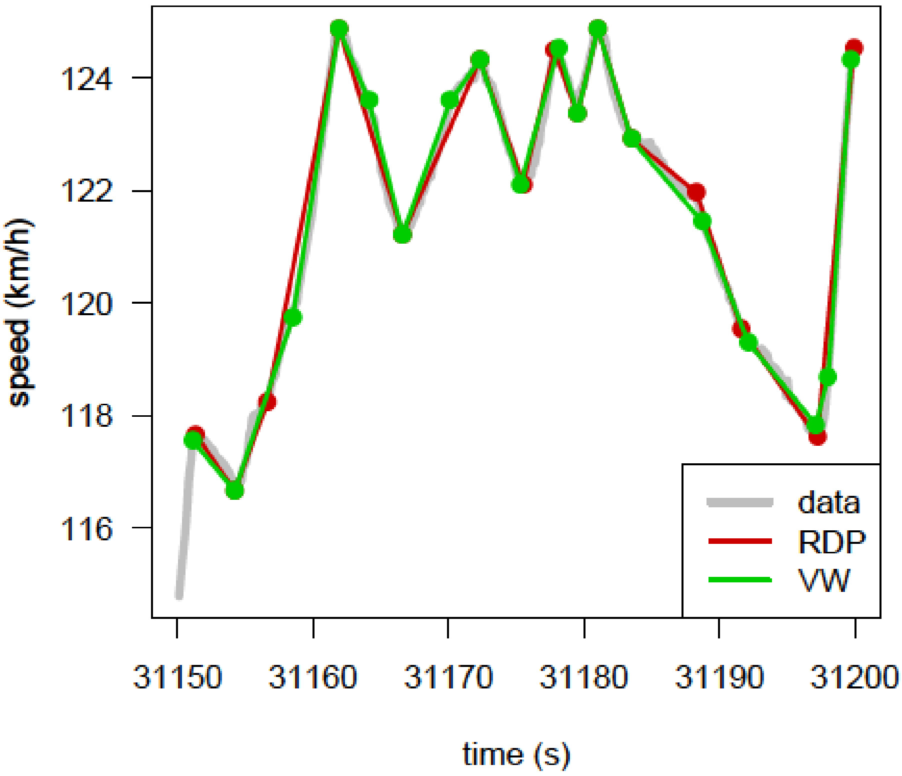


Figure 1. A short piece of the a speed time-series $v(t)$ (in gray), together with the results of the two simplification algorithms described in the text. RDP is in red, VW in green.

Note, that not all driving maneuvers allow for the proper assignment of APs. Especially periods of strong acceleration and deceleration often are more volatile, with acceleration changing faster than is visible in these data. However, car-following periods typically have small accelerations, and in these cases the method seems to work very well.

Distribution of the time intervals between APs

From such a division, the distribution of time-differences δt between subsequent APs can be computed as well, see Figure 2 for the result. Note, that due to holes in the data, some APs might have been missed. The distribution is compatible with a log-normal distribution. The two maxima are around a value of $1.5\,\mathrm{s}$, while the mean values are about $3\,\mathrm{s}$.

Car-following

So far, all data have been used for the analysis to state some basic facts about the APs. Now, a cloaser look at car-following episodes are performed. Car-following is identified as follows. First of all, there should be a data-point that has both distance and speed-difference to the

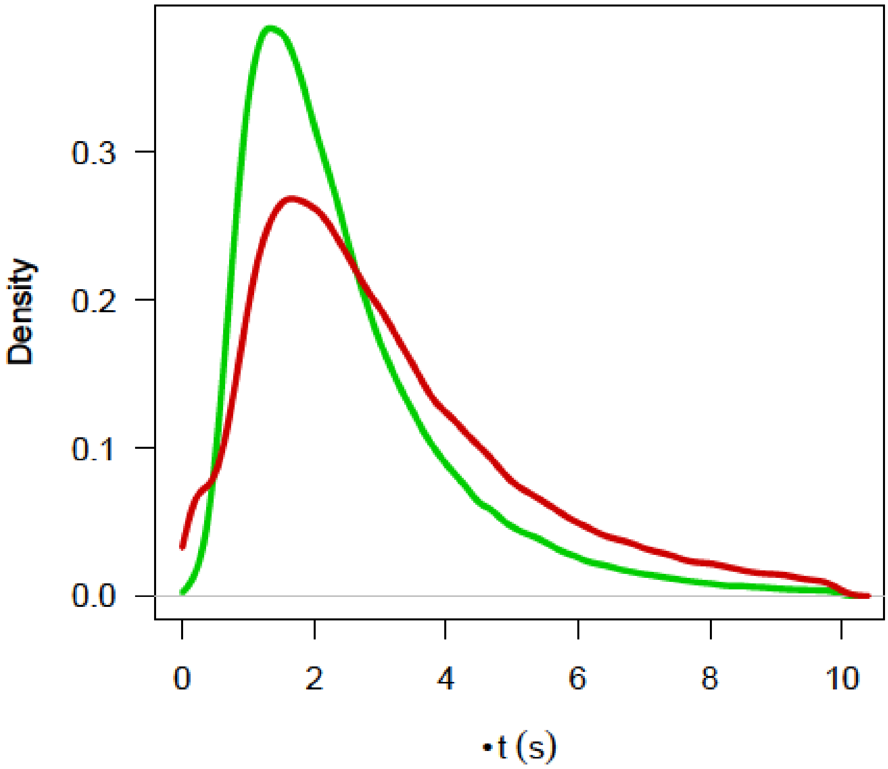


Figure 2. Probability density $p(\delta t)$ of the time-difference δt between subsequent APs. Red line is for the RDP, green line for the VW algorithm. Data are cut-off at ten seconds.

vehicle in front. To give an estimate, this is the case for about 25% of the data. Additional tests have been applied to clean the data. The most important of them is to check for a dynamic consistency. Note $g(t)$ as the gap at time t, and $g(t+\Delta t)$ the gap at a (small) time-step Δt later. The speed-difference is named $\Delta v(t) = V(t) - v(t)$, where $V(t)$ is the speed of the leading vehicle. Then:

$$g(t+\Delta t) = g(t) + \Delta t \Delta v(t) + \mathcal{O}((\Delta t)^2) \tag{1}$$

The size of the $\mathcal{O}$-term is the difference between the acceleration of the lead and the following car, multiplied by the square of the time-step size. By assuming "normal" accelerations of $2.5\,\mathrm{m/s^2}$ and a time-step size $\Delta t = 0.1\,\mathrm{s}$, the expected error here is of the order of $0.05\,\mathrm{m}$. So, the gap error ε_g:

$$\varepsilon_g = g(t+\Delta t) - (g(t) + \Delta t \Delta v(t)) \tag{2}$$

can be used to filter out bad data-points. The same can be done with the speed and acceleration by defining a speed error ε_v:

$$\varepsilon_v = v(t+\Delta t) - (v(t) + \Delta t a(t)) \tag{3}$$

By setting a limit of 1 m for the gap-error and 0.25 m/s for the speed-error (the interquartile distance for the gap-error is of the order 0.2 m, and 0.05 m/s for the speed-error), about 25% of the data gets eliminated.

The remaining data are now being used to search for a difference between the distribution of the APs in the CF-plane and the distribution of all the data. This is done by sampling the data into two histograms and for the APs and all the remaining data. There are several means how two distributions can be compared. Here, a method is chosen that bears a strong resemblance with the well-known -test. It works as follows: let be the histogram sampled from the APs by a particular tesselation of the CF-plane, and the histogram for all the data-points, using the same tesselation of the CF-plane. Then, each bin in the AP-histogram is related to the same bin in the full histogram. The simplest assumption one may have is that the AP-histogram is just a down-scaled version of the full histogram, i.e. it is expected, that the AP histogram can be computed by the multiplication of the full histogram with an empirically defined factor, which for the data here turned out to be around :

$$\tag{4}$$

Then, the Pearson residuum can be defined:

$$\frac{\quad}{\sqrt{\quad}} \tag{5}$$

Clearly, the sum over is just the value. However, in the context here, more can be learned than the simple fact that these two distributions are different: by plotting as function of , the Figure 3 is found.

The result in Figure 3 displays no one-dimensional lines where the AP-distribution would have been larger. In general, the difference between the two distributions is weak, but it displays a clear pattern. For large distances, the AP density is smaller than what can be expected on the basis of Equation (5), while for smaller distances, drivers issue more APs. There is a slight asymmetry between positive and negative , which is as expected: negative values of belong to the dangerous area where the vehicle is approaching. However, the difference itself is weak, values of correspond to a 5% error probability.

Note the unequal tiling of the plane: this has been chosen so that roughly the same number of data-points fall into each box in as well as in -direction. It improves the statistics, at the expense of the accuracy in the location of the boxes. Small boxes corresponds to a large probability to find the system there, and in fact, that maximum of the -distribution is around . About 1.4M data-point had been used to compute this diagram, about 5% of those have been labelled as APs.

Running SUMO with Wiedemann's model

To use SUMO [22] to generate similar data, the following set-up has been used. A large rectangle of a one-lane road has been built with netedit, with changing speed limits to sample from different car-following regimes. Altogether six vehicles had been put to this network, at the start of the simulation they were at a standstill. The first (lead) vehicle was a SUMO default vehicle that drove with constant speed. Its speed-factor had been set to 0.5, so that is drives with half of the speed-limits on the four edges, i.e. at , and . The five following cars were configured as

```
<vType id="followVIS" length="4.61" maxSpeed="70.0" minGap = "1.0">
<carFollowing-Wiedemann accel="1.8" decel="4.5" sigma="0.9"/>
```

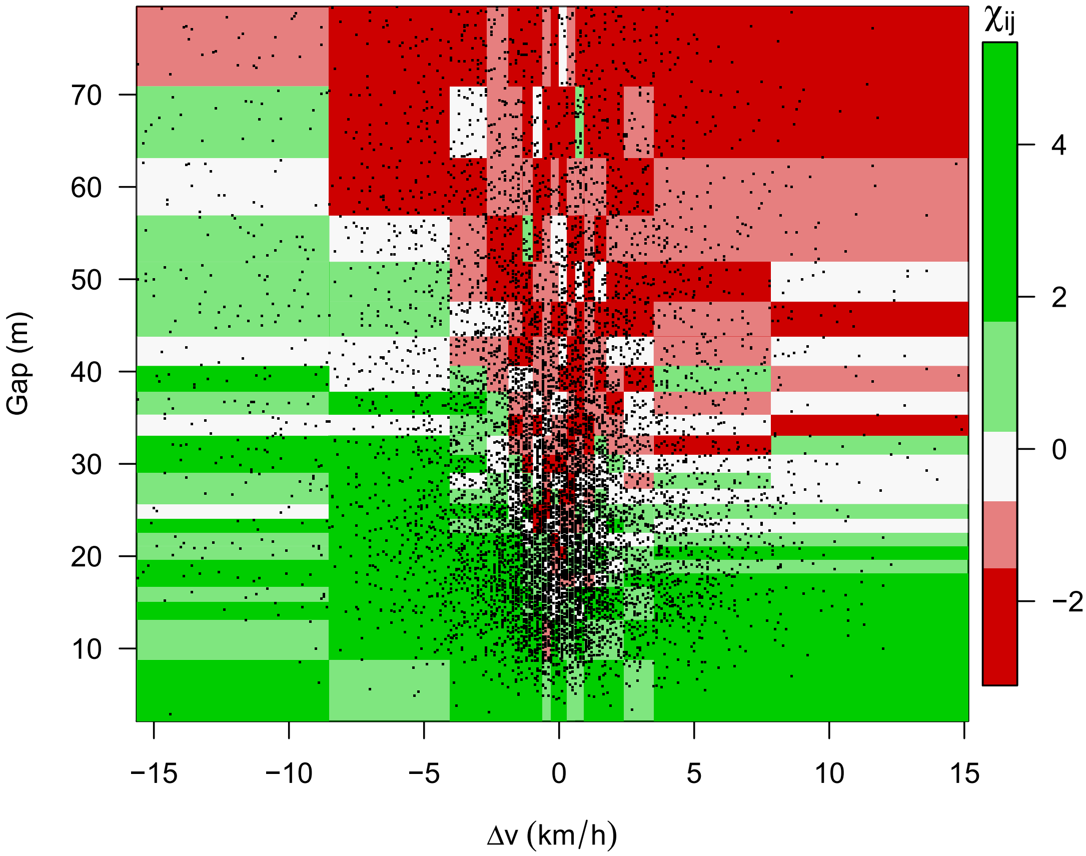


Figure 3. Distribution of the difference between the AP distribution and the distribution of the full frequency distribution $p(\Delta v, g)$. Note, that this plot is roughly divided into two areas: for large distances, less APs are issued, while for short distances, more APs are needed.

```
</vType>
```

SUMO's own deterministic AP mechanism `default.action-step-length value="0.1"` had been set to the step-size of 0.1 s, to be close to the empirical data. The data had been sampled from the simulation by using a netstate dump `netstate-dump value="wiedemannAP.xml"` and subsequently analyzed with R [20]. AP's have been found by searching for points where the acceleration of the vehicle has changed by more than $0.1\,{}^{\mathrm{m}}/_{\mathrm{s}^2}$, the acceleration itself has been computed from the (recorded) speed by a simple difference scheme $a(t) = (v(t) - v(t - \Delta t))/\Delta t$. All the data from the five following cars had been used and analyzed together, as had been done with the simTD data. This yields the results in Figure 4.

The results in Figure 4 demonstrate that there is a strong difference between the APs generated by the Wiedemann model, and the APs identified in the empirical analysis. Nevertheless, these results are completely in line with what to expect from Wiedemann's model of the AP distribution: they are lined up in a scattered manner along the perception thresholds as defined in the Wiedemann model.

Conclusions

These results indicate, that human car-following is not controlled in any manner by perception thresholds. Similar results have also been found in [23] and [24]. The statistical analysis above demonstrates, that APs in fact have a non-trivial distribution: they are issued more often when the situation is dangerous, but the effect is not a very strong one.

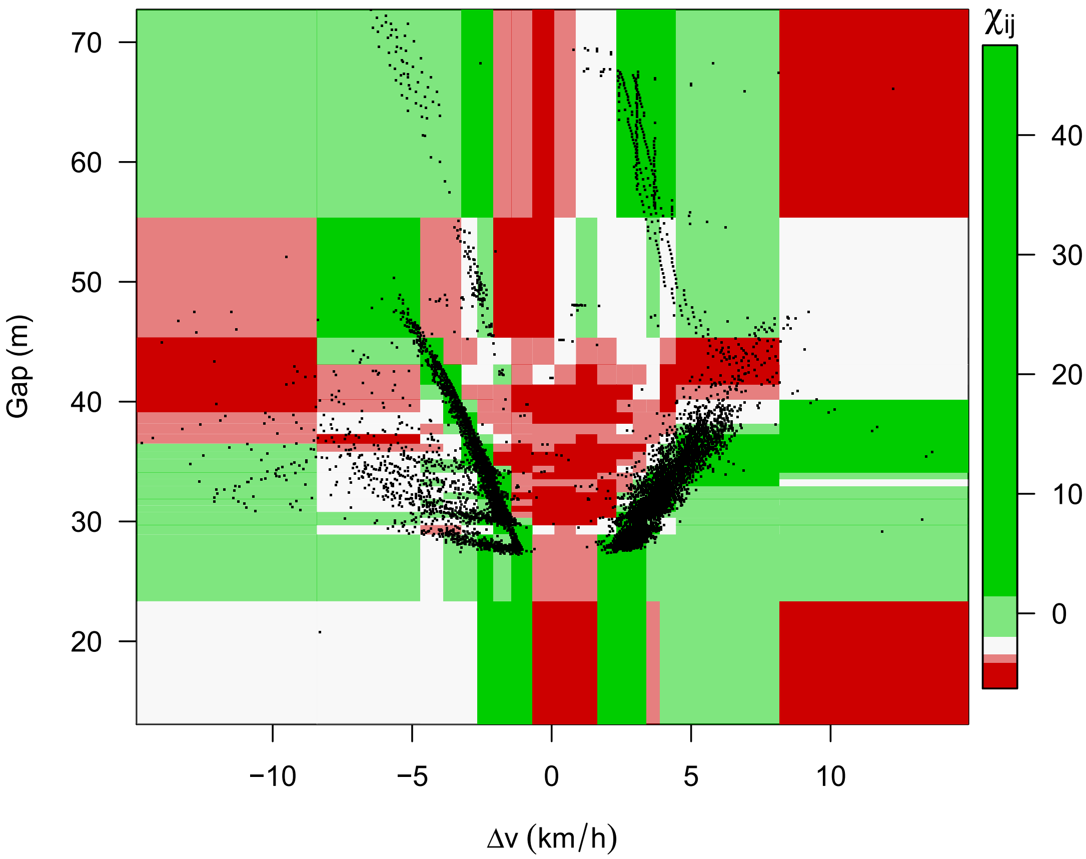


Figure 4. Distribution of the difference between the AP distribution and the distribution of the full frequency distribution $p(\Delta v, g)$ for the Wiedemann model, together with the AP's themselves.

There is still a (small) loop-hole in this analysis for the existence of perception thresholds: to gain statistical power, all the data for all the drivers have been put together, in this case these had been 96 drivers. The data from just one driver are not enough to find those patterns, since they do not happen that often, they are roughly 5% of the total amount of car-following data. So, an experimental design that would look explicitely for those thresholds would try to collect long car-following data from just one driver. However, we think it is highly unlikely for such an approach to succeed, since the distribution of APs found in the simTD data is so completely different from the one from the simulation with Wiedemann's model.

If this result holds true, it might be asked why do the thresholds do not play a prominent role. One of the answers comes from the average time between two APs of 1. . . 3 seconds. Drivers correct their driving style much more often than what would be needed by the thresholds. Therefore, they do not take care of the thresholds. In addition, especially when in car-following, drivers are typically relaxed, at least this can be concluded when looking at the accelerations that are realized in this mode. And this means, that they are also not fully concentrated, and it might be assumed that their perception error is larger, too. And again, this would lead to a smoothing out of any threshold.

Let us finally note that this does not mean that the Wiedemann model is a bad model. On average, the acceleration function $a(\Delta v, g, v)$ of this model does still the correct things. It indicates, however, that this model has a feature (the perception thresholds) that cannot be found in the simTD data. However, some papers have seen these thresholds (at least we have found this reference [25], but we remember to have seen others) and the associated increase in APs issued at these thresholds. It can only be speculated what has been seen in these data, and as mentioned already, it might well be that with a more careful preparation of the car-following experiments those thresholds can be seen.

Data Availability Statement

The data used stem from the simTD project [10] and cannot be shared.

Author contributions

PW did most of the simulation and data analysis. JE and RN provided technical assistance with the simulation, and helped with the writing of the text. All authors have read and approved the text.

Competing interests

The authors declare no competing interests.

Funding

The research was funded by DLR's basic research funds.

References

[1] A. Reuschel, "Fahrzeugbewegung in der Kolonne bei gleichförmig beschleunigtem oder verzögertem Leitfahrzeug," *Zeitschrift des österreichischen Ingenieur und Architektenvereins*, p. 95, Jul. 1950, In German.

[2] R. Herman, E. Montroll, R. Potts, and R. Rothery, "Traffic dynamics: Analysis of stability in car following," *Operations Research*, vol. 7, no. 1, pp. 86–106, 1959. DOI: 10.1287/opre.7.1.86. eprint: https://doi.org/10.1287/opre.7.1.86. [Online]. Available: https://doi.org/10.1287/opre.7.1.86.

[3] D. C. Gazis, R. Herman, and R. W. Rothery, "Nonlinear follow-the-leader models of traffic flow," *Operations Research*, vol. 9, no. 4, pp. 545–567, 1961. DOI: 10.1287/opre.9.4.545. eprint: https://doi.org/10.1287/opre.9.4.545. [Online]. Available: https://doi.org/10.1287/opre.9.4.545.

[4] E. P. Todosiev and L. C. Barbosa, "A proposed model for the driver-vehicle-system," *Traffic Engineering*, vol. 34, pp. 17–20, 1963/64.

[5] E. P. Todosiev, "The actionpoint model of driver vehicle system," Ph.D. dissertation, The Ohio State University, Columbus, Ohio, USA, 1963.

[6] R. Wiedemann, "Simulation des Straßenverkehrsflußes," Institut für Verkehrswesen, Universität Karlsruhe, Tech. Rep., 1974, Heft 8 der Schriftenreihe des IfV, in German.

[7] H.-T. Fritzsche, "A model for traffic simulation," *Transportation Engineering And Control*, no. 5, pp. 317–321, 1994.

[8] P. Fancher, R. Ervin, J. Sayer, *et al.*, *Intelligent cruise control field operational test*, HS 808 849, University of Michigan Transportation Research Institute, U.S. DOT, 1998.

[9] FHWA, *Next Generation Simulation Program*, http://ngsim.camsys.com/, accessed Sept. 2006, 2006.

[10] *Safe and intelligent mobility – test field germany*, URL: http://www.simtd.de/index.dhtml/enEN/index.html, last access 7.7.2015, 2012. [Online]. Available: http://www.simtd.de/index.dhtml/enEN/index.html.

[11] *SHRP2 naturalistic driving study*, URL: https://insight.shrp2nds.us/, last access 31.7.2019, 2019. [Online]. Available: https://insight.shrp2nds.us/.

[12] *EuroFOT*, URL: http://wiki.fot-net.eu/index.php?title=EuroFOT, last access 31.7.2019, 2012. [Online]. Available: http://wiki.fot-net.eu/index.php?title=EuroFOT.

[13] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *2018 IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[14] V. L. Knoop and S. P. Hoogendoorn, "Relation between longitudinal and lateral action points," in *Traffic and Granular Flow '13*, M. Chraibi, M. Boltes, A. Schadschneider, and A. Seyfried, Eds., Cham: Springer International Publishing, 2015, pp. 571–576, ISBN: 978-3-319-10629-8.

[15] S. Hoogendoorn, R. G. Hoogendoorn, and W. Daamen, "Wiedemann revisited: New trajectory filtering technique and its implications for car-following modeling," *Transportation Research Record*, vol. 2260, no. 1, pp. 152–162, 2011. DOI: 10.3141/2260-17. eprint: https://doi.org/10.3141/2260-17. [Online]. Available: https://doi.org/10.3141/2260-17.

[16] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer Graphics and Image Processing*, vol. 1, no. 3, pp. 244–256, 1972. DOI: 10.1016/S0146-664X(72)80017-0.

[17] D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *The Canadian Cartographer*, vol. 10, no. 2, pp. 112–122, 1973. DOI: 10.3138/FM57-6770-U75U-7727.

[18] M. Visvalingam and J. D. Whyatt, "Line generalisation by repeated elimination of the smallest area," CISRG discussion paper ; 10, Cartographic Information Systems Research Group, University of Hull, Tech. Rep., 1992. [Online]. Available: https://hydra.hull.ac.uk/resources/hull:8338 (visited on 07/31/2018).

[19] S. Hügel, *simplification: Fast linestring simplification using RDP or visvalingam-whyatt and a Rust binary*, [Online; accessed 31 Aug 2018], 2016. [Online]. Available: https://pypi.org/project/simplification/.

[20] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2018. [Online]. Available: https://www.R-project.org/.

[21] RStudio Team, *Rstudio: Integrated development environment for r*, RStudio, Inc., Boston, MA, 2015. [Online]. Available: http://www.rstudio.com/.

[22] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*, IEEE, 2018. [Online]. Available: %3Chttps://elib.dlr.de/124092/.

[23] P. Wagner, "Empirical description of car-following," in *Traffic and Granular Flow '03*, S. P. Hoogendoorn, S. Luding, and P. H. L. Bovy, Eds., Springer, 2005, pp. 15–28.

[24] R. G. Hoogendoorn, S. P. Hoogendoorn, K. A. Brookhuis, and W. Daamen, "Longitudinal driving behavior under adverse conditions: A close look at psycho-spacing models," *Procedia - Social and Behavioral Sciences*, vol. 20, pp. 536–546, 2011, The State of the Art in the European Quantitative Oriented Transportation and Logistics Research – 14th Euro Working Group on Transportation & 26th Mini Euro Conference & 1st European Scientific Conference on Air Transport, ISSN: 1877-0428. DOI: https://doi.org/10.1016/j.sbspro.2011.08.060. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877042811014406.

[25] R. Wiedemann and U. Reiter, “Microscopic traffic simulation: The simulation system mission, background and actual state,” Project ICARUS (V1052) Final Report. Brussels, CEC, 2., Tech. Rep., 1992.

SUMO User Conference 2020
https://doi.org/10.52825/scp.v1i.94

Published: 28 Jun. 2022

ECN-based Mitigation of Congestion in Urban Traffic Networks [*]

Levente Alekszejenkó[1] and Tadeusz Dobrowiecki[1]

Budapest University of Technology and Economics, Budapest, Hungary
ale.levente@gmail.com

Abstract

Traffic congestions cause many environmental, economic and health issues. If we are unable to completely get rid of them, the least we shall try to do is to move them outside of residential areas.

In this paper, a novel signal coordination method is proposed, which aims to mitigate traffic congestions. The proposed algorithm is based on the explicit congestion notification protocol, which is well-known from the domain of computer networking.

Our method was tested under Eclipse SUMO. Results show that the proposed algorithm successfully limits the traffic density and the traffic flow to a certain level.

1 Congestion Problem – A Local Perspective

Nowadays, as the number and the size of the road vehicles are rapidly increasing, even more frequent and longer traffic congestions are formed. Therefore, we might encounter heavy traffic in areas where they can cause even more harm than on main roads or highways.

For example in residential areas or near hospitals vast amount of pollution, noise and vibration, coming from the vehicles, can cause health issues. In areas near nursery and elementary schools they can also pose a safety risk. Moreover, the modern city planning is about to ban vehicles from historical city centers as well. The aforementioned areas usually build up of small, narrow, sometimes even dangerously steep roads with many *right-hand-rule* intersections.

These examples show that it would be really beneficial to avoid heavy traffic to reach specific parts of our cities. Of course, this means there might be areas where the congestions will be even bigger than today, but it might be possible to handle the increased traffic more efficiently there (by e.g., variable speed limits, bi-directional lanes on highways and so on), than in the regions mentioned above.

In this paper, we propose a novel traffic signal coordination method which is capable of restraining heavy traffic from reaching a certain area, which ensures that the density of the vehicles is kept below a critical level, therefore resulting in mitigation of congestion on residential road networks.

2 ECN Protocol and its Adaptation to Urban Traffic Networks

Congestion does not uniquely appear in the domain of vehicle traffic, it is also present in computer networks as well. There are numerous solutions which aim to prevent or handle

[*] The research reported in this paper was supported by the BME NC TKP2020 grant of NKFIH Hungary, as well as by the BME- Artificial Intelligence FIKP grant of EMMI (BME FIKP-MI/SC).

the congestion in computer networks. Our idea is that some of these algorithms might be applicable to achieve our goal, mentioned above. This conjecture is based on the insight that these protocols prevent packages from being send with too high transmission rate, therefore the network can handle the incoming messages with ease.

Moreover, many other elements are similar in computer networking and the road traffic domain. The networks themselves, for example, are built up of nodes (routers vs. intersections), which might be even capable of actively managing the flow of packets or vehicles in the network. The message packets and the vehicles also form a quantum stream, which contains unique entities with a space between them. The major difference is that vehicles have physical dimensions too, therefore some protocols of computer networks cannot be applied in the domain of road networks.[1]

One of the well-known congestion avoidance algorithms in computer networking is called *Exponential Backoff.* The basic idea behind the exponential backoff is that in case of a collision (which roughly means that a congestion is forming in the network) the transmitters have to wait for a random time between 0 and 2^c μs to resend their messages, where c stands for the number of unsuccessful transmissions. Obviously, this protocol cannot be applied in the domain of road traffic.[2]

Another example of congestion reduction in computer networks is called *Sliding Window Protocol.* This protocol limits the number of packets that can be transmitted at any given time, and as a result it prevents forming congestions. Unfortunately, this protocol might also cut platoons into half, however, platooning is proved to be really beneficial in the traffic domain. Hence, this method is unsuitable for our purposes.

The algorithm which can be easily applied in road traffic and computer network domains as well, is called the *Explicit Congestion Notification* (ECN) protocol [1]. The main idea behind it is that the routers (or in our case the traffic light controllers) can sense somehow the formation of a congestion. If it happens, the routers inform the corresponding transmitters about this fact (let us call these piece of information an *ECN-signal*). If an ECN-signal has been received, the transmission level shall be reduced toward the sender of the ECN-signal.

3 Percepting and Mitigating Congestion

3.1 Overview of ECN-based Traffic Signal Coordination

Intelligent traffic light controllers can be modeled as intelligent agents, which can communicate with each other. Let us call them *judges*, for convenience, and the ones running the ECN-protocol then will be called *ECN-judges.* These communicating judges form a layer in a multi-agent intelligent system (another layers are the communication between vehicle agents and the communication between vehicle platoons and the intersection judges). In our research, we suppose that the communication is free from errors, and the agents themselves are cooperative, trustworthy and bona-fide.

We assume that an ECN-judge can sense somehow the formation of a congestion, and can inform its topological neighbors upstream about this fact by sending out an ECN-signal. If an ECN-signal arrives, the ECN-judge can alter its program accordingly, hence reduces its throughput towards the forming congestion. In any other case, judge should control the traffic

[1] I.e., these protocols use some operations, e.g., dropping or reorder packets, which take advantage of the non-material existence of data.

[2] Most of the people would be really angry, if they were unable to go out from their garages for a long time, because some nearby intersections cannot receive any more vehicles at the moment.

like an actuated traffic light controller, since it has been proved that actuated traffic lights are one of the best ways [2] to optimize the flow of the traffic.

The tricky part is, that the state-space of a single intersection is enormous, regarding the incoming traffic demand, the received ECN-signals and some traditional expectations (e.g. to be fair). Therefore, it seems to be impossible to store an appropriate TLS-program for every situation, see Appendix A. Consequently, signal plans should be generated in real-time. In the following sections, the components of the proposed system are described in detail.

3.2 Sensing the Formation of a Congestion

Detecting the formation of a traffic congestion is a really challenging task. Our research did not focus on this particular problem, therefore we used here a simple solution.

By analyzing the data, which is supported by the loop detectors, the maximum traffic flow can be found on a specific edge. This traffic flow value corresponds to a particular level of occupancy of the given edge.

Above this occupancy level, we suppose the traffic flow will decrease, meaning that a congestion is forming. Therefore, we shall avoid reaching this point, by limiting the occupancy level to 90% of what corresponds to the maximum flow.

In this way, the occupancy limit is set for all edges (traffic lanes) entering into an intersection that is controlled by an ECN-judge. Every now and then (i.e. ca. every 15 s) the ECN-judge calculates the occupancy levels (as the moving average in consecutive time windows) along these edges, and if somewhere the set limit is reached, an ECN-signal with congestion notification is sent out. If the occupancy level falls below the limit, the ECN-signal informs the other judges that the congestion along this particular edge has been dissolved.

3.3 Generating a Signal Plan

ECN judges have to generate signal plans online. These signal plans are based on a simple round robin scheduling, resulting in a fair schedule for all directions. The phase times are adaptively set, and the plans are also influenced by the congestion state of the neighboring intersections, coupled to the intersection which is governed by an ECN-judge.

Computation of a simple signal phase when generating a traffic light system program (TLS-program) can be formalized as an *integer programming problem (IP)*. Its goal is to maximize the number of directions which may receive a green light. The matrix of constraints defining the problem is composed from the so-called conflict matrix of the given intersection, describing which directions cannot receive a green light simultaneously, due to the risk of accidents. The other part of the constraint matrix are the so-called additional constraints. Here the logic of the scheduling can be defined as well as the desirable reaction to the incoming ECN messages (i.e., describing which direction need or may not receive a green light at the moment). These components shall be set in accordance to the actual traffic and ECN-notifications.

By solving this IP problem, a signal plan can be obtained.

3.4 Periodic Recalculation of Signal Plans

In order to ensure the periodic working of an ECN-judge, signal plans shall be recalculated every now and then. Let us call the time between two recalculations as phase time (T). This time naturally depends on the number of the vehicles which currently receive a green light (N_v).

Using this parameter the phase time is calculated using equation (1).

$$T = \begin{matrix} N_v \cdot 1,5\ s + 5\ s, & \text{if } N_v \leq 23 \\ 40\ s & \text{otherwise} \end{matrix} \tag{1}$$

4 Extending Eclipse SUMO

4.1 Previously Developed MAS System

In our previous works [3, 4], a cooperative multi-agent system has already been implemented by extending Eclipse SUMO [5]. This system consists of connected autonomous vehicles, the so-called *smart cars* and intelligent traffic light controllers (*judges*), which, nevertheless, were not connected to each other. The smart cars and the judges were able to communicate with each other. When the smart cars approached an intersection, they requested permission from the corresponding judge to pass through. The judges used simple scheduling algorithm to find out when this permission shall be granted.

In that earlier system, smart cars, which are following closely each other and have exactly the same trajectories, can form groups, so-called *platoons*, before entering an intersection. Such platooning method can somewhat improve the traffic flow by reducing the impact of changing lanes. Another benefit of this method is that only the leader of a platoon needs to exchange messages with the judges, as every other member of a platoon has to follow the vehicle ahead of it. The reduction of the exchanged messages can significantly improve the performance of the system by lowering the computational demand on the side of the judges.

The interface between the intelligent agent system extension and the base core of Eclipse SUMO was provided by the mechanism of a device. Another modified component was the `SL2015` lane change model. This modified LC-model ensures that vehicles, which are forming a platoon, can change lanes together.

As a part of our current research, the ECN-judges were integrated into this ecosystem. Since the earlier system had been created by modifying some of the SUMO's C++ source-code, the new ECN-judges were also implemented by directly using it and the original codes of SUMO.

4.2 Integrating ECN-judges

In the previous system, an abstract class of intelligent judges had already been defined, therefore ECN-judge was implemented as a child of that abstract class. The abstract judge class uses the concept of *conflict classes*. A conflict class is a group of vehicles which can pass through an intersection simultaneously, which means they are equivalent, and can be treated, from a scheduling-theory point of view, as a large single entity.

Unfortunately, conflict classes are not entirely beneficial when TLS-programs are generated in real-time, because we cannot really differentiate vehicles into more than two classes: one class for those vehicles which currently receive a green light, and another class for those which do not receive a green light in a given moment. Hence, an ECN-judge has to change conflict classes of the vehicles when it switches phases, by e.g. removing them from the class which receives a green light and moving them to the class which currently receives a red light. By this method, the ECN-judges can be integrated into our previously proposed multi-agent system (see Figure 1).

One of the most important issues was to obtain the occupancy state of those edges (lanes) which join to the intersection which is controlled by an ECN-judge. To solve this

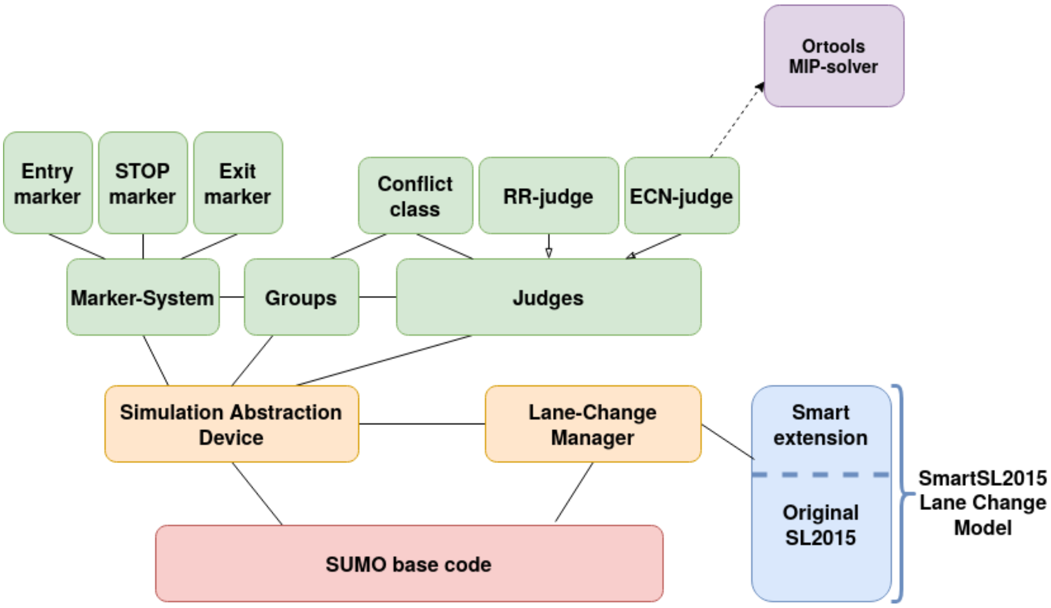


Figure 1: Overview of the extended multi-agent system, based on Eclipse SUMO. SUMO's core functions are interfaced by the two components colored as orange. The parts of the intelligent agent system are colored green. The MIP-solver of Ortools is an external library, developed by Google and capable of solving our integer programming problem.

problem, SUMO's TraCI library was used. This library contains functions (specifically the `libsumo::Edge::getLastStepOccupancy` function) which return the current occupancy state of a given lane. The ECN-judges use this value when calculating whether a congestion is about to form.

The ECN-signals are transmitted as a broadcast message between the judges. As a configuration input, every ECN-judge knows its topological neighbors, therefore when one of its neighbors sends an ECN-signal, indicating a congestion, the phase plans can be changed accordingly. This modified TLS-program will forcibly reduce the throughput towards the forming congestion.

The last problem was to integrate the IP-solver component into the extended Eclipse SUMO-based platform. The used solver is the *OR-Tools Mixed-Integer Programming* toolkit developed by Google. Technically it was simple to add this package, because OR-Tools also use `Cmake` build system. The performance of this toolkit seem to be convincing. As Table 1 shows, our intelligent system can run almost exactly as fast as the original SUMO code.

Simulator	**Scale 1**	**Scale 2**	**Scale 5**	**Scale 10**
Original SUMO	2841.80	1460.32	200.98	77.06
MAS SUMO	2797.78	1367.88	207.66	67.53

Table 1: Comparison of the performance of the original and the extended version of SUMO. Real time factors, provided by SUMO, regarding the scaling of the original traffic demand.

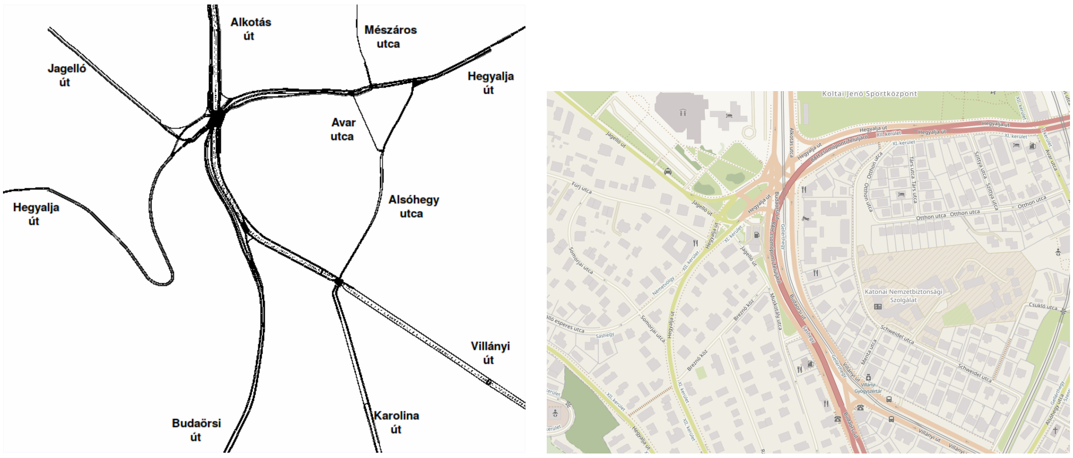


Figure 2: Simulated network of BAH-intersection (left). The central part of the intersection (right) (source: http://osm.org/#map=17/47.486/19.025).

5 Experiments

5.1 The Test-Scenario

Since banning vehicles with old combustion engines from historic city centers is a hot topic nowadays, we applied our system to a major intersection of Budapest, called the BAH-intersection[3] (see Figure 2). To the South of this intersection the M1 and M7 highways terminate, which presumably handle heavy traffic much more easily, than the area to the North from BAH, which is the historical town of Buda, or the residential areas at the eastern and western sides[4] of this intersection. Therefore, our presumption is that applying ECN-judges in this intersection would be beneficial in order to mitigate congestions in the inner-part of the city.

The BAH-intersection and the main roads of its surrounding were fed into Eclipse SUMO [5]. At BAH-intersection, three main roads and two smaller streets intersect at three different junction, topologically close to each other. Moreover, since most of the left turns are prohibited, they also form a bottleneck in the simulated network. For these reasons, ECN-judges shall be placed at these junctions, connected to each other (see Figure 3).

The simulated traffic demand was like a typical workday morning situation (in our measurements, we refer to this case as *Scale 1*[5]). For higher demands, the number of inputted vehicles of this original situation were upscaled by a factor of 2, 3,. . . 10.

[3] The abbreviation of BAH stands for the three biggest roads which intersect at this point of the city: Budaörsi road, Alkotás street and Hegyalja road.

[4] There is even a natural reserve (Sashegy – Eagle Hill) on the western part of BAH intersection.

[5] As exact values are currently not available from the Road Agency of Budapest, an estimated number of vehicles were used in our simulations. In the morning, the majority of the traffic is coming from the highways which terminate in the Budaörsi road. Significant traffic comes from the Jagelló and Hegyalja roads as well. The most vehicles want to go East on the Hegyalja road, because it drives to one of the bridges over the Danube. Alkotás road is a North-South corridor of the Buda-side, used by about the same amount of vehicles in both directions. About 40% of the traffic in our simulations went on the Hegyalja street Eastbound, and about 20% left the city on Budaörsi street, Southbound. Budaörsi street and Alkotás street (in both directions) handle about another 25% of the traffic. The other 15% of our traffic is randomly distributed among other, not-so-typical routes.

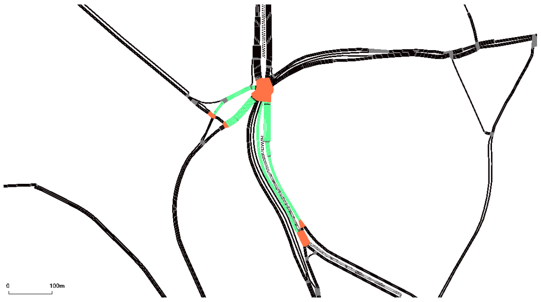


Figure 3: The connected ECN judges. The junctions which are colored as orange are controlled by ECN-judges, which are connected to each other.

5.2 Simulations and their Results

Two different situations were simulated. In the first situation, every traffic light was controlled by an ECN-judge (but only the central three were connected, practically speaking, the others were functioning as simple actuated traffic lights). In the second case, only three central, connected judges were of ECN-type, the others were running a simple Round-Robin scheduler. This allows us to compare the effect of the ECN-judges in itself, instead of comparing the results of the intelligent multi-agent system to the results of the traditional system.

From our previous measurements [3,4], the traffic flow and density values were known, and provide us with a basis for the comparison of the ECN-judge system to the unconnected, yet multi-agent system based solutions (Round Robin (RR) based, practically behaving like an actuated, phase-skipping traffic light) and to the traditional solutions as well. We compared cases when all the traffic controllers were ECN-judges (ECN-only) (much like RR, but instead of scheduling *phases* they schedule *directions*). In the case of ECN-mixed only the three intersection controllers shown on Figure 3 were ECN-type, every other intersections were RR-type.

As the results show (see Table 2 and Figure 4), at low demand levels (at Scale 1 and Scale 2), all of the tested solutions can provide roughly the same results. Then comes a point (around Scale 4), from where the ECN-judges can restrain the density of the traffic, which also means that the traffic flow is limited. As we scaled the traffic demand to higher levels (Scale 8 and Scale 10), throughput of both the traditional and the simple Round-Robin based intelligent solution started to degrade. On the other hand, the ECN-judges were able to stabilize the traffic density and the traffic flow at a certain level, almost regardless of the actual load.

	Scale 1		**Scale 2**		**Scale 4**		**Scale 8**		**Scale 10**	
	D	Q	D	Q	D	Q	D	Q	D	Q
Traditional	22.5	1319	45.0	2617	89.2	4577	137.1	3713	138.4	1098
RR-only	24.4	1267	44.9	2477	NA	NA	139.2	3352	144.7	1287
ECN-only	22.4	1211	44.8	2079	59.3	1987	77.2	2849	78.5	2750
ECN-mixed	22.4	1274	44.9	2120	71.3	2500	83.4	2957	102.8	3782

Table 2: Macroscopic parameters obtained from simulations with different systems. D stands for vehicle density in [$\frac{veh.}{km}$] and Q stands for traffic flow in [$\frac{veh.}{h}$] Unfortunately, the simulation with only Round-Robin judges, due to a yet unknown reason, did not provide an output for Scale 4.

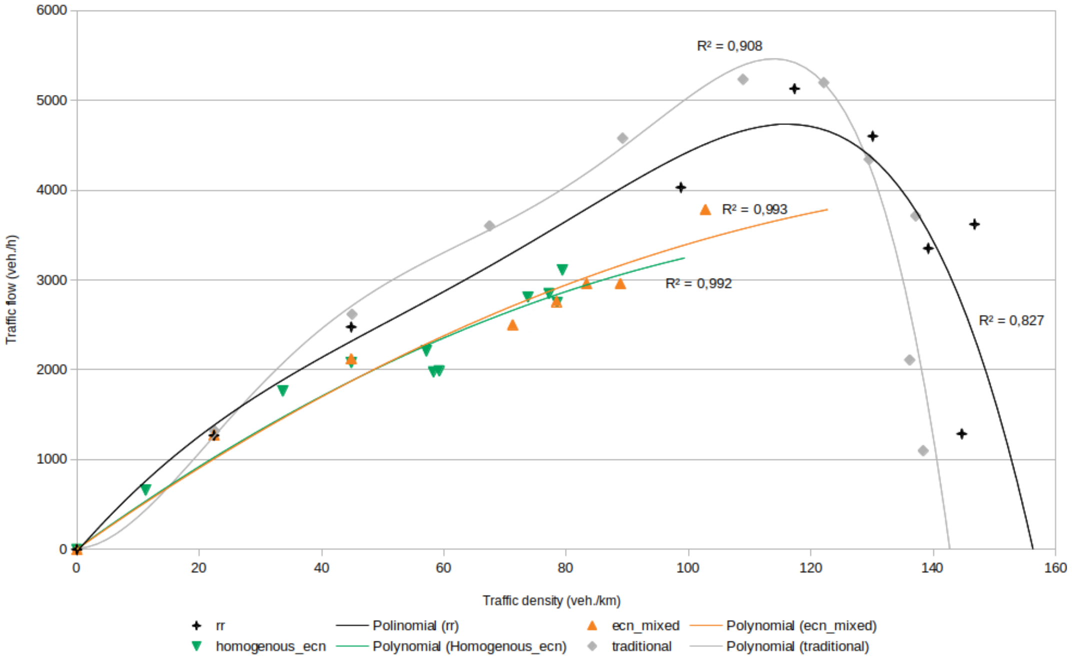


Figure 4: The macroscopic fundamental diagram, provided by the different types of systems. Systems with ECN-judges lack the degrading side of the diagram.

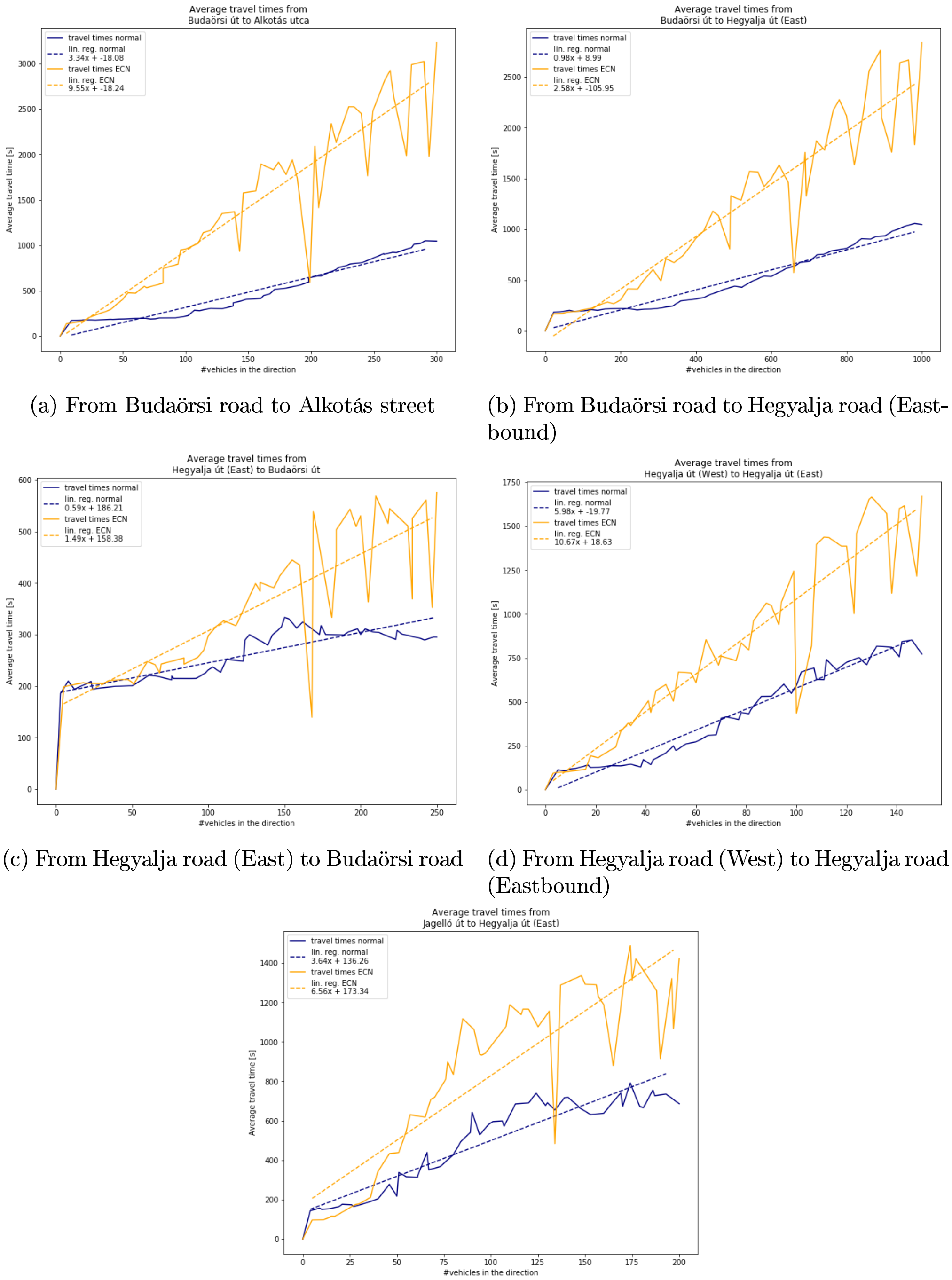
As we found out, that ECN-mixed case behaves slightly better than the ECN-only case, we conducted our measurements with the ECN-mixed setup.

Our expectation might be that the travel time through the congestion-protected edges is significantly increased. This influences the route-decisions of the vehicles, resulting in decreased traffic in this particular area. Thus, on alternative routes, the number of vehicles will be increased accordingly. To analyze the consequences, the average travel times of every vehicle flow were measured as a function of the number of inserted vehicles. As these empirical functions seem to be linear (above a given amount of vehicles), simple linear regressions was fitted to the data points. The equation of these straight line, depicted in Figure 5 can be used to analyze the effects of the ECN judges on the user equilibrium.

Using the `departDelay` and the `departPos` parameters of Eclipse SUMO, the increased probability of congestion on the incoming highways can be assessed[6], see Figure 6. The departure delay somehow reflects how much time it takes to turn on a particular edge (the greater number means more time to wait for a “hole”, which means a denser traffic). Moreover, the departure position reflects how long the congestion gets on the highways (the greater number represents a longer traffic jam). Surprisingly, we can see, the ECN judges create less dense traffic outside of the system perimeter, and in most cases, the length of the congestion is about the same as the traditional traffic control would be used.

Congestions dissolve periodically inside the ECN judge system. When a congestion has just dissolved, the newly incoming vehicles will find really light traffic on the roads, therefore they

[6] For this measurement, length of the Budaörsi street was increased to 5000 m, to model the incoming highways.



(a) From Budaörsi road to Alkotás street

(b) From Budaörsi road to Hegyalja road (Eastbound)

(c) From Hegyalja road (East) to Budaörsi road

(d) From Hegyalja road (West) to Hegyalja road (Eastbound)

(e) From Jagelló road to Budaörsi road (Eastbound)

Figure 5: Travel times and their empirical equations as the function of vehicles traveling in the same direction.

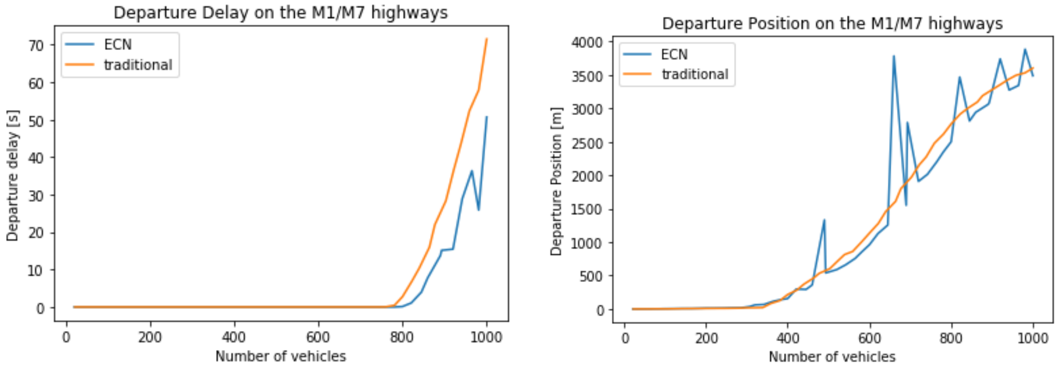


Figure 6: Congestion is increased on the incoming highways. Departure delay (left) is increased on the corresponding edge. Departure position (right) is only moderately increased on the corresponding edge.

can pass through the system more rapidly[7]. This pumping effect allows more distant cars to slowly but constantly move towards the BAH intersection. Perhaps this phenomenon causes on the averagely lighter traffic on the incoming highways.

6 Comparing to Other Approaches

Congestion is an acute problem evident to everybody, no wonder that there are plenty of ideas how to fight it. Some proposals have been tried in practice [6, 7], or only as a theoretical consideration, verified in simplified models by simulations [12, 13, 17–21, 25, 26, 28, 29].

Widely used agent-based paradigm will be even more visible and indispensable in the future, because the character of the possible, or advisable, or demanded level of intelligence for smart cars and their automated decision making capability is not yet clear. With respect to the hierarchically organized multi-agent systems (linking not only ordinary vehicles, but also local intersection managers, then even more centralized area managers, etc.), this paradigm was tried already, see e.g., [8–11, 14, 15, 23, 24, 27, 30].

Results resembling our approach the most are [9, 22, 24, 30]. In those solutions all intersection managers affected by the changes cooperate to adapt signal plans by consensus. In our approach, on the contrary, the affected intersections depend solely on the borders of the sheltered area and on the upstream-downstream direction of the flow. This way the communication load on the agents is less.

The level of an instant occupancy, sensed by detectors, is averaged over a time window, estimated similarly to [9, 24]. In our case the principal novelty is to base the cooperation of the intersection managers on the analogy to the computer network ECN congestion eliminating protocol. At a level of individual intersections signal protocols can be optimized also in a variety of ways, for reviews see [32, 33].

The real vehicle agents may not behave bona fide. Such possibility was considered in [12]. We assume that our agents are well disposed. It is of course a simplification and should be investigated in the future.

[7] Up to the point, when the roads become congested again.

Finally, we did not treat pairing congestion alleviation with route deviation [16,17], or multi-objective problems [31]. We assumed that congestion is the primary problem, and every other problem (related to health, safety and economics) will be almost automatically resolved, once congestions are removed from the city limits.

7 Conclusions

The obtained results can be interpreted that the ECN-judges effectively realize a traffic signal coordination. The aim of this signal coordination, however, differs from the traditional aim of trying to maximize the capacity of the road network. On the contrary, ECN-judges limit the throughput of the network.

Within these constraints, the system can work as a simple controlled road network with actuated traffic lights. If the traffic demand reaches a certain level, this limit will not be exceeded. Naturally, this policy permits to form congestions on the perimeter of the system (i.e. outside of a city), but also ensures that the traffic will be continuous within.

Such reduced traffic would be really beneficial for the residents of a city. The lack of extensive congestions will result in a healthier environment with less pollution, vibration and noise. Road safety will also be increased, therefore bike-riding or riding a scooter would become a more attractive alternative means of transportation.

In the future, a control algorithm shall be developed, which allows to set the traffic limitation to a desired number of vehicles. As far as we know, it strongly depends on the sensing of the congestion-forming.

References

[1] S. Floyd, K. K. Ramakrishnan, and D. L. Black, "The Addition of Explicit Congestion Notification (ECN) to IP." https://tools.ietf.org/html/rfc3168#section-1. last viewed 25. October, 2019

[2] P. Wagner, R. Alms, J. Erdmann, and Y.-P. Flötteröd, "Remarks on Traffic Signal Coordination", in EPiC Series in Computing, 2019, vol. 62, pp. 244–255.

[3] L. Alekszejenkó and T. P. Dobrowiecki, "SUMO Based Platform for Cooperative Intelligent Automotive Agents," in EPiC Series in Computing, 2019, vol. 62, pp. 107–123.

[4] L. Alekszejenkó and T. Dobrowiecki, "Intelligent Vehicles in Urban Traffic – Communication Based Cooperation," in 2019 IEEE 17th World Symposium on Applied Machine Intelligence and Informatics (SAMI), 2019, pp. 299–304.

[5] P. A. Lopez and M. Behrisch and L. Bieker-Walz and J. Erdmann and Y-P. Flötteröd and R. Hilbrich and L. Lüncken and J. Rummel and P. Wagner and E. Wießner, "Microscopic Traffic Simulation using SUMO," in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 2575–2582., url: https://elib.dlr.de/124092/

[6] How to Fix Congestion, Texas A&M Transp. Inst., https://policy.tti.tamu.edu/congestion/how-to-fix-congestion/, last viewed June 22, 2020

[7] M.D. Meyer, "A Toolbox for Alleviating Traffic Congestion and Enhancing Mobility", Inst. of Transportation Engineers, Washington, DC, 1997, https://rosap.ntl.bts.gov/view/dot/2145, last viewed June 22, 2020

[8] R.Z. Wenkstern, T.L. Steel, O. Daescu, J.H.L. Hansen, and P. Boyraz, "Matisse: A Large-Scale Multi-Agent System for Simulating Traffic Safety Scenarios", Ch 22 in J.H.L. Hansen et al. (eds.), Digital Signal Processing for In-Vehicle Systems and Safety, Springer Science+Business Media, LLC 2012

[9] B. Torabi, R.Z. Wenkstern, and R. Saylor, "A Self-Adaptive Collaborative Multi-Agent based Traffic Signal Timing System", 2018 IEEE Int. Smart Cities Conf. (ISC2), Kansas City, MO, USA, 16-19 Sept. 2018

[10] T. Tettamanti, A. Mohammadi, H. Asadi, I. Varga, "A two-level urban traffic control for autonomous vehicles improve network-wide performance", 20th EURO Working Group on Transp. Meeting, EWGT 2017, 4-6 Sept 2017, Budapest, Hungary, Transportation Research Procedia 27 (2017), pp. 913–920

[11] B. Beak, K. Larry Head, Y. Feng, "Adaptive Coordination Based on Connected Vehicle Technology", Transp. Research Record: J. of the Transp. Research Board, Vol 2619, Issue 1, 2017, pp. 1–12

[12] X. Zhang and D. Wang, "Adaptive Traffic Signal Control Mechanism for Intelligent Transportation Based on a Consortium Blockchain", IEEE Access, Vol 7, 2019. pp. 97281-97295.

[13] Y. Ren, Y. Wang, G. Yu, H. Liu, and L. Xiao, "An Adaptive Signal Control Scheme to Prevent Intersection Traffic Blockage", IEEE Trans. on Intelligent Transp. Systems, Vol. 18, No. 6, June 2017, pp. 1519-1528.

[14] A. Namoun, C.A. Marín, B. Saint Germain, Ni. Mehandjiev, J. Philips, "A Multi-Agent System for Modelling Urban Transport Infrastructure Using Intelligent Traffic Forecasts", in V. Mařík, J.L. Martinez Lastra, P. Skobelev (Eds.): HoloMAS 2013, LNAI 8062, pp. 175–186, Springer-Verlag Berlin Heidelberg 2013

[15] B. Torabi, R. Z. Wenkstern, and M. Al-Zinati, "An Agent-Based Micro-Simulator for ITS", 21st Int. Conf. on Intell. Transp. Systems (ITSC), Maui, Hawaii, USA, Nov 4-7, 2018, pp. 2556-2561

[16] R. Alqurashi and T. Altman, "Hierarchical Agent-Based Modeling for Improved Traffic Routing", 16 Oct 2019, Appl. Sci. 2019, 9, 4376, doi:10.3390/app9204376

[17] J. McBreen, P. Jensen and F. Marchal, "An Agent Based Simulation Model of Traffic Congestion" in Proc. of the 4th Workshop on Agents in Traffic and Transportation, Hakodate, May 2006, 43-49

[18] A. Olia, H. Abdelgawad, B. Abdulhai and S.N. Razavi, "Assessing the Potential Impacts of Connected Vehicles Mobility, Environmental, and Safety Perspectives", J. of Intell. Transp. Systems. Techn., Planning, and Operations, Vol 20, 2016, Issue 3, pp. 229-243

[19] Khan, S. M., "Connected and Automated Vehicles in Urban Transportation Cyber-Physical Systems", PhD Diss., Clemson University, 2019, https://tigerprints.clemson.edu/all_dissertations/2475, last viewed June 22, 2020

[20] X. Ban and W. Li, "Connected Vehicle Based Traffic Signal Optimization", Report C2SMART Center, USDOT Tier 1 Univ. Transportation Center, April 2018.

[21] S.M. Khan, M. Chowdhury, "Connected Vehicle Supported Adaptive Traffic Control for Near-congested Condition in a Mixed Traffic Stream", 14 Jul 2019, arXiv:1907.07243 [eess.SP]

[22] M.-D. Cano, R. Sanchez-Iborra, F. Garcia-Sanchez, A.-J. Garcia-Sanchez, and J. Garcia-Haro, "Coordination and agreement among traffic signal controllers in urban areas", ICTON 2016, Tu.A6.3

[23] M. Abdoos, N. Mozayani, and A.L.C. Bazzan, "Holonic multi-agent system for traffic signals control", Eng. Applications of Artificial Intelligence, Vol 26, Issue 5-6, May, 2013

[24] B. Torabi, R.Z. Wenkstern, R. Saylor, "A Collaborative Agent-Based Traffic Signal System For Highly Dynamic Traffic Conditions", 21st Int. Conf. on Intell. Transp. Systems (ITSC), Maui, Hawaii, USA, Nov 4-7, 2018

[25] H. Nakanishi and T. Namerikawa, "Optimal traffic signal control for alleviation of congestion based on traffic density prediction by model predictive control", Proc. of the SICE Annual Conf. 2016, Tsukuba, Japan, Sept 20-23, 2016, pp. 1273-1278

[26] M. Steingraover, R. Schouten, S. Peelen, E. Nijhuis, and B. Bakker, "Reinforcement Learning of Traffic Light Controllers Adapting to Traffic Congestion", BNAIC 2005, Proc. of the 7th Belgium-Netherlands Conf. on Artificial Intelligence, Brussels, Oct 17-18, 2005

[27] I. Tchappi Haman, V. C. Kamla, S. Galland, J. C. Kamang, "Towards an Multilevel Agent-based

Model for Traffic Simulation", The 6th Int. Workshop on Agent-based Mobility, Traffic and Transp. Models, Methodologies and Applications (ABMTRANS), Procedia Computer Science 109C (2017) 887–892

[28] K. Tumer, Z.T. Welch, and A. Agogino, "Traffic Congestion Management as a Learning Agent Coordination Problem", Multi-Agent Systems for Traffic and Transportation Engineering, 2009

[29] G. Orosz, R. E. Wilson and G. Stépán, "Traffic jams: dynamics and control", Phil. Trans. R. Soc. A 2010 368, 4455-4479

[30] J. France and A.A. Ghorbani, "A multiagent system for optimizing urban traffic", IEEE/WIC Int. Conf. on Intell. Agent Technology, 2003. IAT 2003.

[31] J. Jin, X. Ma, "A multi-objective agent-based approach for road traffic controls: application for adaptive traffic signal systems", Postprint of Paper VI: Jin, J. and Ma, X. (2018). A multi-objective agent-based approach for road traffic controls: application for adaptive traffic signal systems, under review, https://www.diva-portal.org/smash/get/diva2:1205233/FULLTEXT01.pdf, last viewed June 22, 2020

[32] E. Namazi, J. Li, and C. Lu, "Intelligent Intersection Management Systems. Considering Autonomous Vehicles: A Systematic Literature Review", IEEE Access, July 8, 2019, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8756239&tag=1, last viewed January 30, 2020

[33] L. Chen, C. Englund, "Cooperative Intersection Management: A Survey", IEEE Trans. on Intell. Transp. Systems, Vol 17, Issue 2, Feb. 2016, pp. 570-586, DOI:10.1109/tits.2015.2471812

A State Space of an ECN Judge

To understand the difference between the required amount of memory of a simple phase-skipping traffic light and an ECN-based traffic light, let us suppose that we have a grid-like road network, see Figure 7. Every intersection is equipped with the same type of traffic controller. Now, let us focus on a simple junction of this network. This junction is believed to be a plain, four-legged intersection.

If this junction is equipped with phase-skipping traffic lights, at least 4, but at most 8 phases are enough to provide safe passing through for every vehicle. Our actions can be to skip 0, 1, 2, or even 3 phases. This gives us the possibility to choose 0, 1, 2, or 3 phases out of four. Using the well-known formula this means that we have $\binom{4}{0} + \binom{4}{1} + \binom{4}{2} + \binom{4}{3} = 15$ choices. Generally, Equation (2) gives the number of choices, which an n-legged intersection can provide. Note that the number of states, in this case, is proportional to 2^n.

$$\sum_{k=0}^{n\ \ 1} \binom{n}{k} = 2^n - 1 \tag{2}$$

Now, let us consider the case of an ECN judge. As all traffic controllers in the road network are ECN-based, every *direction* can be restricted. However, it does not necessarily mean that a phase shall be skipped. It is much more alike a supplementary green light of a traditional system, which allows turning right, even in the time when the main lamp shows red. Therefore, not only the phases but all the directions have to be represented in the state space.

Directions can be modeled as a (directed) graph of which nodes are the incoming and outgoing streets. The edges of the graph represent the connections between every street. As a complete graph, with n nodes, has $\frac{n(n\ \ 1)}{2}$ edges, we have the option to choose directions "randomly" from this amount of possibilities. Analogously to the case of a phase-skipping traffic light, Equation (3) shows the required size of the state space. As we can see, it is

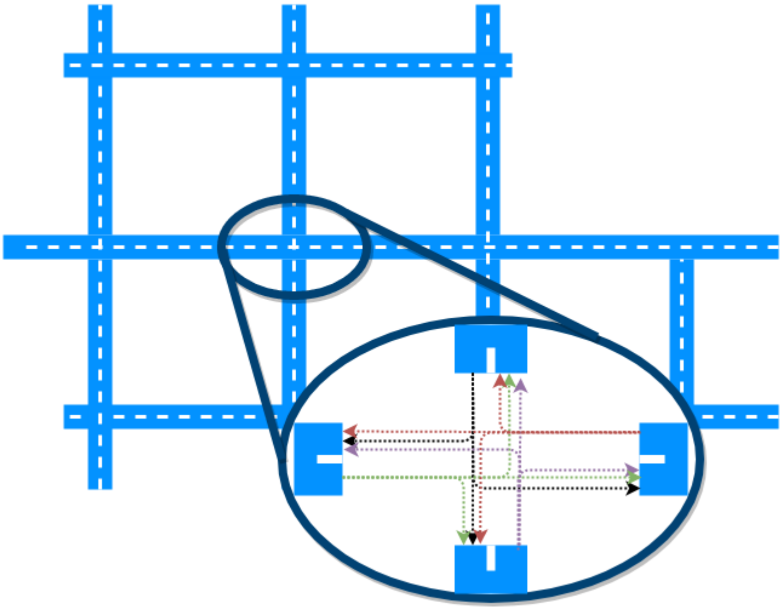


Figure 7: ECN judges controls directions, instead of phases. It means, storing its program requires exponentially more memory than a simple phase-skipping program.

proportional to $2^{\frac{n(n-1)}{2}}$ which is an exponential growth compared to the traditional case.

$$\sum_{k=0}^{n\quad 1}\binom{\frac{n(n\quad 1)}{2}}{k}=2^{\frac{n(n-1)}{2}}-1 \tag{3}$$

The controlling algorithm is likely to be realized by two components. One component is a *Look-Up Table, LUT*, and the other one is a component that searches in this LUT for proper configuration of the “traffic lamps”. Searching itself can be implemented powerfully, but the size of the LUT cannot be smaller than the actual size of the state space. It means, even a four-legged intersection would require a 63-sized LUT (compared to a phase-skipping controller’s requirement of a 15-sized LUT). If we have a greater intersection, with five legs, these numbers will be 1023 and 31, respectively. An even bigger, six-legged intersection with ECN judges will require a LUT capable of storing 32767 entries, meanwhile, a phase-skipping system would use only 63 entries.