SUMO Conference Proceedings

4 | 2023

TIB OPEN PUBLISHING

SUMO User Conference

02 – 04 May 2023

German Aerospace Center, Berlin

Editors:

Pablo Alvarez Lopez
Olaf Angelo Banse Bueno
Mirko Barthauer
Michael Behrisch
Benjamin Couéraud
Jakob Erdmann
Yun-Pang Flötteröd
Robert Hilbrich
Ronald Nippold
Peter Wagner

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

TIB OPEN PUBLISHING

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 4

SUMO User Conference 2023

02 – 04 May 2023, Berlin

Conference papers

https://doi.org/10.52825/scp.v4i

Editors

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

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

Mirko Barthauer, German Aerospace Center, Institute of Transportation Systems

Michael Behrisch, German Aerospace Center, Institute of Transportation Systems

Benjamin Couéraud, 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

Peter Wagner, 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 User conference.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.197

Published: 29 June 2023

Simulation-based origin-destination matrix reduction: a case study of Helsinki city area

Klavdiya Bochenina[1[https://orcid.org/0000-0001-6025-0552]], Anton Taleiko[1[https://orcid.org/1111-2222-3333-4444]], and Laura Ruotsalainen[1[https://orcid.org/0000-0002-4057-4143]]

[1] University of Helsinki, Finland

Abstract. Estimation of a travel demand in a form of origin-destination (OD) matrix is a necessary step in a city-scale simulation of the vehicular mobility. However, an input data on travel demand in OD matrix may be available only for a specific set of traffic assignment zones (TAZs). Thus, there appears a need to infer OD matrix for a region of interest (we call it 'core' area) given OD matrix for a larger region (we call it 'extended' area), which is challenging as trip counts are only given for zones of the initial region. To perform a reduction, we explicitly simulate vehicle trajectories for the extended area and supplement trip values in 'core' TAZs based on the recorded trajectories on the border of core and extended areas. To keep validation results consistent between extended and core simulations, we introduce edge-based origin-destination assignment algorithm which preserves properties of traffic flows on the border of the core area but also keeps randomness in instantiating simulation for the core area.

The experimental study is performed for Helsinki city area using Simulation of Urban MObility (SUMO) tool. The validation was performed using DigiTraffic data from traffic counting stations within the city area for workdays of autumn 2018. Validation results show that the reduced OD matrix combined with edge-based OD assignment algorithm keeps the simulated traffic counts in good agreement with results from the extended area simulation with average MAPE between observed and simulated traffic counts equal to 34%. Simulation time after reduction is equal to 20 minutes compared to 6 hours for the extended OD.

Keywords: Origin-destination matrix estimation, Traffic demand model, Urban mobility, Data-driven traffic simulation, SUMO

1. Introduction

Data-driven models of vehicular urban mobility are widely used to develop and to test strategies of future transportation, to estimate the economical, societal and environmental effects of traffic planning decisions and to develop novel algorithms for controlling vehicles and city infrastructure. The applicability of the model for these purposes is significantly determined by the extent to which it resembles existing patterns of traffic flows within the city. Thus, development of realistic simulation of urban vehicular mobility requires collecting, preprocessing and fusion of heterogeneous data sources including data about road network layout, traffic signal controls, speed limits, types and amounts of vehicles, and travel demand during different times of a day and different seasons.

The situation is complicated by the fact that the initial data which are required for creation and validation of a model may be not available directly for a certain area of interest. For example, data on travel demand may be delivered from another model and may be available not for the city itself but for the larger region. This poses a problem for a developer of how to extract and to compose data for training and validation of the model in a reproducible and time-efficient way.

In this study, we focus on the problem of estimating origins and destinations of the vehicles for the case when simulated area (we call it 'core' area) is located within larger ('extended') area, and data on travel demand are available only for this larger area. This is a typical case when travel demand is produced by an external four-step mobility model (see e.g. [1]). Given origin-destination (OD) matrix for an extended area, we aim to solve the following two problems:

- to infer origin-destination matrix for a core area;
- to estimate origins and destinations of vehicles (starting and ending edges) for a core area.

For the first problem, naïve approach is to use a submatrix of OD matrix for extended area, corresponding to traffic assignment zones which also belong to the core area. However, this underestimates travel demand as trips which start or end outside the core area are not accounted. For the second problem, a basic approach is to use random assignment for core OD matrix. The disadvantage of this approach is that traffic flows on the border of the core area become less accurate.

To tackle these problems, we use vehicle trajectories from the extended simulation to infer origins and destinations for the core simulation. Data on trajectories are used in our novel edge-wise OD assignment algorithm. This algorithm may be used to get arbitrary number of instances of core simulation with sufficient level of variability after a single run of extended simulation. As simulation of an extended area may be much more time-consuming than simulation of a core area, proposed approach can reduce overall time of modelling workflow.

The experimental study of the method is performed using case study of Helsinki city area using Simulation of Urban MObility (SUMO) tool [2]. The extended area comprises of 1972 traffic assignment zones while the core area (Helsinki municipalities) has 381 zones. In this case, a single run of extended simulation takes more than 6 hours. For the purposes of validation, we use the data from traffic counting stations located within the city. In the experimental study, we show that: (i) the core simulation reproduces traffic flows of the extended simulation, (ii) edge-wise assignment provides sufficient variability of simulation instances and at the same time does not significantly influence model quality. The code implementing proposed approach is available on GitHub [3].

The paper is organized as follows. Section 2 gives an overview of the related work. Section 3 contains formal problem statement. Section 4 describes the proposed method and its SUMO implementation. Section 5 describes the data for the experimental study, and the results are discussed in Section 6.

2. Related work

The problem of reduction of OD data has often been considered in the studies aiming at creation of validated models of vehicular traffic in particular urban areas. In the recent study [1], authors propose large-scale agent-based traffic microsimulation for Barcelona city. As in our study, extended area (577 traffic assignment zones) is simulated to get input data for core area (296 zones). However, instantiating of core simulation is performed by cropping the paths for extended simulation, which requires launching extended simulation each time when one needs to get new instance of core simulation. In our study, we propose edge-wise assignment to avoid multiple launches of time-consuming extended simulation. The case study of Nanjing, China, is considered in [4]. Initial values of OD matrix are supposed to be given, but they are improved by Adaptive Fine-Tuning algorithm to minimize the error between simulated SUMO results and real-world Radio Frequency Identification Data. This is an example of OD matrix calibration problem when an initial matrix is tuned to increase its correspondence to the observed urban data. In [5], the source of origin-destination matrix for traffic modelling of Köln, Germany, is Travel and Activity Patterns Simulation (TAPAS) framework which uses population information, data on points of interests within the city as well as the time use patterns. Authors report that the resulting demand still needed to be improved: TAPAS model provides demand not limited to vehicular traffic, and variability of traffic over short time scales is not realistic. To cope with that, initial OD matrix is adjusted (only trips corresponding to vehicular traffic were considered) and smoothed (random offsets were added to departure times).

Another strand of research is related to estimation of origin-destination matrix using various types of available data. When estimates of a number of travellers in different areas are provided, a gravity model [6] is commonly used. For instance, in [7], the gravity model is applied to infer OD matrix for Bogor city. For the case when data from urban sensors are available (such as link counts, flows and travel times), more sophisticated methods of data fusion are usually applied. In [8], time-dependent demand is estimated via Bayesian approach when the posterior distribution of OD matrices is updated based on traffic counts data. Traffic counts are also used for OD matrix estimation in [9], when an iterative bilevel framework is proposed to minimize the deviation between estimated and real-time link counts. [10] presents combination of approaches: initial OD matrix is estimated by gravity model, and after that the process of dynamic OD matrices' estimation is performed. The latter includes using a macroscopic traffic simulator to model the traffic flows and an optimization algorithm which aims to minimize the normalized variation between the historical and the simulated link flows. These studies assume that the input data are available for the whole area for which OD matrix is estimated. In contrast, in the current study, we focus on the case of OD reduction, that is, estimation of demand for a subarea of the initial area, which is rarely considered in the field of OD estimation.

In [11], authors consider the problem of origin-destination trip demand estimation for subarea analysis. They propose two-step procedure: (i) generation of induced OD demand for a subarea network, (ii) OD updating based on the induced demand and archived traffic measurements. Generation of induced demand is performed based on the path-based traffic assignment results as in our study; however, in [11] the focus is mostly on demand calibration while we consider algorithmic and computational aspects of demand estimation.

Beyond the estimation of origin-destination matrix, there are algorithms and tools for trips assignment available at urban mobility simulation frameworks. In [12], authors perform experimental comparison of different demand generation tools available in SUMO. They divide all tools in two groups: (i) countless, which do not require any extra data – named randomTrips, SAGA and randomActivityGen, (ii) tools which are using traffic counts – named dfrouter, flowrouter, cadyts and routesampler. For the considered use case (Wildau, Germany) routesampler showed the best results in terms of root mean squared error of vehicle count and network coverage.

3. Problem statement

Let's assume that we have an initial origin-destination (OD) matrix $A_{m\times m}$, where m is a number of traffic assignment zones (TAZs). Each element of the matrix $a_{ij}, i, j \in 1, \dots, m$ represents the number of trips between TAZ i and TAZ j during a certain time period (e.g. one hour). The set of TAZs for matrix A is denoted as P. We will call matrix A an *extended OD matrix*, area covered by TAZs from P – an *extended area*, and simulation of vehicle movement according to origins and destinations in matrix A – an *extended simulation*.

Let's also assume that we have another set of TAZs, denoted as Q, which is a subset of the extended area: $Q \subset P$. We will call Q a core area. The problem is to obtain an instance of core simulation for area Q given A, P, Q as inputs, that is, to induce core simulation from extended one.

Extended and core areas are depicted in Figure 1a. TAZs in Figure 1a are shown schematically as squares for a simplicity, but they may have any shape which is usually represented in polygon format. Depending on if origin/destination zones belong to Q, or to $P-Q$, all trips may be categorized into four groups:

1. in-in, including trips starting and ending in Q.
2. in-out, including trips starting in Q and ending in $P-Q$.
3. out-in, including trips starting in $P-Q$ and ending in Q.
4. out-out, including trips starting in $P-Q$ and ending in $P-Q$.
   a) out-out trips where all trajectory belongs to $P-Q$.
   b) out-out trips where part of the trajectory belongs to Q, and part of the trajectory belongs to $P-Q$.

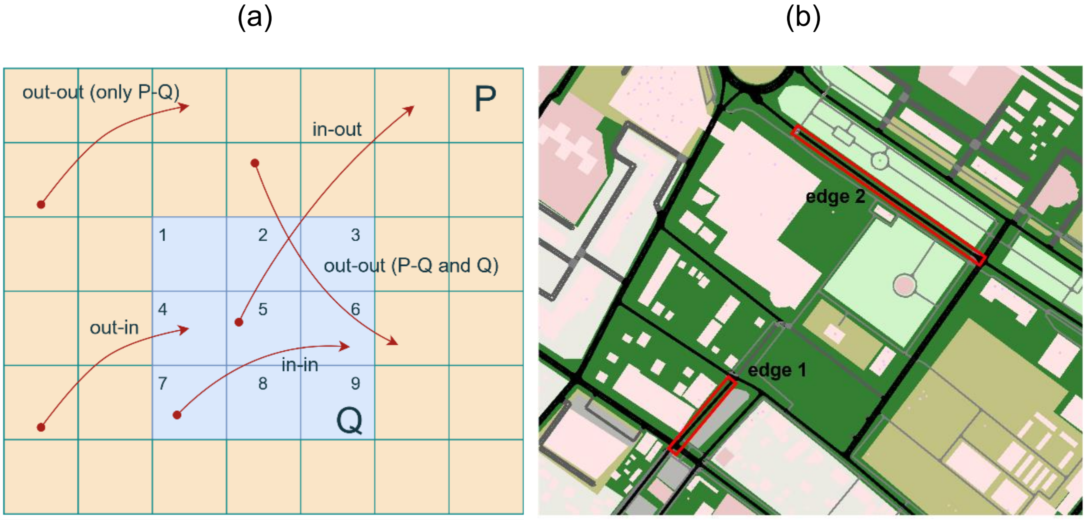


Figure 1. (a) Extended (P) and core (Q) simulation areas. By arrows, different types of trips in extended simulation are presented. “in” denotes trips which start/end in Q, “out” denotes trips which start/end in P. Core area Q is depicted in blue, yellow color depicts TAZs belonging to $P-Q$. Numbers represent indices of traffic assignment zones in Q.
(b) Edge assignment problem for a fixed TAZ.

If a trip does not pass through core area Q, it is not needed to be reproduced in a core simulation. Therefore, for the core simulation, all trips should be accounted except of type 4a. Moreover, only in-in trips (type 1) would have the same origins and destinations for extended and core simulations. For in-out, out-in, and out-out (passing both $P-Q$ and Q) trips only part of the trajectory passes through Q. This means that for these types of trips we need to find new starting/ending points of trips which are inside Q. We call this problem an origin-destination reduction problem and address it in this study.

The origin-destination reduction may imply two steps:

1. Estimation of a core origin-destination matrix $B_{n\times n}$, where n is a number of traffic assignment zones in Q, $n<m$. Rows and columns of matrix B correspond to TAZs for the core area. It should include all the trips from matrix A which have at least part of the path inside core area Q. For example, if we assume that we have 5 trips in Figure 1a represented with arrows, non-zero elements of matrix B will be $B_{2,6}=1$, $B_{4,4}=1$, $B_{5,3}=1$ and $B_{7,6}=1$.
2. Trip origin/destination edge assignment, executed for each trip in the core OD matrix B. Each traffic assignment zone (Figure 1b) has a correspondent road network which may be represented as a set of edges which are inside this TAZ. To perform simulation, for each trip one needs to specify starting and ending edges. The most common

way to assign the edges is a random selection from the set of edges for the origin/destination TAZ.

It is worth to mention that in general step 2 (origin/destination edge assignment) does not require stating core OD explicitly (step 1), but not vice versa (after having an OD matrix, one still needs to specify starting and ending points of the trajectories).

4. Method

As it was described in Section 3, a problem of reduction of extended simulation to core simulation consists of two steps: (i) origin-destination matrix reduction, (ii) edge assignment. Both steps are presented in Figure 2.

Origin-destination matrix reduction is presented by steps 1, 2, 3, 4a, 5, 6a, 6c in Figure 2. The input for the algorithm is the OD matrix for an extended area, denoted as A in Section 3. For each pair P_i and P_j from traffic assignment zones for a core area (which correspond to rows and columns of matrix A) and for each trip the random assignment of origin and destination edges is performed (step 1). After that, SUMO simulation for extended area is launched (step 2). The result of the simulation are vehicle trajectories (step 3). Given the vehicle trajectories, one may determine for in-out, out-in and out-out trips the TAZs from Q which serve as origin and destination areas for a core simulation (step 4a). Then, the resulting OD matrix for a core area is composed (steps 6a, 6b) from a submatrix of OD matrix A for in-in trips (step 5), and OD matrices for other types of trips created at step 4a. The pseudocode for OD matrix reduction is presented in Table 3.

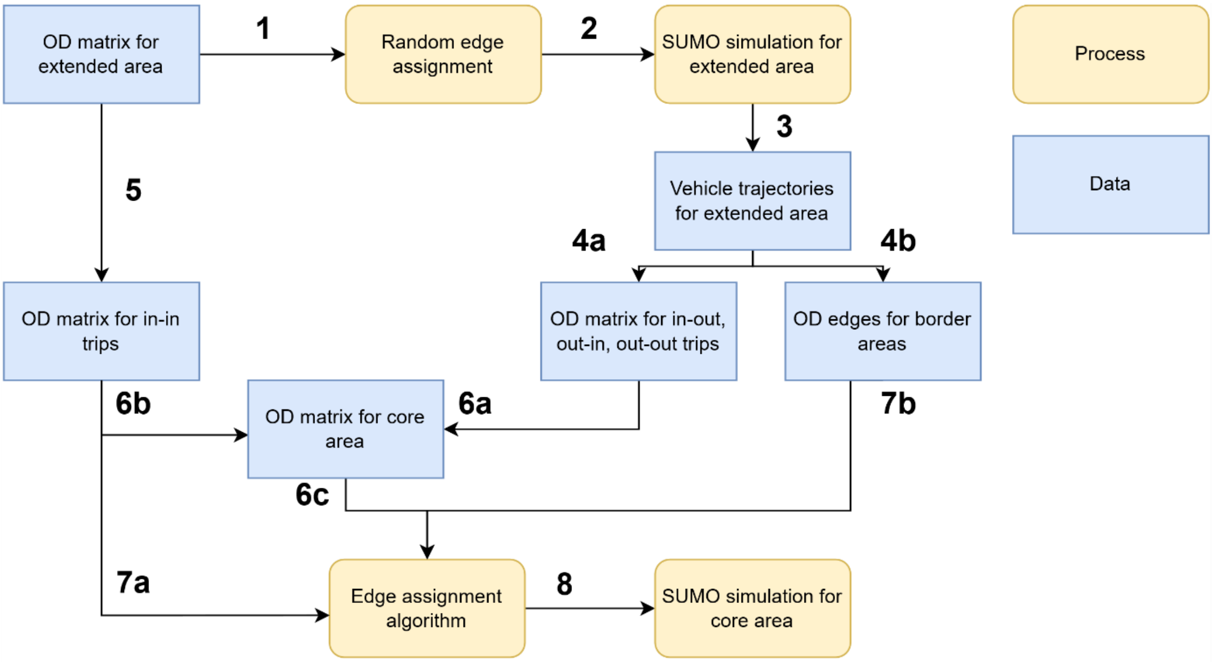

OD matrix for extended area
1
Random edge assignment
2
SUMO simulation for extended area
Process
Data
3
Vehicle trajectories for extended area
5
4a
4b
OD matrix for in-in trips
OD matrix for in-out, out-in, out-out trips
OD edges for border areas
6b
OD matrix for core area
6a
7b
6c
7a
Edge assignment algorithm
8
SUMO simulation for core area

Figure 2. Simulation-based OD reduction: a main scheme.

The second output from parsing vehicle trajectories depicted in Figure 2 (step 4b) are indices of edges when a vehicle enters or exits the core area. These edges are also used to determine new origin and destination traffic assignment zones for a core simulation (Table 3).

Table 1. Pseudocode for origin-destination matrix reduction step.

Functions: *originExtTAZ*(t) – get origin TAZ from matrix A for a trip t,
destExtTAZ(t) – get destination TAZ from matrix A for a trip t,
coreEntryEdge(t) – get first edge of a trip t belonging to core area Q,
coreExitEdge(t) – get last edge of a trip t belonging to core area Q,
getTAZ(edge) – get TAZ index by an edge index

```
  Q ← $0$ // initialize reduced OD matrix Q with zeroes
  for each trip in the extended simulation:
      if trip is of type “in-in”: // copy origin and destination TAZ from matrix A
          originTAZ = originExtTAZ(trip); destTAZ = destExtTAZ(trip);
// for out-in/out trips, find TAZ to which an entry edge in core simulation belongs
      if trip is of type “out-in” or trip is of type “out-out”:
          originTAZ = getTAZ(coreEntryEdge(trip));
// for in-out and out-out trips, the same for destination edge
      if trip is of type “in-out” or trip is of type “out-out”:
          destTAZ = getTAZ(coreExitEdge(trip));
// increment number of trips for the pair of origin and destination zones
  Q[originTAZ, destTAZ] += 1
```

The second stage of a simulation-based OD reduction is an edge assignment after which one can start simulation for a core area (step 8). The simplest strategy of the edge assignment is a random assignment (step 6c). With random assignment, for each pair of origin and destination TAZs for core OD matrix Q we select a trip and assign a starting and an ending edge of the trip by random selection of an edge from corresponding TAZs. The drawback of this approach is that we do not account pre-calculated flow directions from an extended simulation. However, traffic flows coming to the core area mostly use major roads which serve as main entrances to the city. Random assignment strategy does not account for the size of the roads, and then leads to unrealistic flows on the border of a core area.

Regarding that fact, we propose another assignment strategy that we call edge-wise assignment (steps 7a and 7b in Figure 2). The idea of the edge-wise assignment is to keep edges which were used by vehicles to enter and to exit the core simulation area. From the other side, if one just keeps all the trajectories from the extended simulation, one will get only one fixed instance of a core simulation. As our goal here is to provide input data for multiple instances of core simulation, we need to keep some randomness in the assignment process. Thus, the proposed algorithm for edge-wise assignment is summarized in Table 2.

For trips which start and end inside the core area, we apply random assignment. For trips which start and/or end in the extended area, we fix those edges from extended simulation which correspond to entry and/or exit edges which vehicles use to enter/exit the core area.

Table 2. Edge-wise assignment algorithm. RA denotes random selection of edge within corresponding TAZ, / denotes selection of entry/exit edge from the extended simulation.

Case (type of trip)	Assignment strategy
in-in	Random assignment (RA)
out-in	/ (originTAZ), RA (destTAZ)
in-out	RA (originTAZ), / (destTAZ)
out-out	/ (originTAZ), / (destTAZ)

The final scheme of the implementation of simulation-based OD reduction in SUMO is presented in Figure 3. It shows programming routines and data files which are used during simulation and validation processes. For routing, SUMO's default routing procedure is used. To find positions of vehicles on the border of the core area, we use SUMO FCD device because we need to check all border edges (the other option was to use induction loops, but their manual placement is too time-consuming and also needs to be redone for each new core area).For out-in and out-out trips we also record departure times of the vehicles from extended simulation to make starting time of vehicles in a core simulation more realistic. To validate the results of the core simulation, we use data on traffic counts from traffic counting stations (TCS), so we record simulated traffic counts for the same positions where real traffic counting stations are located.

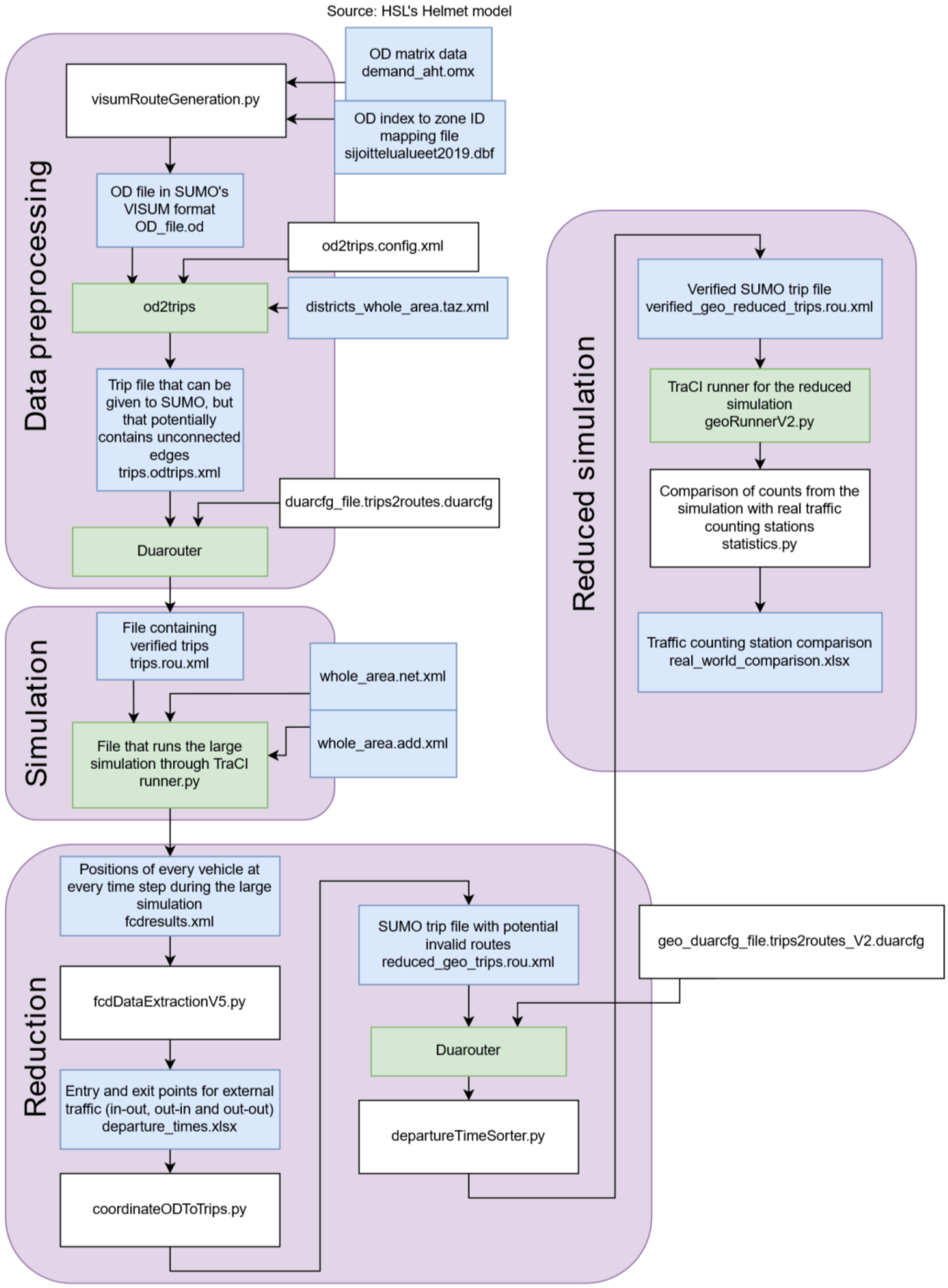

Source: HSL's Helmet model
Data preprocessing
visumRouteGeneration.py
OD matrix data demand_aht.omx
OD index to zone ID mapping file sijoittelualueet2019.dbf
OD file in SUMO's VISUM format OD_file.od
od2trips.config.xml
od2trips
districts_whole_area.taz.xml
Trip file that can be given to SUMO, but that potentially contains unconnected edges trips.odtrips.xml
duarcfg_file.trips2routes.duarcfg
Duarouter
Simulation
File containing verified trips trips.rou.xml
whole_area.net.xml
whole_area.add.xml
File that runs the large simulation through TraCI runner.py
Reduction
Positions of every vehicle at every time step during the large simulation fcdresults.xml
fcdDataExtractionV5.py
Entry and exit points for external traffic (in-out, out-in and out-out) departure_times.xlsx
coordinateODToTrips.py
SUMO trip file with potential invalid routes reduced_geo_trips.rou.xml
geo_duarcfg_file.trips2routes_V2.duarcfg
Duarouter
departureTimeSorter.py
Reduced simulation
Verified SUMO trip file verified_geo_reduced_trips.rou.xml
TraCI runner for the reduced simulation geoRunnerV2.py
Comparison of counts from the simulation with real traffic counting stations statistics.py
Traffic counting station comparison real_world_comparison.xlsx

Figure 3. SUMO implementation of simulation-based OD reduction.

4. Data

In this study, we test the proposed approach of OD reduction using Helsinki city area as an example. Initial traffic assignment zones and origin-destination matrices were obtained from Helmet [13], transport demand model system developed by HSL (the Helsinki Regional Transport Authority).

Extended and core simulation areas are shown in Figure 4. Extended area includes 1972 TAZs, which are the same as the zones used in the Helmet model. The core simulation area includes 381 TAZs which cover municipalities included in a Helsinki city area. One may observe that traffic assignment zones have different sizes being fine-grained for dense urban areas.

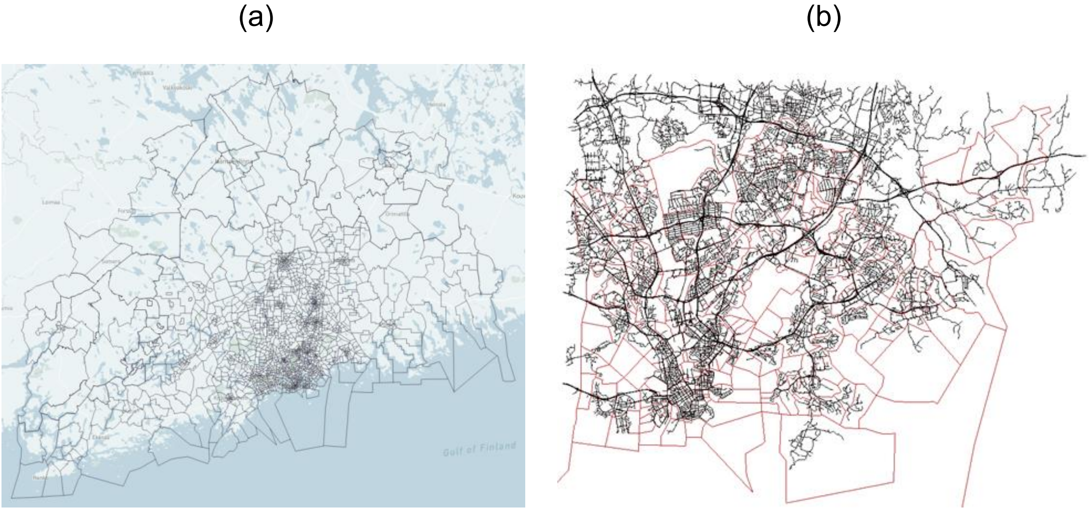


Figure 4. Traffic assignment zones: (a) for the extended area, (b) for the core area.

Origin-destination data include matrices for different transportation means (e.g. private car, bicycle, truck, van) and different times of the day (morning rush hour, day hour, evening rush hour). To get OD matrix for passenger transportation, we summed up matrices for private cars and vans. Each element of a matrix is a number of trips between two traffic assignment zones during a selected time of a day. In this study, we created a model for morning rush hour (07:20-08:19).

Demand matrices in Helmet are calculated using a set of models and data including land use data, growth factors of external traffic, car ownership data, models of destination and mode choice, applied for a particular time period. In this study, demand matrices were generated based on the data from workdays, Autumn 2018. Available demand matrices represent average demand for e.g. morning rush hour over this period. We used the same time period while collecting validation data from traffic counting stations.

For validation purposes, we used a Digitraffic dataset [14] containing hourly traffic counts for a set of 15 traffic counting stations (TCSs) located within a core simulation area. All stations may be divided into two types:

- border TCSs, which are located at the edge of a core simulation area and mostly measure flows of traffic coming from / to an extended simulation area;
- inner TCSs, which are located within a core simulation area.

From all available TCSs, 7 are border TCSs and 8 are inner TCSs.

Data for morning rush hour for separate days were averaged to get mean traffic counts comparable with Helmet demand matrices.

Traffic assignment zones were uploaded to SUMO together with Helsinki city infrastructure network obtained from OpenStreetMap. The traffic assignment zones were originally in shapefile format (.shp) and had to be converted into SUMO's TAZ format (.taz). This was done by first converting the original file into OpenStreetMap format (.osm) using Java OpenStreetMap Editor and from OpenStreetMap format to SUMO's polygon format (.poly) using a selfmade Python script. After that the tool edgeInDistricts.py that comes with SUMO was used to create a TAZ file from the polygons. Traffic counting stations were simulated as induction loops placed on all lanes in both directions at the same locations as real TCSs. Counts from the same direction were saved to the same file.

In the experimental study, default SUMO parameters were used.

Python implementation of the algorithms proposed in the study is available on GitHub [2].

5. Experimental study

Table 3 shows the results of the comparison of validation metrics for two instances of SUMO simulation: simulation of the extended area for initial Helmet OD matrix and simulation of the reduced area with origins and destinations assigned by a proposed algorithm. Metrics are calculated for all 15 traffic counting stations (TCSs), as well as separately for border and inner TCSs.

We may observe that average mean absolute percentage error (MAPE) is between 34% and 45%. Traffic flows at border stations are reproduced significantly better (by 10%) than for inner stations. This may be explained by the fact that border stations are located at main entries / exits to the city area (Figure 5) and measure in/out and out/in traffic flows. These flows are generally less dependent on routing procedure than flows simulated within the city. Second observation is that there is no difference of MAPE between initial and reduced simulations which shows that our algorithm performs reduction correctly.

Generally, there is more traffic counting events in Digitraffic data than in SUMO model. For example, for extended area simulation we have sum of traffic counts for all stations equal to 51,3K in real data, and to 42,2K in the model (16% difference). Here we also see that border stations are reproduced better than inner stations (7-10% difference for border stations and 22-23% for inner stations). This, again, supports our conclusion about larger influence of routing results for internal city area than for entries and exits to and from the city. Reduction process results in slightly (2%) less amount of traffic counts. This may be related to the cases when routing procedure cannot find the path from an origin edge to a destination edge (these trips are eliminated from simulation), because the amount of these cases tend to be larger for smaller areas. Summarizing, there is not significant difference with basic model after reduction considering both metrics, that is, the reduction algorithm provides consistent results.

Table 3. Comparison of traffic flow metrics for initial and reduced areas. EA – extended area, RA – reduced area, B/I/A – border/inner/all traffic counting stations. For example, EA-I denotes inner stations for an extended area.

Metric	EA-B	EA-I	EA-A	RA-B	RA-I	RA-A
MAPE of traffic counts, averaged by TSC, %	35	45	40	34	45	40
Difference of sums of traffic counts, %	7	22	16	10	23	18

Figure 5 shows MAPE for distinct traffic counting stations. Here we show absolute observed traffic volumes and MAPE, averaged for both road directions. Stations marked with green label have MAPE less than average MAPE for border or inner stations. Red labels denote MAPE larger than average. We may see that stations located in the northern part of Helsinki are reproduced better than in the southern and eastern parts. In the extension of this study, flows crossing ‘red’ stations need to be considered manually in more details.

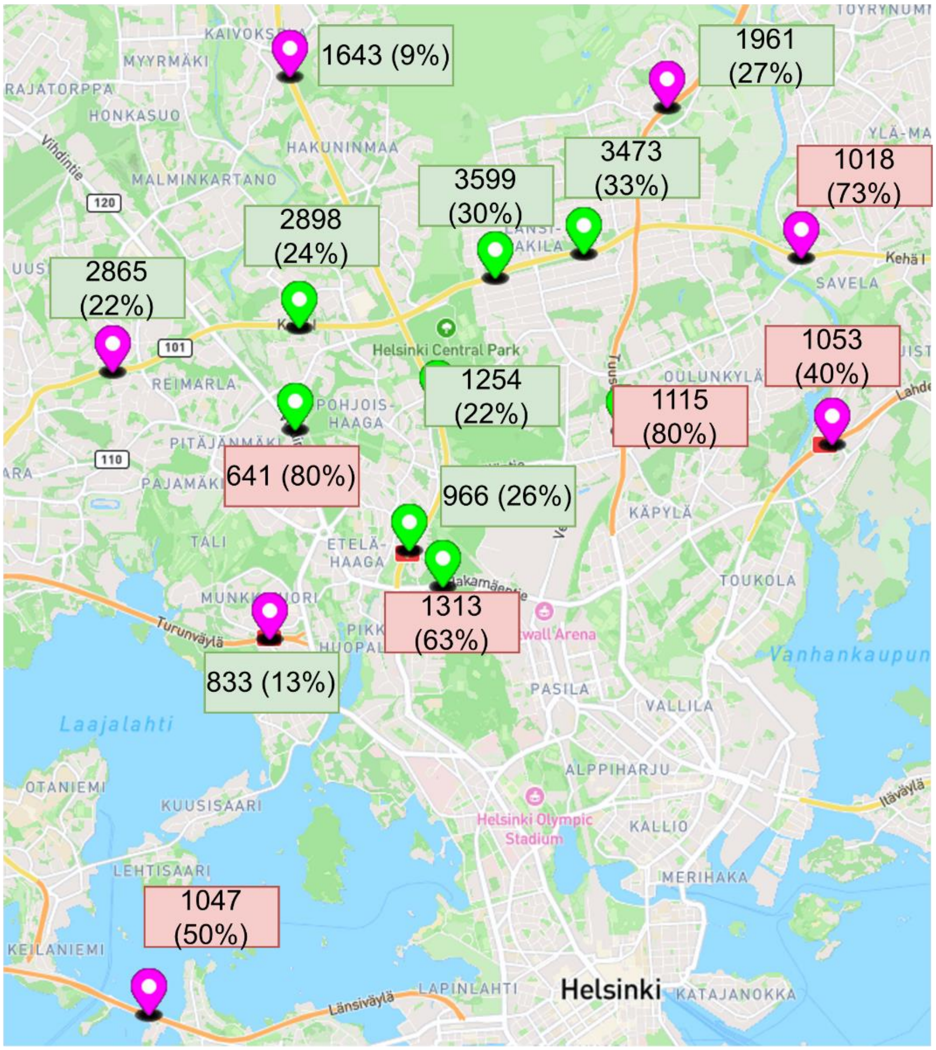


Figure 5. Absolute traffic volumes and MAPEs for different TCS. Border TCS have magenta color, inner TCS have green color. Labels of TCS are of the following format: observed vehicle count (MAPE), green / red labels denote MAPE smaller / larger than average.

Validation results show moderate correspondence of SUMO model to the observed traffic counts which may be explained by peculiarities of the initial data. Firstly, Helmet OD matrices are produced based on basic 4-step traffic simulation model [15] using census and land usage data, and then, are not well tailored to actual traffic count measurements. Secondly, initial OD matrices cover more than 100x larger area than Helsinki city area (a square of the extended area is equal to 24500 km^2 while reduced area is only 210 km^2), that is, they were aimed to reproduce coarse-grained patterns of vehicular mobility rather than city-scale behaviour. The overall quantity of simulated traffic resembles observed data with error equal to 16% but the distribution of cars between the roads is still not reproduced sufficiently good (MAPE is 40%). To give an idea about the typical values of reported metrics, in [1] it is stated that “according to the established literature, any value of [%RMSE] below 30% can be considered good”. The current validation quality of our Helsinki model may be further improved using several approaches including OD calibration based on traffic measurements, tuning hyperparameters of models and using SUMO calibrator objects for removing or inserting vehicles according to the desired flows on the inspected edges. We consider this as an extension of the current study as the main goal of this paper was to investigate the quality of reduction process itself.

To check that a proposed algorithm of edge assignment allows for getting different instances of core simulation while keeping correspondence of simulated traffic counts to the observed ones, we analysed the results of 10 different runs of core simulation with edge-wise origin/destination assignment algorithm. To measure the level of randomness between different runs, we calculate coefficient of variation c_v (1) of simulated counts for different traffic counting stations:

$$c_v^i = \frac{\sigma_i}{\mu_i}. \tag{1}$$

Here σ_i is a standard deviation of simulated traffic counts for i-th traffic counting station, μ_i is a mean of simulated traffic counts for i-th TCS.

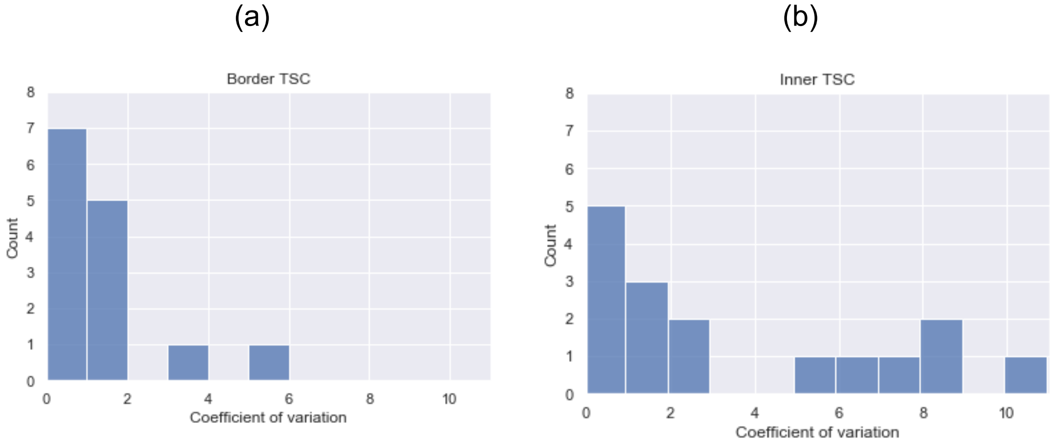


Figure 6. Coefficient of variation for simulated traffic counts: (a) border TCS, (b) inner TCS.

Figure 6 shows distributions of coefficients of variation for border (a) and inner (b) traffic counting stations. Here we present a separate value for each direction (14 values for 7 border stations, 16 values for 8 inner stations). The mean value of c_v for border stations is equal to 1.3%, for inner stations is equal to 3.5%. Variation of traffic counts for border stations is in the range 0.05%-5%, and for inner stations is in the range 0.4%-10%. Thus, we see that in both cases edge-wise assignment allows for getting random instances of a core simulation. At the same time, the level of randomness is lower for border stations than for inner ones. It is explained by the essence of the proposed algorithm because for border areas larger number of edges are fixed and do not change between runs as they are inherited from extended simulation to keep validation metrics sufficiently high. The latter is confirmed by the average value of coefficient of variation for MAPE which is equal to 6% for these 10 runs. Thus, proposed algorithm allows for generation of random instances of core simulation with sufficiently stable values of validation metrics.

6. Conclusions and discussion

In this study, we consider the problem of subarea demand estimation. Given origin-destination matrix for a larger area, we aim to infer origin-destination matrix and perform trips assignment for the reduced area. Proposed approach is simulation-based which means that we use traffic modelling tool to get the results for the extended area, and then post-process these results to get input data for the simulation of the core (reduced) area. As simulation of the extended area may be time-consuming, to avoid multiple runs of extended simulation in case when one wants to test different simulation instances for a core area, we proposed edge-wise origin/destination assignment heuristic. The experimental study for Helsinki city area has showed the applicability of our approach for the problem of OD reduction.

Proposed method is mainly purposed for reconstructing external traffic flows coming from an extended area to a core simulation area (and vice versa), that is, reproducing the observed values of traffic counts for border traffic counting stations. To reduce MAPE for inner traffic counting stations, methods of data-driven OD calibration may be applied. For example, in [11] authors propose dynamic OD estimation method for sub-area analysis which includes iterative OD updating procedure based on induced OD demand (similar to our reduced OD) and archived traffic measurements. Then, a two-step procedure for minimizing the errors may be proposed as an extension of current study:

- calibrating SUMO parameters (e.g. device.rerouting.probability and weights.priority-factor using grid search as in [1]) using traffic counts from edge TCS;
- calibrating reduced OD matrix using traffic counts from inner TCS.

Proposed procedure of OD reduction may be applied for any case study. The only assumptions here are that a core area is a subset of an extended area, and that borders of traffic assignment zones within core area are the same as corresponding zones in extended area. To support a decision of this typical task for a modeler, we share a code repository [2] with implementation of OD reduction procedure which may be reused during creation of large-scale traffic models for different cities.

Data availability statement

The data supporting the results of contribution can be accessed on GitHub [3].

Underlying and related material

The code of the implemented method as well as the manual on its usage can be accessed on GutHub [3].

Author contributions

Klavdiya Bochenina – conceptualization, formal analysis, investigation, methodology, validation, visualization, writing – original draft, writing – review and editing; Anton Taleiko – data curation, software, visualization, writing – review and editing; Laura Ruotsalainen – supervision, writing – review and editing.

Competing interests

The authors declare that they have no competing interests.

Funding

This work was supported by the Academy of Finland, project 347197 Artificial Intelligence for Urban Low-Emission Autonomous Traffic (AIforLEssAuto), and the Department of Computer Science, University of Helsinki.

References

1. J. A. Sánchez-Vaquerizo, “Getting Real: The Challenge of Building and Validating a Large-Scale Digital Twin of Barcelona’s Traffic with Empirical Data,” ISPRS Int J Geoinf, vol. 11, no. 1, Jan. 2022, doi: https://www.doi.org/10.3390/ijgi11010024.

2. M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, “SUMO – Simulation of Urban MObility: An Overview,” in Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation, Oct. 2011, pp. 23–28.
3. “SUMO Helsinki city traffic model, github repository”. https://github.com/helsinki-sda-group/sumo-hki-cm (04.04.2023).
4. C. Zhu, J. Wu, and A. Kouvelas, “Demand estimation and spatio-temporal clustering for urban road networks,” in 9th Symposium of the European Association for Research in Transportation (hEART 2020), 2020, doi: https://www.doi.org/10.3929/ethz-b-000456499.
5. S. Uppoor and M. Fiore, “Large-scale urban vehicular mobility for networking research,” in 2011 IEEE Vehicular Networking Conference (VNC), Nov. 2011, pp. 62–69. doi: https://www.doi.org/10.1109/VNC.2011.6117125.
6. P. C. E. Bouchard R. J., “Use of gravity model for describing urban travel,” Highway Research Record, vol. 88, p. 5, 1965.
7. I. Ekowicaksono, F. Bukhari, and A. Aman, “Estimating Origin-Destination Matrix of Bogor City Using Gravity Model,” IOP Conf Ser Earth Environ Sci, vol. 31, p. 012021, Jan. 2016, doi: https://www.doi.org/10.1088/1755-1315/31/1/012021.
8. H. Yu, S. Zhu, J. Yang, Y. Guo, and T. Tang, “A Bayesian Method for Dynamic Origin–Destination Demand Estimation Synthesizing Multiple Sources of Data,” Sensors, vol. 21, no. 15, p. 4971, Jul. 2021, doi: https://www.doi.org/10.3390/s21154971.
9. X. Zhou, X. Qin, and H. S. Mahmassani, “Dynamic Origin-Destination Demand Estimation with Multiday Link Traffic Counts for Planning Applications,” Transportation Research Record: Journal of the Transportation Research Board, vol. 1831, no. 1, pp. 30–38, Jan. 2003, doi: https://www.doi.org/10.3141/1831-04.
10. A. Abadi, T. Rajabioun, and P. A. Ioannou, “Traffic Flow Prediction for Road Transportation Networks With Limited Traffic Data,” IEEE Transactions on Intelligent Transportation Systems, pp. 1–10, 2014, doi: https://www.doi.org/10.1109/TITS.2014.2337238.
11. X. Zhou, S. Erdogan, and H. S. Mahmassani, “Dynamic Origin-Destination Trip Demand Estimation for Subarea Analysis,” Transportation Research Record: Journal of the Transportation Research Board, pp. 176–184, 2006, doi: https://www.doi.org/10.1177/0361198106196400119
12. M. Behrisch and P. Hartwig, “A comparison of SUMO’s count based and countless demand generation tools,” SUMO Conference Proceedings, vol. 2, pp. 125–131, Jun. 2022, doi: https://www.doi.org/10.52825/scp.v2i.107.
13. “HSL Helmet model”, https://github.com/HSLdevcom/helmet-ui (04.04.2023).
14. “Digitraffic data”, https://www.digitraffic.fi/en/road-traffic/lam/ (04.04.2023).
15. McNally M. G., The four-step model. Emerald Group Publishing Limited, 2007.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.199

Published: 29 June 2023

Development, calibration, and validation of a large-scale traffic simulation model: Belgium road network

Behzad Bamdad Mehrabani[1][https://orcid.org/0000-0001-8585-7879], Luca Sgambi[1][https://orcid.org/0000-0002-4132-2087], Sven Maerivoet[2][https://orcid.org/0009-0000-1040-4525] and Maaike Snelder[3,4][https://orcid.org/0000-0001-7766-2174]

[1] Université Catholique de Louvain, Louvain Research Institute for Landscape, Architecture, Built Environment (LAB), Louvain-la-Neuve, Belgium

[2] Transport & Mobility Leuven, Leuven, Belgium

[3] Delft University of Technology, Civil Engineering Faculty, Delft, Netherlands

[4] Netherlands Organization for Applied Scientific Research (TNO), Hague, Netherlands

Abstract. Development of large-scale traffic simulation models have always been challenging for transportation researchers. One of the essential steps in developing traffic simulation models, which needs lots of resources, is travel demand modeling. Therefore, proposing travel demand models that require less data than classical travel demand models is highly important, especially in large-scale networks. This paper first presents a travel demand model named as probabilistic travel demand model, then it reports the process of development, calibration and validation of Belgium traffic simulation model. The probabilistic travel demand model takes cities' population, distances between the cities, yearly vehicle-kilometer traveled, and yearly truck trips as inputs. The extracted origin-destination matrices are imported into the SUMO traffic simulator. Mesoscopic traffic simulation and the dynamic user equilibrium traffic assignment are used to build the base case model. This base case model is calibrated using the traffic count data. Also, the validation of the model is performed by comparing the real (extracted from Google Map API) and simulated travel times between the cities. The validation results ensure that the model is a superior representation of reality with a high level of accuracy. The model will be helpful for road authorities, planners, and decision-makers to test different scenarios, such as the impact of abnormal conditions or the impact of connected and autonomous vehicles on the Belgium road network.

Keywords: Travel demand modelling, Belgium road network, Mesoscopic traffic simulation, SUMO

1. Introduction

Several transportation researchers use traffic simulation models to test different scenarios in different network scales. The inputs of traffic simulation models are usually supply and demand data. Supply data includes all the information related to the transportation network and services, such as the geometric and functional specification of the road network, traffic control; public transport services; and other data, such as fleet vehicles. Also, the travel demand data are typically extracted from travel demand models. The travel demand model is a set of mathematical relationships which describes when, why, and how people and goods move within a particular geographic area. These models estimate travel behavior and demand for a specific (future) time frame, based on several assumptions about the population, land use, household, etc. Travel demand models incorporate economic aspects, technical aspects, lifestyle aspects of society, a specific individual, psychological elements, and factor of time to provide the most accurate representation of a specific travel demand problem (e.g., number of passengers traveling between Brussels to Antwerp by car at P.M. peak hour). Demand data contain the mobility needs of people and goods, which questionnaires, mobile data, etc., can collect.

Several travel demand models have been developed, like 1- Classical four-step model, 2- Tour-based model, 3- Activity-scheduling model, etc. [1]. The most well-known travel demand model is the classical four-step model: 1. Trip generation: determines the number of passengers traveling from a specific city or region. 2. Trip distribution: estimates the number of trips between particular cities or regions. The output of this step is an Origin-Destination (O-D) matrix. The O-D matrix determines the number of trips between each origin and destination. 3. Modal split: determines which transportation mode passengers will use when traveling from their origin to their destination. 4. Traffic assignment: determines the routes passengers choose to reach their destination. For a classical transportation model, the output of the first three steps, an O-D matrix for each transport mode, is fed into a traffic assignment model (either simulation-based or analytical traffic assignment models) to calculate links loads. The final outputs are used to describe, explain, correlate, and forecast transport demand. The four-step travel demand model has been used by many researchers and showed its excellent performance. However, the most critical disadvantage of this model is that it relies heavily upon household surveys and census data that are very costly and time-consuming to collect. Thus, the availability of household data is very challenging in implementing a four-step model, especially for large-scale road networks (e.g., Belgium). In this study, an alternative approach to four-step modeling is proposed to develop a traffic simulation model for Belgium road network. First, a travel demand model named as probabilistic travel demand model is proposed. This travel demand model only considers the population, distances, passenger-kilometer traveled, and the number of yearly truck trips as the input data to provide hourly O-D matrices on a country level (Belgium). The reason for developing such a travel demand model is that it needs less data than classical four-step modeling. Then, the extracted O-D matrices are inputted into the traffic simulation model. This traffic simulation model is calibrated and validated by traffic count and travel time data.

The following section (section 2) gives information about the case study (available supply and demand data). Section 3 explains the methodology, including the development of the probabilistic travel demand model and building the base case model. Then, the process of calibration and validation of the model is provided in section 4. Finally, sections 5 and 6 describe the results and the paper's conclusion.

2. Case Study

Belgium is a European country with a land area of 30,688 km² and a population of 11.5 million [2]. Belgium is divided into three regions: the Flanders in the north, the Wallonia in the south, and the Brussels-Capital Region. The transport network in Belgium, including road, rail, sea, and air, is well-developed and well-connected to other parts of Europe. This transport network includes 13.2 thousand kilometers of main/national roads; 5 international airports; 3,602 kilometers of usable rail network; and five seaports [3], [4]. Belgium plays a crucial role in road travel in Europe and ranks 7 in terms of passenger-kilometer among European Union countries. Also, Belgium's motorway network is the third dense after the Netherlands and Luxembourg in Europe [5]. There are more than eight international E-roads in Belgium which connect the east of Europe to the west and south of Europe to the north. In Belgium, three mobility surveys have been carried out by SPF Mobilité et Transports to examine mobility and road safety patterns in detail, using both household and individual data. These surveys, named MOBEL (1999), BELDAM (2012), and MONITOR (2018), provide a comprehensive understanding of the subject [6]. As far as the authors are aware, the origin-destination matrix from these studies is not accessible to the public

2.1 Supply Data: Belgium Road Network

The Open Street Map (OSM) file was extracted to build the network. The OSM is a free editable map of the whole world that users make. The road network file is directly imported into traffic simulation software (SUMO). The OSM file contains all categories of roads (named Motorway, Trunk, and Primary roads in OSM). Roads in Belgium are categorized into three types, which are 1- Highways ("Autoroute") (e.g., A2 (E314)); 2- Provincial and regional roads ("Routes provinciales et régionales") (e.g., A501); 3- Municipal roads ("Routes communales"). Since this study provides a travel demand model for outer-city trips, only highways and provincial and regional roads (classified as Motorway, Trunk, and Primary roads on OSM) are modeled, and inner-city traffic roads are not modeled. The network is checked and fixed manually for any error in the SUMO environment (using the SUMO network warning and error tool). Cities are considered as centroids that can generate and attract trips. In total, 60 cities are modeled. The criteria for selecting cities are described in the next section. The OSM file already included road network features like speed and capacity. However, they were double-checked with Google Maps data to ensure accuracy. **Figure 1** shows the inserted network into SUMO.

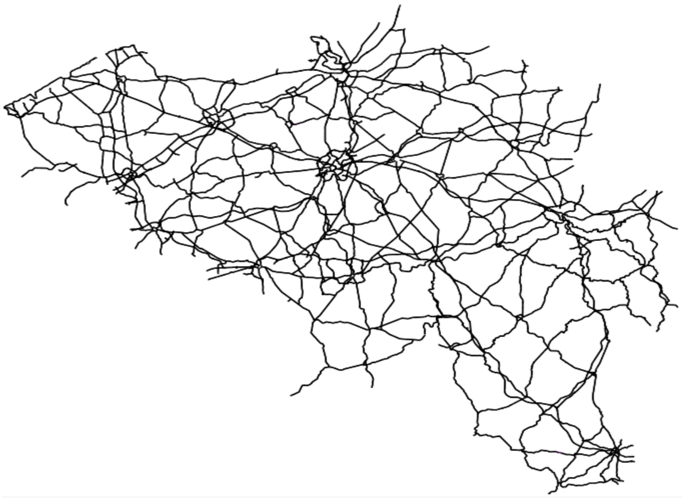


Figure 1. Belgium road network in SUMO

Also, to ensure that the network's geometry and the highways' length are imported to SUMO correctly, the shortest distance between the cities in the simulation is compared with reality. The accurate shortest distances are taken from Google API. A comparison between the shortest distances in reality and the simulated network is given in **Figure 2**. The shortest distances between 3600 O-D pairs in reality and simulation are compared. This figure shows that the length of highways and the geometry of the network is simulated with a high accuracy.

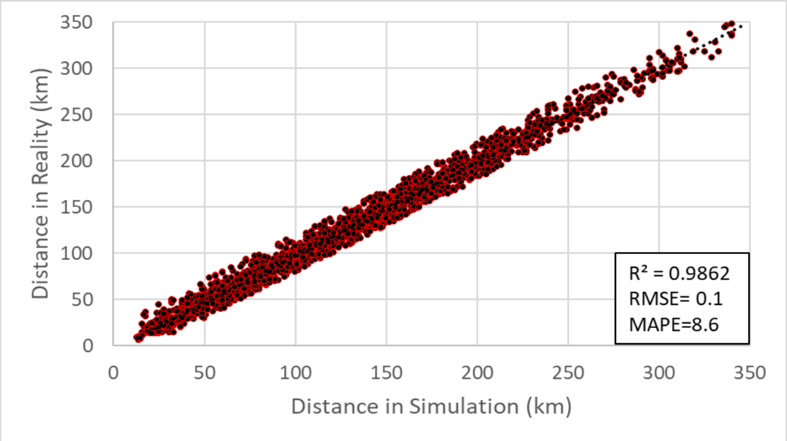


Figure 2. Comparison of distances between cities in simulation and reality

2.2 Demand Data

The data considered as inputs to the probabilistic travel demand model for passenger car trips include the population of each city, the distance between cities, and the passenger-kilometer traveled by the Belgium population each year. Population data for all Belgian municipalities are extracted from STATBEL [7]. STATBEl is the Belgian statistical office that collects, produces, and disseminates reliable and relevant figures on the Belgian economy, society, and territory. Belgium has three regions and 11 provinces. The provinces are subdivided into 43 administrative arrondissements and 581 municipalities, 83 of which have a population of more than 30,000. A list of Belgium's arrondissements with their municipalities was provided. Then, 60 municipalities (cities) were chosen as centroids in the travel demand model based on the following criteria:

A. All municipalities with over 30,000 population were first selected as centroid (to address the effects of most populated cities). Then, if the distance between two municipalities (within the same arrondissement) is less than 15 km, two municipalities are combined and considered one centroid.
B. If an arrondissement consists of less than four municipalities, the first populous municipality is selected (even if the population is less than 30,000).
C. If the arrondissement consists of more than five cities, at least three are included.

The geographic distance between cities is also calculated based on the longitude and latitude of cities. The passenger-kilometer travel data for passenger cars is extracted from Federal Planning Bureau (FPB) website [8]. This independent public agency draws up studies and projections on economic, social, and environmental policy issues. According to **Table 1**, the total passenger-kilometer traveled per year for all three regions of Belgium in highways, provincial and regional roads are equal to 85.762×10^9.

Table 1: Passenger kilometers traveled per year in Belgium (million passenger-kilometer)

	Wallonia	Flanders	Brussels-Capital
Highways	16373.6	23676.5	520.7
Provincial and regional roads	18286.0	25058.0	1848.1
Municipal roads	9456.2	10662.2	1059.1

Also, for modeling the freight transport trips (trucks), the number of yearly trips by Belgian trucks (by country of loading and unloading) is extracted from STATBEL [9]. The number of truck trips loaded in Belgium and unloaded in Belgium is given in **Table 2**.

Table 2: Number of yearly Trucks trips in Belgium

	Loaded in Belgium	Unloaded in Belgium	Total
Number of Yearly trucks trips	16173401	15829069	32002470

3. Methodology

The overall process of the traffic simulation modeling for the Belgium road network is illustrated in Figure 3. Each step is described in as follow.

3.1 Probabilistic Travel Demand Model

The probabilistic travel demand model first determines the total number of daily trips based on the passenger-kilometer traveled data. Then, it indicates the origin and destination of each trip by applying a random selection on the weighted distribution function of population and distances between the cities [10], [11]. The core assumption behind the model is that the larger a city's population, the greater its likelihood of being chosen as the origin city of a trip. Also, more population of a city and lower distance between cities increase its likelihood of being selected as the city of destination. This model's principles are similar to the gravity model [12]. The steps of the probabilistic travel demand model are described in the following subsections.

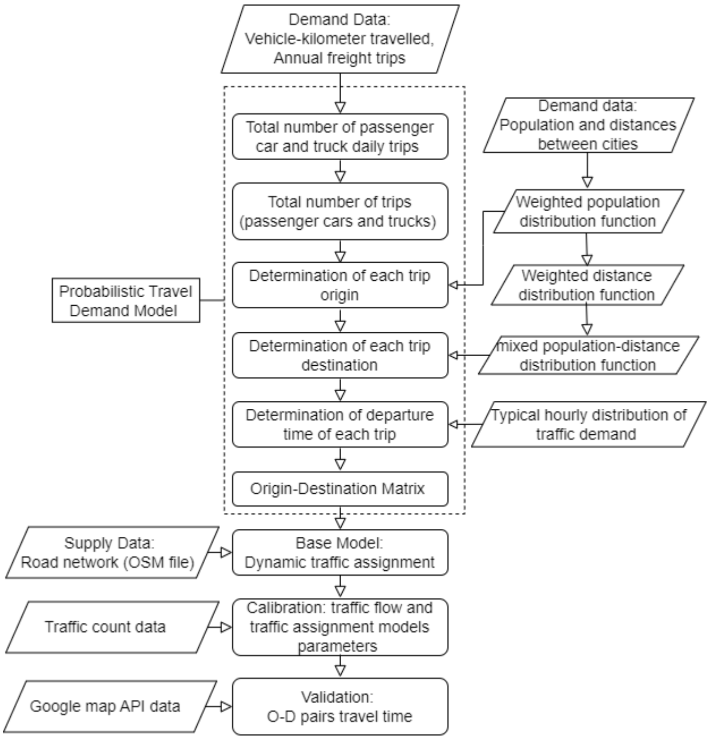

Demand Data: Vehicle-kilometer travelled, Annual freight trips
Total number of passenger car and truck daily trips
Total number of trips (passenger cars and trucks)
Determination of each trip origin
Determination of each trip destination
Determination of departure time of each trip
Origin-Destination Matrix
Probabilistic Travel Demand Model
Demand data: Population and distances between cities
Weighted population distribution function
Weighted distance distribution function
mixed population-distance distribution function
Typical hourly distribution of traffic demand
Supply Data: Road network (OSM file)
Base Model: Dynamic traffic assignment
Traffic count data
Calibration: traffic flow and traffic assignment models parameters
Google map API data
Validation: O-D pairs travel time

Figure 3: Overall process of the traffic simulation modeling for Belgium

3.1 Probabilistic Travel Demand Model

The probabilistic travel demand model first determines the total number of daily trips based on the passenger-kilometer traveled data. Then, it indicates the origin and destination of each trip by applying a random selection on the weighted distribution function of population and distances between the cities [10], [11]. The core assumption behind the model is that the larger a city's population, the greater its likelihood of being chosen as the origin city of a trip. Also, more population of a city and lower distance between cities increase its likelihood of being selected as the city of destination. This model's principles are similar to the gravity model [12]. The steps of the probabilistic travel demand model are described in the following sections.

3.1.1 Determination of the total number of passenger cars daily trips

As shown in **Table 1**, the total number of passenger-kilometer traveled on highways, and the provincial and regional road is equal to 85.762×10^9. Assuming that each passenger travels 85 kilometers per trip and the weekend traffic is considered to be half as high, this gives approximately 3.2 million trips per day in the Belgium road network. The equation for calculating the total number of passenger cars daily trips (T_{PC}) is as follows:

$$T_{PC} = \frac{passenger\ kilometer\ travelled\ (yearly)}{days\ in\ each\ year \times day\ weight \times average\ length\ of\ trips} (1)$$

3.1.2 Determination of the total number of trucks daily trips

Table 2 shows the total number of Belgian trucks' yearly trips. To calculate the number of daily truck trips, equation 2 is used. Similar to passenger car trips, it is assumed that the number of weekend trips is half the weekday trips. Equation 2 gives 102572 trucks' trips per day.

$$T_{TR} = \frac{number\ of\ trucks'\ trips\ (yearly)}{days\ in\ each\ year \times day\ weight} (2)$$

By summing up the passenger car trips and truck trips, the total number of all trips can be calculated.

$$T_T\ (t \in T_T) = 3.2 + 0.1 = 3.3\ million\ trips\ (3)$$

3.1.3 Determination of trips Origins

After determining the total number of daily trips in Belgium (both passenger cars and trucks) ($T_T = 3.3$), the origin for each trip (t) should be specified. It is assumed that the probability of selecting a city as the origin city is proportional to its population. A random selection is applied to the weighted probability distribution of cities' populations. The weighted probability distribution function is given in equation 4. The logic behind this selection is that a city with a larger population is more likely to be chosen as the origin city.

$$f_{PO}(x_i) = p_i(X = x_i) = \frac{Po_i}{\sum_{i=1}^{60} Po_i}\ (4)$$

$$0 < f_{PO}(x_i) < 1;\ \sum_{i=1}^{60} f_P(x_i) = 1$$

Where $f_{PO}(x_i)$ is the weighted population probability distribution function; $p_i(X = x_i)$ is the probability of city i to be chosen as the origin city for trip t; and Po_i is the population of city i.

3.1.4. Determination of trips Destinations

In this step, the destination for each trip is determined. The decision for selecting the city of destination is based on the assumption that the larger the population of a city and the closer it is to the city of origin, the higher the likelihood of it being chosen as the destination city. To put this assumption into mathematical form, the weighted distance distribution function is defined for each origin city i as follows:

$$f_D(x_{ij}) = p(X = x_j | x_i) = \frac{\frac{1}{(Dis_{ij})^3}}{\sum_{j=1}^{j=60} \frac{1}{(Dis_{ij})^3}}\ (5)$$

$$0 < f_D(x_{ij}) < 1;\ \sum_{j=1}^{60} f_D(x_{ij}) = 1$$

Where $f_D(x_{ij})$ is the weighted distance distribution function; $p(X = x_j | x_i)$ is the probability of selecting city j as the destination if city i is chosen as origin city; and Dis_{ij} is the distance between cities i and j. Then, the population distribution and the distance distribution are mixed to generate the new distribution. The mixed population-distance probability distribution function is defined as:

$$f_M(x_{ij}) = p(X = x_j | x_i) = \lambda \times f_{PO}(x_i) + (1 - \lambda) \times f_D(x_{ij})\ \ (6)$$

Where $f_M(x_{ij})$ is the mixed population-distance distribution function; $p_{ij}(X_{ij} = x_{ij} | x_i)$ is the probability of selecting city j as destination if city i was selected as origin; $f_{PO}(x_i)$ is the weighted population probability distribution function; $f_D(x_{ij})$ is the weighted distance probability distribution function; λ is a calibration parameter that indicates the importance of considering the population or distance between the cities in the probability of being selected as the destination city. The destination city of trip t is determined by applying a random selection on the mixed population-distance distribution function mentioned above.

It's worth noting that in this study, the behavior of both passenger cars and trucks trips are assumed to be the same in terms of origin and destination. However, the nature of truck trips may vary based on the locations of terminals, distribution centers, and companies within different sectors, and further research is necessary to investigate these differences.

3.1.5 Determination of departure times

Each trip t is assigned to each hour of the day by the typical hourly distribution of travel (based on the type of vehicles: passenger car or truck), which is presented by NCHRP (2004) (**Figure 4**).

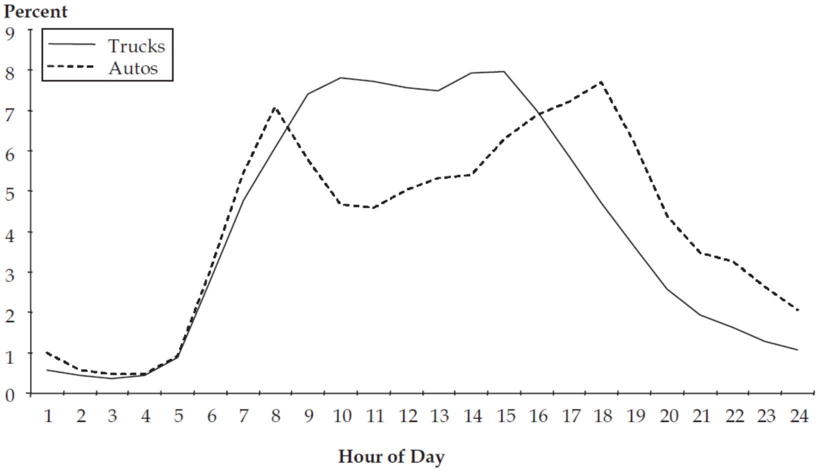


Figure 4: Typical hourly Distribution of Traffic Demand [13]

At the end of step 5 of this probabilistic travel demand model, the hourly O-D matrix for each hour of the day is available. The Pseudocode of the model is given in **Table 3**.

Table 3: Pseudocode code of probabilistic travel demand model

```
Probabilistic Travel Demand Model
Input: road network, yearly passenger-kilometer traveled, yearly trucks trips
Output: hourly O-D matrix
Initialization: set the total number of daily trips $T_T = 3.3 \times 10^6$ and trip index $t = 1$
Main loop:
For trip $t$ in the range [1, $T_T$]:
    i.   Form the weighted population probability distribution function ($f_{PO}(x_i)$)
    ii.  Apply a random selection on $f_{PO}(x_i)$ to determine the city of origin for trip $t$.
    iii. Form the weighted distance function ($f_D(x_{ij})$) for the origin city.
    iv.  Form the mixed population-distance function ($f_M(x_{ij})$) for the origin city.
    v.   Apply a random selection on $f_M(x_{ij})$ to determine the city of destination for the trip $t$.
    vi.  Apply a random selection on the typical hourly traffic demand distribution (based on the type of vehicle) to determine the departure time of trip $t$.
    vii. $t = t + 1$
End of loop
```

3.2 Base Case Model

After obtaining the hourly O-D matrix, a base model was simulated. Dynamic User Equilibrium (DUE) traffic assignment was used to assign travel demand to the network. The traffic assignment tool in SUMO, dualterate.py, is used to perform DUE. Please refer to [14], [15] for more information on dynamic traffic assignment in SUMO. Dijkstra's algorithm is used to find the shortest path. The Logit model is used as the route choice model. The simulation is performed at the mesoscopic level, distinguishing between passenger cars and trucks.

It should be noted that in this basic model, all of the traffic flow model's parameters and the traffic assignment model's parameters are considered the default values of SUMO. Then, they are modified based on the calibration process explained in the next section. The simulation period was 24 hours; however, this paper reports on calibration and validation for the morning peak.

4. Calibration and Validation of the Model

Model calibration is the process of variating model parameters in a way that the system performance of the model meets real data output. This is the most critical and complex step of traffic simulation. Previous studies have suggested that two types of parameters should be calibrated in traffic simulation [16]–[18]:

1. Calibration of traffic flow model parameters (capacity calibration): This type of calibration consists of local and global parameter modifications and tries to reproduce observed traffic capacities in the field by modifying traffic flow model parameters (reaction time, headway, etc.).
2. Calibration of dynamic traffic assignment (global or local parameters) model parameter: This calibration is intended to make the path selection of vehicles in simulation close to reality. Usually, it is done by comparing the real and simulated traffic counts on specific links. To make the simulated traffic count closer to reality, either the inserted O-D matrix is modified, or the traffic assignment model is modified (by changing the assignment method, route choice model parameters, number of iterations, etc.).

4.1 Calibration of traffic flow model parameters

The traffic flow modeling is performed on the mesoscopic scale in SUMO. The mesoscopic model of SUMO is based on the work of Eissfeldt [19]. This model is a queue base model which computes the time at which a vehicle travels from a queue based on the traffic state in the current and subsequent queue, the minimum travel time, and the stage of intersection (e.g., red, green, yellow). Some examples of mesoscopic parameters are minimum headway, queue length, junction control, edge length, etc. This model's parameters are calibrated for large-scale networks in the work of Presinger [20]. This study uses the same parameters of the queuing model as the work of Presinger. Please refer to (DLR, 2021; Presinger, 2021) for more information about queuing model parameters.

4.2 Calibration of dynamic traffic assignment model parameters

The dynamic traffic assignment model was calibrated based on a comparison of real count data and the model-assigned count data for 50 detectors on the network. The segments are selected in a way that covers the entire Belgian network. The traffic count data was extracted from the website of the Flanders government [21]. This calibration consists of two parts. First, the O-D matrix is adjusted by testing the different values of λ. As mentioned in previous sections, λ is a calibration parameter in the travel demand model. It determines the importance of a city's population and its distance from the city of origin in determining each trip's destination. After testing several values of λ, it was finally concluded that $\lambda = 0.25$ leads to an O-D matrix with the closest simulated traffic count to the real traffic count. The second part is to modify parameters of DUE. The models and parameters considered in the calibration process are warm-up time, routing algorithm, route choice model (e.g., deterministic or stochastic), route choice model parameters, number of available alternatives, swapping algorithm, and number of iterations. By altering these parameters, it was found that the Dijkstra routing algorithm with the Logit route choice model and ten iterations gives the best results.

Figure 5 shows the hourly simulation results versus the observed flows. In addition to the scatter plot, the GEH criterion is calculated to compare the simulated and real traffic volumes. GEH determines the tolerance of relative and absolute errors on the network's traffic count. GEH formula is as follows:

$$GEH = \sqrt{\frac{2(M-C)^2}{M+C}} \quad (7)$$

M is the simulation's hourly traffic volume, and C is the real-world hourly traffic count. The average GEH is equal to 4.9. In this study, 78% of observations have GEH criteria less than 7.5, which is in the acceptable threshold [16].

4.3 Validation

Various methods exist to validate traffic simulation models, including traffic count and travel time comparison between reality and simulation. One of the new methods of large-scale traffic simulation models' validation, which has been used in previous studies [22], is the comparison of real and simulated travel times at the origin-destination pair level. This method extracts the simulated travel time between each origin and destination (cities) from DUE (for a specific time interval). The real travel times are exported from Google Maps' Distance Matrix API. The Distance Matrix API provides travel distance and time for a matrix of origins and destinations [23]. In this study, this method is implemented for validation.

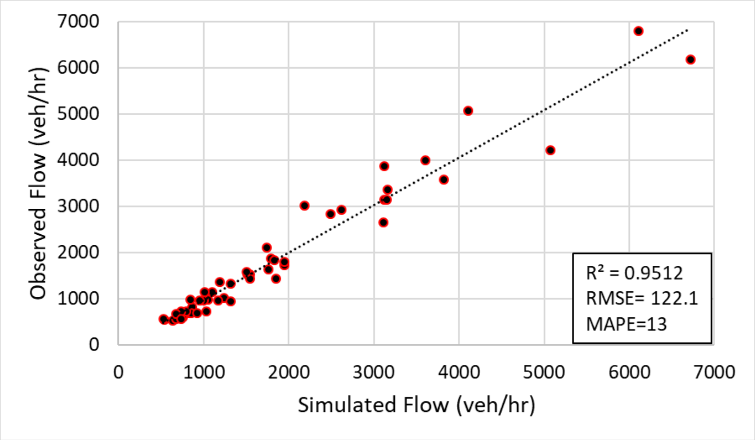


Figure 5: Observed and simulated hourly traffic volumes

To validate the traffic simulation model, the O-D pairs' travel times are compared in reality and simulation in A.M. peak hour conditions. Figure 6 compares simulated and real travel times in A.M. peak hours. Figures 6 consists of 3600 (60×60) O-D pair travel times throughout the network and shows that the simulation model is a good representation of reality with a high level of accuracy $(R^2 = 0.93)$ in congested conditions. The closeness of the travel times in reality and the simulation in a congested state can indicate that the traffic counts in the simulation are near the real traffic count. Since the travel time between O-D pairs depends on the number of vehicles on the path.

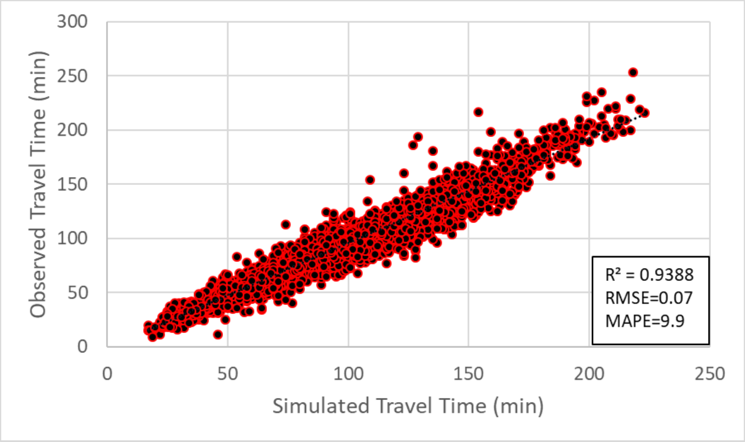


Figure 6: Observed path travel times obtained from Google vs. simulated path travel times extracted from DUE

5. Calibrated and Validated Model Results

The findings of the calibrated and validated traffic simulation model for Belgium during a A.M. peak hour are displayed in Table 4. This table reveals that the overall travel time was 338896 hours and the average speed for all vehicles was 61 km/hr. Additionally, Figure 7 presents a comparison between the simulated and actual speeds on the roads in Belgium. As demonstrated in the figure, the model correctly identifies the locations of traffic congestion.

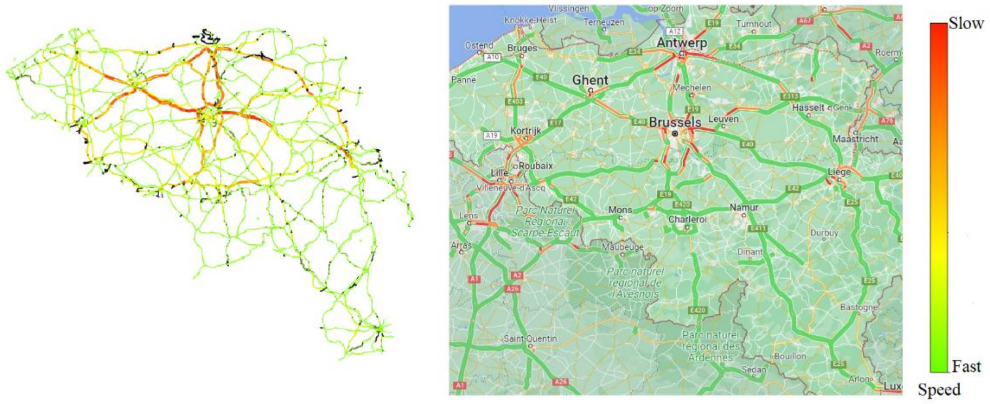
Table 4: Calibrated and Validated traffic simulation model results

Traffic assignment	Total travel time (hr)	Average speed (km/hr)	Average travel time (min)
DUE	338896	61	79.2

6. Conclusion

This study reports the development, calibration, and validation of the traffic simulation model of Belgium. First, a probabilistic travel demand model is developed using population, distances between the cities, yearly vehicle-kilometer travelled by passenger cars, and annual truck trips. The probabilistic travel demand model calculates the number of all trips based on the yearly vehicle-kilometer travelled and the trucks' trips. The origin and destination of each trip are determined by applying random selection on the population distribution and the mixed distribution of population and the distance between the cities, respectively. The departure time of each trip is based on the typical distribution of travel demand. Then the probabilistic travel demand model's outputs (hourly O-D matrices) are imported to SUMO's traffic simulation software. After that, a base model is simulated using the mesoscopic feature of the SUMO traffic simulator and the DUE traffic assignment to assign travel demand to the network. This basic model is calibrated by real traffic count data. The calibration process includes the calibration of traffic flow model parameters (queuing model) and the parameters of dynamic traffic assignment. Finally, the model is validated using real travel times between cities in congested conditions. The real travel times are extracted from Google map Distance Matrix API. The results of the validation prove the accurate performance of the traffic simulation model.

The proposed traffic simulation model of Belgium can help researchers, decision-makers, and policy-makers, to test different transportation planning scenarios at the country level. For future studies, developing the proposed probabilistic travel demand model for other case studies is recommended to check the model's performance. Also, in this study, for modeling the freight demand, only trucks with Belgian license plates were taken into consideration. However, for achieving highly precise outcomes, it's imperative to factor in the transit traffic of cargo vehicles from other nations as well. This task, though, requires access to cargo data from those other countries.



(a) Simulated typical speed in peak hour

(b) Real typical speed in peak hour

Figure 7: Simulated and real links speed of the Belgium road network

Data availability statement

The data that support the findings of this study are not publicly available. Access to the data may be granted upon request to the corresponding author (behzad.bamdad@uclouvain.be).

Author contributions

Behzad Bamdad Mehrabani contributed to the conceptualization of the study, developed the methodology, conducted the investigation, curated the data, wrote the original draft, and created the visualizations. Luca Sgambi contributed to the conceptualization of the study, validated the results, reviewed and edited the writing, provided supervision, managed the project administration, and acquired the funding. Sven Maerivoet assisted with the methodology, validated the results, and reviewed and edited the writing. Maaike Snelder contributed to the conceptualization of the study, developed the methodology, validated the results, analyzed and interpreted the data, and reviewed and edited the writing. All authors reviewed the results and approved the final version of the manuscript.

Competing interests

The authors declare that they have no competing interests.

Funding

The Université Catholique de Louvain supported the corresponding author under the "Fonds Speciaux de Recherche" and the "Erasmus +" programs.

Acknowledgement

The corresponding author would like to express his gratitude to UC Louvain for providing the funding which made this research project possible.

References

[1] E. J. Miller, "Travel demand models, the next generation: Boldly going where no-one has gone before," in *Mapping the Travel Behavior Genome*, Elsevier, 2020, pp. 29–46.

[2] OECD, "Population (indicator)," 2022. .

[3] Statista, "Total length of the road network in Belgium," 2009. https://www.statista.com/statistics/449794/belgium-length-of-road-network-by-road-type/ (accessed Nov. 21, 2022).

[4] Statista, "Total length of the railway lines in Italy from 2011 to 2018.," 2020. https://www.statista.com/statistics/450034/length-of-railway-lines-in-use-in-belgium/ (accessed Nov. 21, 2022).

[5] W. Decoster, L. Van Elsen, P. De Splenter, and A. Van Snick, "BELGIAN TRANSPORT & LOGISTICS," 2020.

[6] SPF Mobilité et Transport, "Premiers résultats de l'enquête Monitor sur la mobilité des Belges," *2019*. Bruxelles, pp. 1–6, 2018.

[7] STATBEL, "Structure of the Population | Statbel," *Structure of the Population - Statbel - Belgium in figures*, 2022. https://statbel.fgov.be/en/themes/population/structure-population#panel-14 (accessed Nov. 21, 2022).

[8] FPB, "Base de données transport," 2017. https://www.plan.be/databases/PVarModal.php?VC=TTBE_PX_RD_PKM&D1[]=EU15_BE1&D1[]=EU15_BE2&D1[]=EU15_BE3&D2[]=W50PRIVATE&D3[]=WW10SNEL&D3[]=WW20GEN&D3[]=WW30GEM&lang=fr&DB=TRANSP (accessed Dec. 20, 2022).

[9] STATBEL, "Road freight transport," *Road freight transport*, 2022. https://statbel.fgov.be/en/themes/mobility/transport/road-freight-transport#figures (accessed Nov. 26, 2022).

[10] L. Sgambi, T. Jacquin, N. Basso, and E. Garavaglia, "The robustness of infrastructure network assessed through a probabilistic flow model and a static traffic assignment algorithm–the case of the Belgian road network," in *IABMAS2020 10th Int. Conf. on Bridge Maintenance, Safety and Management*, 2021, pp. 1–6.

[11] T. Jacquin, "Modélisation temporelle du trafic pour des études de résilience sur le réseau routier belge," Université catholique de Louvain, 2019.

[12] J. de D. Ortúzar and L. G. Willumsen, *Modelling Transport*. John wiley & sons, 2011.

[13] NCHRP, *Traffic Data Collection, Analysis, and Forecasting for Mechanistic Pavement Design*, vol. 538. Transportation Research Board, 2004.

[14] B. Bamdad Mehrabani, J. Erdmann, L. Sgambi, and M. Snelder, "Proposing a Simulation-Based Dynamic System Optimal Traffic Assignment Algorithm for SUMO: An Approximation of Marginal Travel Time," in *SUMO Conference Proceedings*, 2022, vol. 3, pp. 121–143, doi: 10.52825/scp.v3i.119.

[15] DLR, "SUMO User Documentation," 2021. https://sumo.dlr.de/docs/index.html.

[16] M. Aghababaei, S. Costello, and P. Ranjitkar, *South Island Model: development and calibration*. 2019.

[17] J. Barceló, *Fundamentals of traffic simulation*, vol. 145. Springer, 2010.

[18] J. Casas, J. L. Ferrer, D. Garcia, J. Perarnau, and A. Torday, “Traffic simulation with aimsun,” in *Fundamentals of traffic simulation*, Springer, 2010, pp. 173–232.

[19] N. G. Eissfeldt, “Vehicle-based modelling of traffic. Theory and application to environmental impact modelling,” *University of Cologne*. Universität zu Köln, p. 199, 2004, [Online]. Available: https://kups.ub.uni-koeln.de/1274/.

[20] D.-I. C. Presinger, “Calibration and Validation of Mesoscopic Traffic Flow Simulation.” Graz University of Technology, 2021.

[21] Vlaanderen, “Verkeersindicatoren,” 2022. http://indicatoren.verkeerscentrum.be/vc.indicators.web.gui/indicator/index.

[22] S. Shafiei, Z. Gu, and M. Saberi, “Calibration and validation of a simulation-based dynamic traffic assignment model for a large-scale congested network,” *Simul. Model. Pract. Theory*, vol. 86, pp. 169–186, 2018, doi: 10.1016/j.simpat.2018.04.006.

[23] Google LLC, “Distance Matrix API: Developer Guide,” *Google Maps Platform*, 2017. https://developers.google.com/maps/documentation/distance-matrix.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.211

Published: 29 June 2023

SUMO Roundabout Simulation with Human in the Loop

Giorgio Previati[1] [https://orcid.org/0000-0001-6450-1566], and Gianpiero Mastinu[1] [https://orcid.org/0000-0001-5601-9059]

[1] Politecnico di Milano, Milan, IT

Abstract: Traffic simulators rely on calibrated driver models in order to reproduce human behavior in different traffic scenarios. Even if quite accurate results can be obtained, the actual interaction between human being and traffic cannot be completely reproduced. In particular, as automated vehicles are being developed, the human in the loop is required to understand whether drivers feel comfortable and safe in mixed traffic conditions. In recent years, dynamic driving simulators have been developed to test vehicles in complex or dangerous situations in safe and controlled environments. However, driving simulators are mostly devoted to the study of vehicle dynamics more than traffic situations.

This paper presents an integration of SUMO with a high end dynamic driving simulator with the aim to study human reactions while negotiating a roundabout in mixed traffic conditions. SUMO is in charge of traffic simulation, while a full vehicle model is employed for the simulation of the dynamic of the human driven car. To allow a human to effectively drive the car, both simulation environments have to run in real time while exchanging the required information. Also, scenario graphics, sound and driving simulator feedback motion have to be accurately realized and synchronized with the simulations. A real-time server is employed for the synchronization of the different environments. As SUMO does not consider vehicle dynamics, particular attention is devoted to the a realistic reconstruction of trajectories and vehicle dynamics to be represented in the scenario.

Some preliminary tests are shown where a panel of testers has been asked to negotiate the roundabout with different percentages of automated vehicles. The results of the tests show that drivers were able to perceive differences in the behavior of other vehicles and that the proposed approach is effective for understanding the feeling of comfort and safety of the human driver.

Keywords: SUMO cosimulation, Human in the loop, Driving simulator, Autonomous and connected vehicles

1 Introduction

Microscopic traffic simulators (MTS) are a powerful tool for the study of traffic and infrastructures. Each vehicle in a given road network is simulated individually allowing, among other things, for a detailed analysis of infrastructure design and modification, traffic control, behavioral studies and testing of connected and automated vehicles

(CAV) [1]. CAVs are expected to provide a huge opportunity for increasing traffic flow, increase transport safety and reduce fuel consumption and emissions [2], [3]. However, the introduction of CAVs requires a huge amount of tests and data collection both by real world tests and by simulations [4]. CAVs and human driven vehicles will be required to share the same traffic environment during the transition period [5]. Reliable simulations in mixed traffic conditions require a realistic model of the human driver. To this respect, even if MTS provide several models describing driver behavior, due to human complexity and variability, such models are not able to fully catch human effects in traffic simulations [1]. Also, human acceptance and preferences when driving in a mixed traffic condition is still being researched [6].

Driving simulators, on the other hand, are developed with the aim to introduce the human driver into the simulation and study its interaction with the simulated vehicle and environment. In general, driving simulators are more focused on the simulation of the driven vehicle and the surrounding traffic simulation can be not fully accurate [5]. Those simulators employ sophisticated 3D visualization creating realistic and immersive scenarios. In cases, driving simulators can be coupled with actuated platforms and give also motion feedback to the driver. Such dynamic driving simulators often feature full scale vehicle cockpit and audio surrounding to create a completely immersive experience in order to get a more natural response of the human driver.

Several papers can be found in the literature describing the integration of MTS and driving simulators. Even if such integration is not a novelty [7], [8], only in recent years graphical and computational performances have allowed the realization of realistic scenarios [1], [4]–[6], [9]–[14]. From these papers, the principal technical challenges related to the co-simulation between MTS and driving simulators can be summarized in the following aspects.

- *Network correspondence*. To obtain a proper co-simulation, the same road network must be reproduced both in the MTS and in the driving simulator. This problem is discussed in the great majority of the cited papers. Depending on the chosen software for the simulations, different approaches, mostly manual, are described.
- *Trajectories*. MTS do not consider a realistic vehicle dynamics, but the simulation is focused only on traffic flow. As a result, the trajectories of the vehicles are not realistic, but unrealistic effects such us sharp turning angles or instantaneous line changes are usually present. In [1] interpolation schemes are proposed in order to obtain smooth bending trajectories.
- *Synchronization and real time simulation*. In order to include a human in the loop, the simulation must run in (or close to) real time. Also, the simulation time of the two software must be synchronized and frequent exchange of information has to take place. Different strategies are presented. In [1] the built-in real time function of the employed MTS has been exploited to trigger the simulations. Alternatively, when a different MTS without such function has been used, the integration parameters have been set to obtain a similar effect. In [13], a dynamic driving simulator has been employed and used to synchronize the simulations.
- *Delay*. Delay between the two simulations is very important to provide a realistic and consistent experience to the human driver. Especially in urban scenarios, delay is very important to allow a correct perception of the positions of the other vehicles. Delay depends on the rate of data transfer between the two simulators. Usually, driver simulators run with very short simulation steps (from a maximum of 33 ms [11] up to 1 ms [15]). Smaller time steps have to be preferred to enhance

the experience felt by the driver. MTS have larger integration steps, with common values between 0.1 to 1s, with the lower value typically used for these applications. This causes large delays (of the order of 0.1-0.2 s [1], [11], [13]) between the two simulations which may alter the driver perception.

In this paper, a co-simulation between SUMO [16] and a high end dynamic driving simulator is developed for the simulation of mixed traffic conditions with CAVs and human driven vehicles in a roundabout scenario. With respect to the considered papers, the described application employs a real time scheduler to have a very accurate synchronization between the two simulations and the real-time. Also, by setting up a communication frequency of 200Hz between the simulation, a very short delay of 5 ms is obtained. The employment of a high end dynamic simulator allows the driver to have a fully immersive experience, including the motion feedback. Additionally, a model of reinforcement learning artificial intelligence is run in parallel to the SUMO simulation by using the Flow library [17] to drive the CAVs. Preliminary tests with a restricted panel of drivers show the potentialities of the application.

2 Driving simulator and VI-Worldsim environment

The dynamic driving simulator utilized is the cable-driven DiM400 Dynamic Driving Simulator of the DRISMI laboratory [18] of Politecnico di Milano. The simulator is produced by VI-grade [19] and shown in Figure 1. The driving simulator features a full size vehicle cockpit (Figure 1 right)

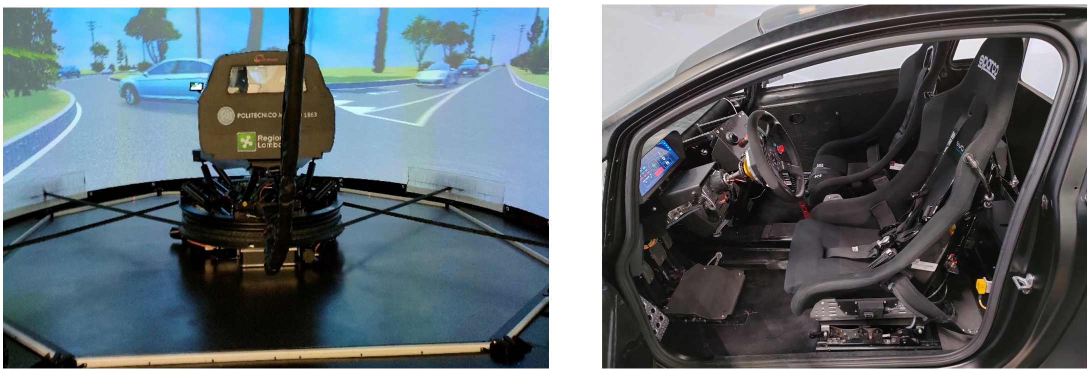


Figure 1. Driving simulator DIM400 at Politecnico di Milano inside the DRISMI lab (www.drismi.polimi.it). On the right, detail of the cockpit interior.

The cockpit motion is obtained by a redundant system of actuators, conceived to decouple the low-frequency and high-frequency motions. A lower stage of actuation composed by a cable driven platform with in-plane degrees of freedom (longitudinal, lateral and yaw) is coupled by a higher stage realized by a Stewart platform providing all six degrees of freedom. The first stages is capable of large motion at relative low frequency (up to 3 HZ), while the second stage realized smaller motions at higher frequencies (up to 30 Hz). By combining the two stages, both low and high vehicle frequencies can be reproduced. To reproduce the higher frequencies related to NVH (noise and vibration harshness), eight shakers, able to provide vibrations up to 200 HZ, are located in engine and suspension connecting points. Table 1 reports the driving simulator performances, further details on the driving simulator can be found in [20]. Haptic seat belts, air cushions, interactive steering wheel and active brake complete the cockpit equipment. A 270°-wide 120 Hz screen surrounds the cockpit. Five speakers reproduce the sources of noise in and out of the vehicle while driving.

The motion of the dynamic driving simulator is controlled by a cueing algorithm based on a Model Predictive Control and able to provide linear and rotational acceleration consistent with the expected acceleration in the considered situation [21]. The motion of the human driven vehicle in the simulation is computed by a 14-degrees of freedom model implemented in VI-CarRealTime [19].

Table 1. Driving simulator DIM400 specifications.

Physical quantity	Values
Platform size	6m x 6m
Visual system (H)	270^o
Visual system (V)	45^o
Degrees of freedom	9
Longitudinal acceleration	1.5g
Vertical acceleration	2.5g
Lateral acceleration	1.5g
Lateral travel	4.2m
Longitudinal travel	4.2m
Vertical travel	± 298mm
Yaw angle	$\pm\ 62^o$
Pitch angle	$\pm\ 15^o$
Roll angle	$\pm\ 15^o$

The graphical environment and the other vehicles are reproduced by VI-WorldSim [19]. VI-WorlSim (Figure 2) provides a full 3D traffic visualization realized by Unreal Engine. It is a commercial software, ready to use, and it also includes a basic traffic generator. Optionally, the traffic generator can be disabled and the vehicles can be controlled by external signals. VI-WorldSim is installed on a Intel i7-9700K@3.60 GHz workstation with 32 GB and Windows 10 pro.

The driving simulator is controlled by a 2 x Intel Xeon Gold 6144@3.50 GHz with 48 GB and Linux RedHawk 7.3 real time server ([22]). The server is in charge of synchronizing all process, run the VI-CarRealTime simulation of the human driven vehicle, run the cueing algorithm for the control of the simulator and manage all network connections, sensors and cockpit actuators. Graphic and sound are managed by six Intel i7-9700K@3.60 Ghz with 32 GB and Windows 10 pro workstations equipped with a GeForce RTX 2080 Ti. A real time database is updated at each simulation step on the real time server and is shared with all other workstations. The simulation step is set at 1 ms. The real-time server constraints each simulation step to be performed in a time interval of 1 ms, assuring a real-time simulation. All models involved must be optimized to have computational times less than the allotted time interval.

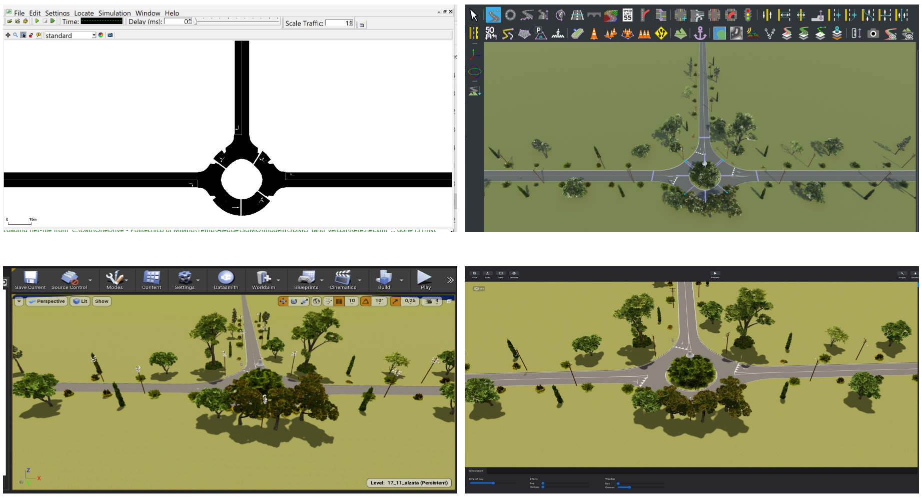
3 Reference scenario

The reference scenario for this application is a three-leg single-line roundabout with mixed traffic conditions. Connected and automated vehicles share the roundabout with human driven vehicles. The human driven vehicles are driven by a IDM (intelligent driver model) algorithm [23] implemented in SUMO. One of the human driven vehicles is actually driven by the human in the loop in the driving simulator. All the vehicles are considered to be connected and exchange data related to their trajectories, velocities and accelerations. CAVs are controlled by artificial intelligence defined by a reinforcement learning approach designed for realizing a policy able both to drive safely CAVs

Figure 2. VI-Worldsim virtual environment, adapted from [19].

into the roundabout and to optimize the traffic flow. The communication protocol based on an innovative V2N2V (Vehicle to NEtwork to Vehicle) approach with 5G communication and edge node computing has been specifically developed in the AI@EDGE project [24]. This reference scenario is part of the AI@EDGE project and represent a use-case for the validation of the project 5G and edge computing technologies. The roundabout scenario has been chosen as roundabouts are currently one of the most critical scenarios for automated driving [25] providing a challenging real world problem to test the technologies developed in the AI@Edge project. For interested readers, more details on the project and on the AI can be found in [24]. In this paper, only the part of the project related to the integration between the two simulators is discussed.

The roundabout network has been realized in SUMO. The network has then been exported to Mathworks Roadrunner and translated by the Unreal engine to be imported in VI-WorldSim. This procedure guarantees the correspondence between the road network in SUMO and in VI-WorldSim. In Figure 3 the road network in SUMO and in Mathworks Roadrunner are depicted.



File Edit Settings Locate Simulation Window Help
Time:
Delay (ms):
Scale Traffic:
standard
Perspective
Lit
Show
Level: 17_11_alzata (Persistent)

Figure 3. Different models of the road network used for the conversion from SUMO to VI-WorldSim. From top left clockwise: SUMO, Mathworks/Roadrunner, Unreal, VI-WorldSim.

4 SUMO - VI-Worldsim integration

The scheme for the integration of SUMO with the driving simulator is depicted in Figure 4. The core of the connection is the real-time database located on the real-time server. The database is accessible by all the workstations of the network. The real-time server provides the base real-time clock of 1 ms to synchronize all connected workstations. In particular, by considering Figure 4, the following processes are considered.

- *Human in the loop vehicle.* The vehicle driven by the human in the loop is simulated by a fourteen degree of freedom model implemented in VI-CarRealTime [19] and running on a dedicated core of the real-time server. The vehicle is simulated with a simulation step of 1 ms. Each simulation step is performed within the allotted real-time clock of 1 ms. The driving commands for the vehicle are given by the human in the loop through the steering wheel and pedals of the cockpit of the simulator. The commands are stored in the real-time database and read by the simulation in VI-CarRealTime. In turns, VI-CarRealTime writes on the real-time database the state of the human driven vehicle. Such states are fed to the controller of the dynamic driving simulator for the motion feedback and to the graphic servers via VI-WorldSim for the visual and audio feedback.
- *SUMO connection.* SUMO runs on a Intel i7-11700F@2.50 GHz with 32Gb and Linux Ubuntu 18.04.6 LTS workstation. The workstation is connected to the real-time server by a UDP connection via a python script. The python script is also in charge of communicating with SUMO by using the TraCI library. The UDP connection is used to synchronize the simulations. The real-time server sends the state of the human driven vehicle every 5 base real-time clocks of 1 ms, i.e at constant time intervals of 5 ms. The python interface waits until the vehicle state is available. When available, it reads the state, updates the vehicle position in the SUMO simulation and then launches a simulation step of 5 ms. When the simulation step is done, the python interface retrieves the states of all other vehicles and sends the information via UDP to the real-time database. It is important to notice that if the computation time required for the SUMO simulation is less than 5 ms, real-time simulation and synchronization are guaranteed. In this way, a delay of only 5 ms is present between the states of the human driven vehicle and the states of the other vehicles. The number of the vehicles that can be simulated in the network without violating the real time constraint depends on the available computational power and the network complexity. With the employed hardware configuration, the scenario considered in this paper can be simulated in real-time with up to 60 vehicles in the network.
- *Artificial intelligence for CAVs control.* A second python instance runs on the same Linux workstation with a second instance of TraCI connected to the same SUMO simulation and to the Flow library. This interface is in charge of communicating with SUMO, retrieve the state of the simulation and provide the commands for controlling the CAVs according to the AI trained by the reinforced learning.
- *VI-WorldSim connection.* VI-WorldSim is connected directly to the real-time database for the standard interactions with VI-CarRealTime to get the motion of the ego vehicle and set the graphical environment accordingly. A second custom connection to the real-time database is established via a Matlab/Simulink interface to provide the motion of the vehicles controlled by SUMO (either driven by a human IDM model or controlled by the IA via TraCI).

The described communication method is designed for the particular configuration of the employed driving simulator. However, the method can be applied to any generic driving simulator program. In fact, the real-time server can be configured to run with most of the most diffused driving simulator programs and graphical environment. Therefore, the general scheme of synchronization and real-time application can be adapted to any software configuration. The advantages of the proposed scheme are a rigorous real-time simulation and a delay between the different simulations of the order of the larger simulation step used (in the described case, 5 ms). Also, as the real-time scheduler can share the real-time database with any number of workstations, more driving simulators can be added to the network allowing for the inclusion of more than one human in the loop. The additional driving simulators can be of any kind (laptop or desktop workstation, static simulators, dynamic simulators) and, in principle, can run different simulation software. Some integration tests have already been run by adding a second driver by connecting a desktop driving simulator.

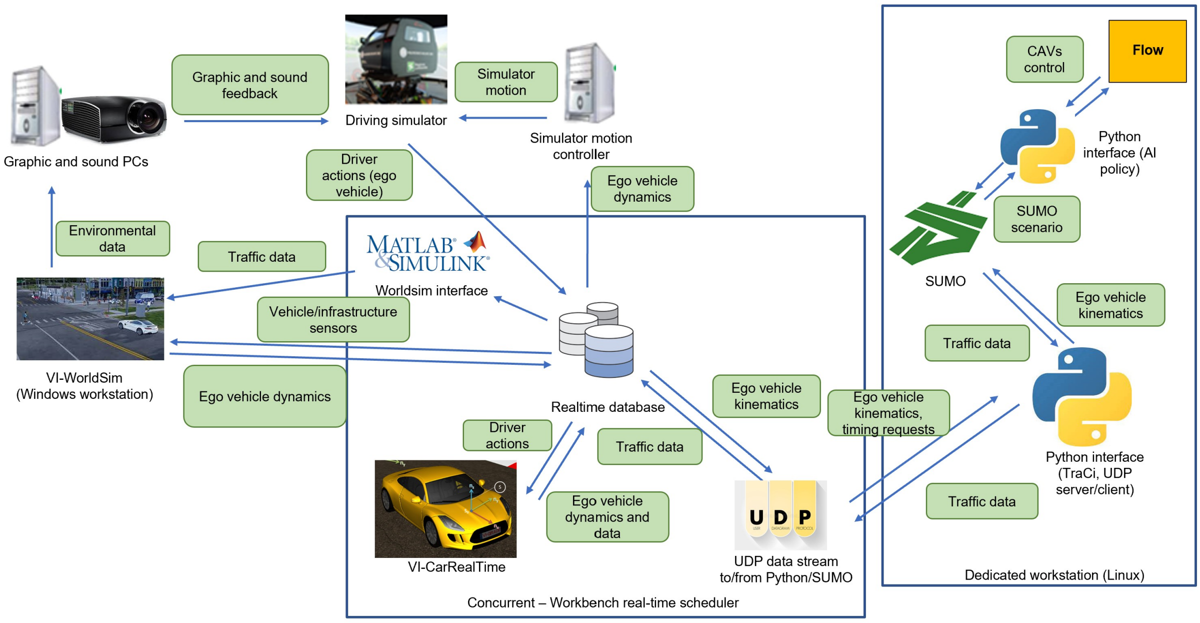

Graphic and sound PCs
Graphic and sound feedback
Driving simulator
Simulator motion
Simulator motion controller
Driver actions (ego vehicle)
Ego vehicle dynamics
Environmental data
Traffic data
MATLAB SIMULINK
Worldsim interface
Vehicle/infrastructure sensors
VI-WorldSim (Windows workstation)
Ego vehicle dynamics
Realtime database
Driver actions
Traffic data
Ego vehicle dynamics and data
VI-CarRealTime
Ego vehicle kinematics
Ego vehicle kinematics, timing requests
U D P
UDP data stream to/from Python/SUMO
Concurrent – Workbench real-time scheduler
CAVs control
Flow
Python interface (AI policy)
SUMO scenario
SUMO
Ego vehicle kinematics
Traffic data
Python interface (TraCi, UDP server/client)
Traffic data
Dedicated workstation (Linux)

Figure 4. Connections scheme.

4.1 Trajectories

Even if SUMO integration step is very short, the trajectories in SUMO do not account for vehicle dynamics and do not appear natural when used to move vehicles in the VI-WorldSim virtual environment. However, the small integration step, corresponding to a 200 HZ sampling of the motion of the vehicles is higher than the frequency of the screen (120 HZ), thus an interpolation between steps is not necessary to obtain a fluid motion of the vehicles. Therefore, the only operation on the trajectories extracted from SUMO is a simple transformation. For each trajectory in the road network loaded in SUMO, a corresponding trajectory is modeled in an auxiliary network. The corresponding trajectory connects the same nodes of the SUMO trajectory, but with a smooth and ”natural” path. At each time instant, the position of each vehicle is red in SUMO. Before sending the position to VI-WorldSim, the position is slightly modified to be consistent with the auxiliary and more natural corresponding trajectory. This operation is computationally very fast as it is just a modification of the coordinates according to a correspondence table and allows a much more realistic movements of the vehicles.

Alternatively, in some papers [1] large time increments of the order of 0.2 s are used for traffic simulation. In this case, the trajectories of the vehicles simulated by the MTS have to be interpolated to compensate for the very low update frequency of their position and orientation that prevent a fluid motion. By this approach, larger time increments of the MTS simulation allow for the simulation of larger networks, however larger position delays between the two simulation environments have to be expect. In the present paper, a smaller integration step for the MTS has been preferred to minimize the position delay.

5 Preliminary tests

The integrated system with SUMO and the dynamic driving simulator has been used to perform some preliminary experimental test with a relatively small panel of twelve drivers. The aim of the tests is to understand if the set up is able to let the testers appreciate different behaviors of the other vehicles. In particular, the target is if by changing the percentage of CAVs in the scenario, the participants could perceive a different traffic flow and if they feel comfortable and safe while driving.

As discussed in Sect. 3, the selected scenario is a three-legged single-lane roundabout. As the scenario is quite small and the maneuver is very quick, participants have been asked to enter the roundabout from all legs and always exit at the second exit. In this way, the driver, while on the circulatory roadway, has to cross one entry and can observe the behavior of the other vehicles when approaching the roundabout. A small queue of about three to five vehicles is present at each leg and the tester has to wait her/his turn to enter the roundabout. For each leg, the drivers repeat the maneuver two times, once with 20% and once with 80% of CAVs. Participants are not informed on the presence of automated cars. The scenario comprises forty vehicles, including the human driven one. After the test, a brief questionnaire is proposed to the participants. The aim of the questionnaire is to understand if the designed scenario permits to the participant to feel some differences in the two traffic situations. The participants are asked to indicate which of the two traffic situations, if any, has a more smooth traffic, feels more safe while driving and which they prefer.

Table 2. Summary of the results of the preliminary tests.

Preferred scenario	Traffic smoothness	Safety feeling	Overall preference
20% CAVs scenario	4	3	3
80% CAVs scenario	6	3	6
No difference	2	6	3

The feedback collected from the drivers involved in the test is reported in Table 2. The results show that most of the driver were able to observe differences in the two situations. Speaking with them after the tests, they also reported that the simulation was quite realistic, the traffic was smooth and the other vehicles interacted correctly with their vehicle. Referring to the answers to the questionnaire, there is a slight trend to prefer the scenario where 80% of traffic actors were automated cars. However, the panel is too small to be able to derive conclusions and more tests will be performed. The main point, however, is that the proposed integration between SUMO and a high end dynamic driving simulator can be effectively employed for the analysis of different traffic scenarios and for the study of the interactions between human drivers and CAVs.

6 Conclusion

In the present paper a integration between SUMO and a high end dynamic driving simulator has been presented. The proposed scheme of integration relays on a real-time scheduler in order to guarantee a very accurate synchronization between the two simulations and the real-time. By setting an exchanging data frequency of 200 Hz between the SUMO simulation and the driver simulation, a very small delay of 5 ms is present between the two simulations. Vehicle trajectories computed by SUMO are transformed in more natural trajectories by slightly modifying the computed position of the vehicles before sending the data to the driving simulator. The high refreshment frequency of vehicle positions and the correction of the trajectories has allowed for a very smooth and realistic co-simulation.

The proposed integration scheme, even if derived for the actual software configuration used in the paper, is actually general as real-time scheduler can be easily used to interface the most diffused driving simulator software and microscopic traffic simulators. Given the utilization of the real-time scheduler and a shared real-time database, any number of driving simulators (laptop or desktop workstations, static simulators or dynamic simulators), even running different simulation software, can be added to the network.

Preliminary tests, performed on a relatively small panel of twelve drivers, have shown that the proposed approach can be effectively employed for experimentation with microscopic traffic simulators and human in the loop. The participants have been asked to navigate a roundabout in a mixed traffic condition with different percentage of connected automated vehicles. The participants were able to observe differences in the traffic flow for different percentages of connected and automated vehicles. The participants also reported that the simulation was realistic and the vehicles simulated by SUMO interacted with their vehicle as one might expect.

Data availability statement

All available data has been included in the paper.

Author contributions

- Conceptualization: G. Mastinu and G. Previati
- Data curation: G. Previati
- Formal Analysis: G. Previati
- Funding acquisition: G. Mastinu
- Investigation: G. Mastinu and G. Previati
- Methodology: G. Mastinu and G. Previati
- Project administration: G. Mastinu
- Resources: G. Mastinu and G. Previati
- Software: G. Previati
- Supervision: G. Mastinu
- Validation: G. Mastinu and G. Previati
- Visualization: G. Previati
- Writing – original draft: G. Mastinu and G. Previati
- Writing – review and editing: G. Mastinu and G. Previati

Competing interests

The authors declare that they have no competing interests.

Funding

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101015922.

Acknowledgements

The authors wish to thank dott. Francesco Comolli and eng. Alessandro Francesconi for their help in setting up the driving simulator and performing the tests. Eng. Elena Campi, eng. Alvaro Valera Daniel and Lorenzo Uccello are gratefully acknowledged for their activities in the project.

References

[1] M. Hasan, D. Perez, Y. Shen, and H. Yang, "Distributed microscopic traffic simulation with human-in-the-loop enabled by virtual reality technologies," *Advances in Engineering Software*, vol. 154, pp. 1–16, Dec. 2021. DOI: 10.1016/j.advengsoft.2021.102985.

[2] S. E. Li, Y. Zheng, K. Li, *et al.*, "Dynamical modeling and distributed control of connected and automated vehicles: Challenges and opportunities," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, pp. 46–58, 3 Sep. 2017, ISSN: 19411197. DOI: 10.1109/MITS.2017.2709781.

[3] H. Zheng, J. Wu, K. Pan, W. Meng, and R. Li, "Research on control target of truck platoon based on maximizing fuel saving rate," *SAE International Journal of Vehicle Dynamics, Stability, and NVH*, vol. 4, 2 2020, ISSN: 23802162. DOI: 10.4271/10-04-02-0010.

[4] D. Cui, Y. Shen, H. Yang, *et al.*, "Extensible co-simulation framework for supporting cooperative driving automation research," *Transportation Research Record: Journal of the Transportation Research Board*, p. 036 119 812 211 212, Sep. 2022, ISSN: 0361-1981. DOI: 10.1177/03611981221121263.

[5] X. Zhao, X. Liao, Z. Wang, *et al.*, "Co-simulation platform for modeling and evaluating connected and automated vehicles and human behavior in mixed traffic," *SAE International Journal of Connected and Automated Vehicles*, vol. 5, 4 Apr. 2022, ISSN: 2574075X. DOI: 10.4271/12-05-04-0025.

[6] U. E. Manawadu, M. Ishikawa, M. Kamezaki, and S. Sugano, "Analysis of preference for autonomous driving under different traffic conditions using a driving simulator," *Journal of Robotics and Mechatronics*, vol. 27, pp. 660–670, 6 2015, ISSN: 18838049. DOI: 10.20965/jrm.2015.p0660.

[7] S. Espié and J. Auberlet, "Joint use of driving simulation and traffic simualtion for the study of road infrastructures and equipments," in *Joint International Conference on Computing and Decision Making in Civil and Building Engineering*, 2006, pp. 2554–2563.

[8] I. Vladisavljevic, J. M. Cooper, P. T. Martin, and D. L. Strayer, "Importance of integrating driving and traffic simulations: Case study of impact of cell phone drivers on traffic flow," in *Transportation Research Board 88th Annual Meeting*, 2009.

[9] L. Yue, M. Abdel-Aty, and Z. Wang, "Effects of connected and autonomous vehicle merging behavior on mainline human-driven vehicle," *Journal of Intelligent and Connected Vehicles*, vol. 5, pp. 36–45, 1 Feb. 2022, ISSN: 2399-9802. DOI: 10.1108/jicv-08-2021-0013.

[10] S. K. Chada, D. Gorges, A. Ebert, R. Teutsch, and C. G. Min, "Learning-based driver behavior modeling and delay compensation to improve the efficiency of an eco-driving assistance system," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, Oct. 2022, pp. 415–422, ISBN: 978-1-6654-5258-8. DOI: 10.1109/SMC53654.2022.9945577. [Online]. Available: https://ieeexplore.ieee.org/document/9945577/.

[11] M. Barthauer and A. Hafner, "Testing an adaptive cruise controller with coupled traffic and driving simulations," in *EPiC Series in Computing, SUMO User Conference 2019*, vol. 62, 2019, pp. 48–55.

[12] D. Nalic, A. Eichberger, G. Hanzl, M. Fellendorf, and B. Rogic, "Development of a co-simulation framework for systematic generation of scenarios for testing and validation of automated driving systems," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, Oct. 2019, pp. 1895–1901, ISBN: 978-1-5386-7024-8. DOI: 10.1109/ITSC.2019.8916839. [Online]. Available: https://ieeexplore.ieee.org/document/8916839/.

[13] M. Barthauer and A. Hafner, "Coupling traffic and driving simulation: Taking advantageof sumo and silab together," *EPiC Series in Engineering, SUMO 2018-Simulating Autonomous and Intermodal Transport Systems*, vol. 2, pp. 56–66, 2018.

[14] S. M. Taheri, K. Matsushita, and M. Sasaki, "Virtual reality driving simulation for measuring driver behavior and characteristics," *Journal of Transportation Technologies*, vol. 07, pp. 123–132, 02 2017, ISSN: 2160-0473. DOI: 10.4236/jtts.2017.72009.

[15] J. Kaths, B. Schott, and F. Chucholowski, "Co-simulation of the virtual vehicle in virtual traffic considering tactical driver decisions," in *EPiC Series in Computing, SUMO User Conference 2019*, 2019, pp. 21–28.

[16] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582. DOI: 10.1109/ITSC.2018.8569938.

[17] Flow-project. "Flow." (2019), [Online]. Available: https://flow.readthedocs.io/en/latest/index.html (visited on 04/12/2223).

[18] Politecnico-di-Milano. "Drismi - driving simulator politecnico di milano." (2022), [Online]. Available: https://www.drismi.polimi.it/ (visited on 04/12/2223).

[19] VI-Grade. "Vi-grade: Driving simulator." (2023), [Online]. Available: https://www.vi-grade.com/ (visited on 04/12/2223).

[20] G. Previati, G. Mastinu, and M. Gobbi, "Influence of the inertia parameters on a dynamic driving simulator performances," in *Society of Allied Weight Engineers 81st Annual Conference*, 2022, pp. 1–14.

[21] M. Bruschetta, F. Maran, and A. Beghi, "A nonlinear, mpc-based motion cueing algorithm for a high-performance, nine-dof dynamic simulator platform," *IEEE Transactions on Control Systems Technology*, vol. 25, pp. 686–694, 2 Mar. 2017, ISSN: 10636536. DOI: 10.1109/TCST.2016.2560120.

[22] Concurrent-Real-Time. "Concurrent real-time." (2017), [Online]. Available: https://concurrent-rt.com/ (visited on 04/12/2223).

[23] Eclipse-Foundation. "Sumo simulation of urban mobility." (2001), [Online]. Available: https://www.eclipse.org/sumo/ (visited on 04/12/2223).

[24] AI@Edge. "The ai@edge h2020 project." (2021), [Online]. Available: https://aiatedge.eu/ (visited on 04/12/2223).

[25] L. García Cuenca, J. Sanchez-Soriano, E. Puertas, J. Fernandez Andrés, and N. Aliane, “Machine learning techniques for undertaking roundabouts in autonomous driving,” *Sensors*, vol. 19, no. 10, 2019, ISSN: 1424-8220. DOI: 10.3390/s19102386. [Online]. Available: https://www.mdpi.com/1424-8220/19/10/2386.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.214

Published: 29 June 2023

Comparing Measured Driver Behavior Distributions to Results from Car-Following Models using SUMO and Real-World Vehicle Trajectories from Radar

SUMO Default vs. Radar-Measured CF model Parameters

Max Schrader[1] [https://orcid.org/0000-0003-3673-7672], Mahdi Al Abdraboh[1] [https://orcid.org/0009-0001-4906-297X], and Joshua Bittle[1] [https://orcid.org/0000-0003-4524-3316]

[1] University of Alabama, Tuscaloosa, Alabama, USA

Abstract: In this study, the physical principles governing car-bfollowing (CF) behavior and their impact on traffic flow at signalized intersections are investigated. High temporal-resolution radar data is used to provide valuable insights into actual CF behavior, including acceleration, deceleration, and time headway distribution. Demand-calibrated SUMO simulations are run using empirical CF parameter distributions, and three CF models are evaluated: IDM, EIDM, and Krauss. By emulating radar data in SUMO and processing simulated vehicle traces, discrepancies between empirical and simulated parameter distributions are identified. Further analysis includes comparisons with default SUMO CF model parameters. The findings reveal that measured accelerations differ from CF model parameter accelerations and using the empirical value ($\mu = 0.89 m/s^2$) leads to unrealistic simulations that fail volume-based calibration. Default parameters for all three models reasonably approximate the mean and median of measured parameters, but fail to capture the true distribution shape, partly due to homogeneity when using default parameters. The results show that it is more effective to simulate with the default parameters provided by SUMO rather than using measurements of real-world distributions without additional calibration. Future work will investigate closing the loop between the measured real-world and SUMO distributions using traditional calibration tactics, as well as assess the impact of calibrated vs. default CF parameters on simulation outputs like fuel consumption.

Keywords: traffic micro-simulation, car-following models, car-following calibration, intelligent driver model, roadside radar data

1 Introduction

The modeling of car-following behavior is a crucial component of traffic micro-simulation, as it captures the longitudinal interactions between drivers and their preceding vehicles. Car-following (CF) models have been studied extensively by researchers since the 1950s, with early models proposed by Reuschel [1] and Pipes [2]. Over the years, the CF model space has evolved into several subcategories, including mathematical or engineering models, data-driven models, and hybrid models [3]. While the latter two modeling techniques have gained significant research attention, the use of mathematical models such as the intelligent driver model (IDM)[4] is still prevalent in traffic micro-simulation software such as SUMO [5]. Traffic micro-simulation has many use cases, but regardless, the reliability of the insights gained from simulation depends entirely on the model's ability to represent the underlying real-world traffic network[6].

To gain confidence in model outputs, a process of calibration and validation is required [6], [7], where each of traffic micro-simulation's sub-models, including the CF model, are optimized. There are several types of calibration referenced in literature, but they can generally be summarized in three categories: capacity calibration, route/demand calibration, and individual trajectory-based calibration [6], [8]. The first two calibration categories typically rely on aggregate measures, such as loop detector counts, travel times, and saturation flow rates. Although widely used, these measures may not always accurately reflect real-world vehicle behavior [8], [9]. Moreover, studies have shown that different CF model behaviors can lead to similar travel times, saturation flow, delays, and queues [10]. This lack of uniqueness in the CF model parameters that best fit the observed traffic measures has been previously studied [11] and can be problematic in estimating the emission and energy consumption of a traffic network using microsimulation. This is because factors such as acceleration, deceleration, jerk, and speed are highly influential in vehicle-level emissions and fuel consumption, which are difficult to estimate accurately without considering individual vehicle trajectories [10], [12].

Fuel consumption estimation is given as an example to emphasize the importance of the CF model parameters themselves. They are a central part of traffic simulation, yet its difficult to find consensus on the correct parameter settings and/or range of settings. The default values in SUMO can differ substantially from calibration literature (*i.e.* SUMO default acceleration[1] is 2.6m/s^2, whereas commonly cited calibration literature ranges from 1.00 to 1.58m/s^2 [13]). Further, the vast majority of calibration literature relies on NGSIM dataset, which was collected in 2005 [14], even though it is known that CF parameters evolve over time or change depending on geography and locality [15].

The solution would seem to require that practitioners calibrate their own CF models, however the complexity of performing trajectory-based calibration in practice cannot be understated. It is not as simple as measuring accelerations or velocity in the field, as many CF model parameters cannot be derived from macroscopic measurement or do not have physical equivalents [16]. Instead, the practitioner must collect high resolution trajectory data, clean and process the data, extract leader-follower pairs, and then use computationally expensive optimization methods, such as genetic algorithms, to find the correct CF parameter settings. Recent studies have shown that state-of-the-art CF model calibration still results in significant error [17] and according to Ossen and Hoogendoorn ”calibration based on real trajectory data turns out to be far from trivial” [18]. On the topic of fuel consumption, sensitivity analyses show that CF model parameters are important in individual trajectory estimation and subsequent fuel and emissions estimation, but their importance diminishes as more vehicles are considered in aggregate and average parameters may be sufficient [8].

The review of literature leads practitioners to the following conundrum: it is understood that CF model parameters are important tuning knobs for calibrated outputs, but without high-resolution and complete trajectory information for the network, what should the parameter settings be? This study aims to provide additional context to this

[1] https://sumo.dlr.de/docs/Definition_of_Vehicles%2C_Vehicle_Types%2C_and_Routes.html

problem using data from a real-world network outfitted with radars that record vehicle positions and speeds, which (along with cameras) is becoming more popular with the rise of intelligent traffic systems (ITS). While literature has shown that these partial trajectories may be insufficient for traditional CF model calibration [17], it is still possible to extract distributions of observed parameters such as vehicle acceleration, headway and the free-flow speed. The efficacy of using these measured distributions as the CF model parameters is explored, in addition to the ability of the SUMO CF models with their default parameters to recreate the measured distributions.

2 Car Following Models

Intelligent Driver Model: The IDM CF model was first proposed by Treiber, Hennecke and Helbing in 2000 [4]. The IDM model determines the acceleration of the follower vehicle, $\dot{v}_f$, as a function of current velocity, v_f, the distance to the leading vehicle, s, and the difference in velocity between the leader and follower, Δv. The acceleration, $\dot{v}_f$, at any time t is written as

$$\dot{v}_f(v_f, s, \Delta v) = a\left[1 - \left(\frac{v_f}{v_0}\right)^{\beta} - \left(\frac{s^*(v_f, \Delta v)}{s}\right)^2\right] \tag{1}$$

where s^* is the desired minimum following gap, v_0 is the free-flow speed on the road, β is a tuning acceleration exponent, and a is the maximum follower acceleration. The target simulation network described in Section 3.2 has varied speed limits, thus a static v_0 is not applicable. SUMO instead models the desired velocity as a speed factor, SF_v, which is a multiplier on the speed limit, making the equation for v_0

$$v_0 = \text{SF}_v \cdot \text{speed limit}_f \tag{2}$$

where speed limit_f is the follower vehicle's applicable speed limit. The following gap is formulated as a function of current velocity and the difference in leader and followers velocity and given as

$$s^*(v_f, \Delta v) = s_0 + \tau + \frac{v_f \Delta v}{2\sqrt{ab}} \tag{3}$$

where a is again the follower's maximum acceleration, b is the follower's maximum deceleration, τ is the minimum time headway, and s_0 is the minimum space between the follower and lead vehicle. In total, the IDM model has 6 tuning parameters $\{$ a, b, τ, s_0, SF_v, β $\}$ with [13] fixing $\beta = 4$, reducing the dimensionality to 5.

Krauss Model: The default CF model in SUMO is Krauss's [19], [20]. It is a collision free model, with each follower vehicle having a safe following speed, v_{safe}, computed at every simulation step from the velocity, v_f, using the following equation:

$$v_{\text{safe}}(t) = v_l + \frac{g(t) - v_l(t) \cdot \tau}{\frac{v_f}{b \cdot v_f} + \tau} \tag{4}$$

where t is the simulation time, $v_l(t)$ is the velocity of the lead vehicle at time t, $g(t)$ is the gap between the vehicles, τ is the reaction time of the driver, and b is the deceleration function [21]. Because the acceleration should be bounded by the physical limitations of the vehicle, the actual desired speed, $v_{\text{des}}(t)$, is calculated as:

$$v_{\text{des}}(t) = \min\left[v_{\text{safe}}(t), v_f(t) + a, v_0\right] \tag{5}$$

where a is the maximum acceleration capability of the vehicle and v_0 is the maximum speed that the driver would drive on the road according to Equation 2. The maximum acceleration parameter does not represent the physical limit of the vehicle, but rather the maximum acceleration that a driver would choose[2] [21]. Like the IDM model, a, b, and τ are available as tuning parameters.

Extended Intelligent Driver Model: The extended intelligent driver model was proposed by Salles, Kaufmann, and Reuss in 2020 to more accurately model human driving behavior, especially the drive-off trajectories [22]. The model is based on IDM but includes many improvements from both Treiber ([23], [24]) and the authors themselves. For brevity, the equations for the EIDM are not listed in this work, rather the reader is redirected to the referenced work. Along with IDM and the Krauss model, it also contains tuneable acceleration, deceleration and headway parameters.

3 Data and Models

3.1 Radar Processing

The trajectory data was collected using radars from InnoSenT GmbH [3]. The radar used for trajectory processing is located on the north side of the west-most traffic signal (TL1) in Figure 1 and captures the west-bound approach, as well as the east-bound departure. The radars can report vehicle position, velocity, and vehicle length every 50ms, however, due to internet network speed restrictions, data for individual trajectories is actually recorded with a period of 100-200ms. Vehicle positions are reported in the radar's coordinate system and must be transformed to match the positions to the underlying road network. Vehicles that enter the radar's field of view are assigned a unique identifier, making it easy to extract the complete trajectory of a vehicle as it passes through the radar. However, filtering the trajectories is necessary as trajectories can have discontinuities, the radar can identify non-vehicle objects as vehicles (data points in the grass in Figure 1), and the object id for a vehicle can switch while the vehicle is still in the sensing range.

The radars are used for two tasks in this work: extraction of CF behavior from the velocity profiles and volume calculation. To deal with radar data issues in the context of velocity profile processing, a polygon is placed around the west-bound approach to TL1 in Figure 1. This region has high data integrity and low probability of interference. The trajectory data is filtered so that only vehicles which pass completely through the box are considered. Their trajectories are truncated to only include the intra-box data and then processed according to the methodology in Section 3.3. The other tasks is volume calculation, which uses the radars' positional information to count vehicles that cross the stop bar at all of TL1's approaches. Filtering the radar with the aforementioned strategy could introduce bias into the dataset as it only keeps vehicles for which the radar is able to maintain a steady track, however these cases typically result when vehicles become obstructed from the radar.

[2] https://sumo.dlr.de/docs/Definition_of_Vehicles%2C_Vehicle_Types%2C_and_Routes.html#vehicle_types

[3] https://www.innosent.de/

3.2 SUMO Simulation

The simulated SUMO network represents a two intersection corridor of Tuscaloosa, Alabama primarily consisting of US-82McFarland Blvd between Airport and Harper Roads. Figure 1 shows the SUMO model of the target network overlaid on geo-located satellite images. The call out displays the location of the radar, as well as a sample of its geo-located data and the box used to filter trajectories. Both intersections in the network are signalized (marked as TL1 and TL2 in Fig. 1), and the field controllers were emulated using NEMA controllers in SUMO with matching configurations [25]. They all operated in coordinated mode during the simulated period of time, with actuation on the non-coordinated phases.

The polygon used to filter the real-world data was also integrated into the geo-located SUMO simulation. By combining the floating car data output of SUMO with polygon-based filtering, a data set was obtained that closely resembles the data described in Section 3.1. The floating car output was processed using the same methodology that was applied to the radar data.

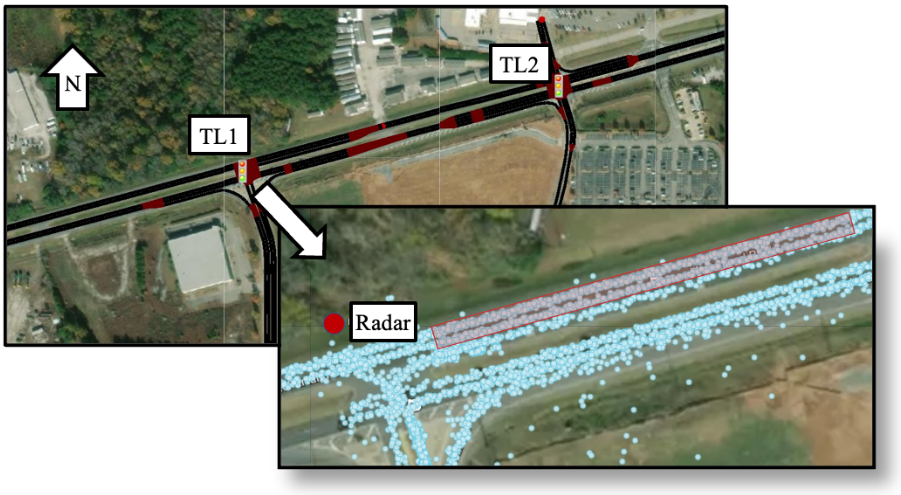


Figure 1. SUMO model of the simulated network including two intersections, overlaid with a close up of the radar location, sample data points, and the box used for filtering.

Simulation demand is generated using both the radar displayed in Figure 1 as well as a radar on the other side of the intersection that captures the east bound approaches. Data was collected from the radars on January 13th, 2023 and the simulation time spans from 5:00AM to 11:00PM on the target day, covering periods of low volume as well as the morning rush around 8AM, which is apparent in Figure 2. The box used for radar filtering corresponds to the West bound straight volume.

To turn the radar data into traffic counts, the number of tracked objects that cross the intersection stop bar for each approach are aggregated into five-minute intervals. Additionally, turn count restrictions are put in place at TL2 so that the majority of traffic traverses the entire network on US-82 (the major east/west road). The volumes and turn counts are passed as an input to `routeSampler` [26] with the Poisson flow option, which is chosen to increase the randomness of the simulation. The calibrated simulation is evaluated against the GEH metric [27], which compares simulated volumes to observed volumes using the following equation:

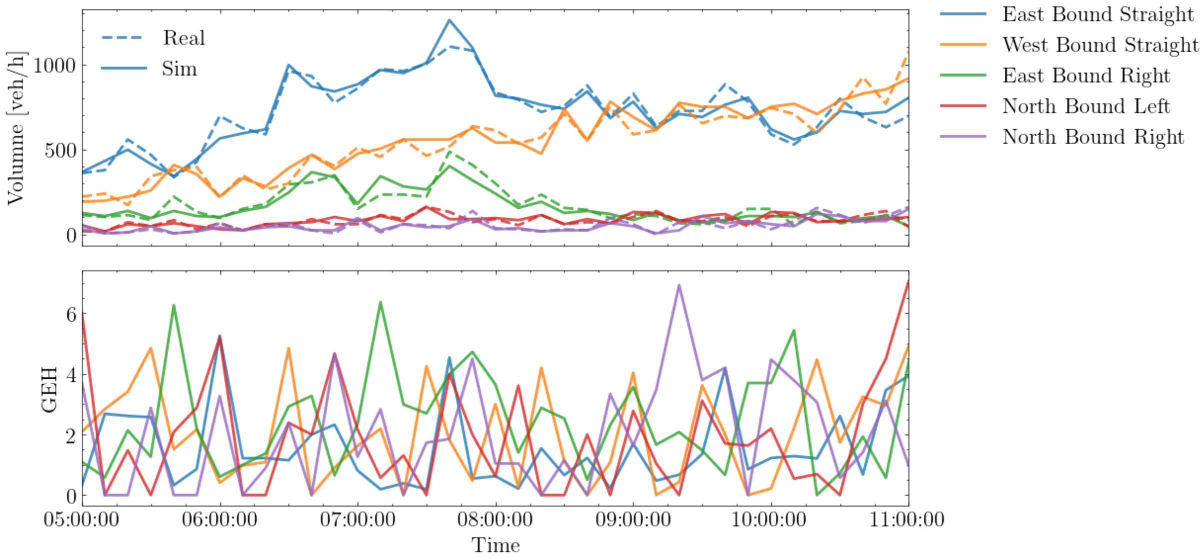


Figure 2. Time plots of (top) SUMO simulation vs. measured volume passing through TL1 and (bottom) corresponding GEH results.

$$\text{GEH}_i = \sqrt{\frac{2\left(M_t^i - C_t^i\right)^2}{M_t^i + C_t^i}} \tag{6}$$

where M_t^i is the simulated hourly volume at location i and aggregation period t. Correspondingly, C_t^i is the corresponding measured hourly volume. Applying Equation 6 to the simulation output results in a 10 minute GEH < 5 at 94% of time window - location pairs when simulating with SUMO default CF model parameters, which passes the common GEH target of < 5 at 85% of locations [6], is visually represented in Figure 2.

3.3 Trajectory Processing

3.3.1 Acceleration & Deceleration

A sample of the raw and processed radar trajectories is shown in Figure 3. The left figure is a time-space diagram of west-bound vehicles as they traverse the network. The right plot shows a piece-wise linear fit overlaid on the raw velocity data of the red vehicle in the time-space diagram. From each segment a fit quality, R^2, slope, and duration are obtained for further processing. The vehicle trajectory shown in Figure 3 captures sample of an accelerating and decelerating vehicle.

A graphical representation of the impact of minimum time and fit quality thresholds on the distribution of measured parameters is depicted in Figure 4. The top row shows contour of median acceleration and the number of unique vehicles as a function of minimum time and R^2 threshold. The bottom row shows the measured distributions for acceleration and deceleration, with the label and color corresponding to the annotated positions on the top row. Based on the figure, location 5 was chosen as the final aggregation settings, with a minimum time of 1 second and $R^2 > 0.95$, due to the high confidence in the linear fit of trajectory while still containing enough samples to be representative of the population. The resulting distributions are presented quantitatively later in Table 1. The shapes of the distributions of acceleration and deceleration determined in Figure 4 are right-skewed due to the low mean acceleration values and because each parameter has a lower bound of 0. Consequently, median acceleration

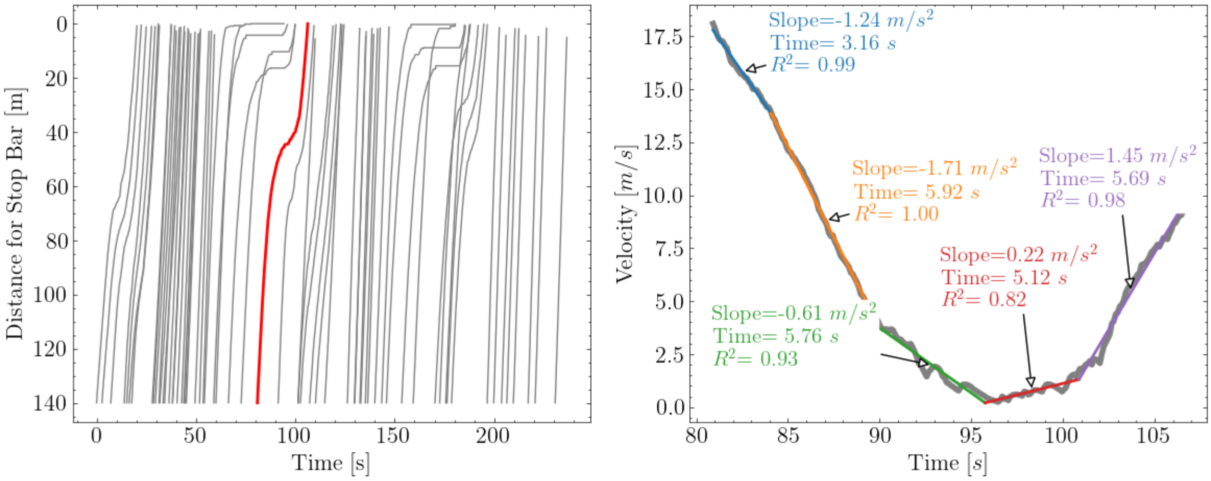


Figure 3. Examples of (left) raw data over a short time period capturing many vehicles and (right) an example of piece-wise linear fit applied to velocity profile of a sample vehicle trace.

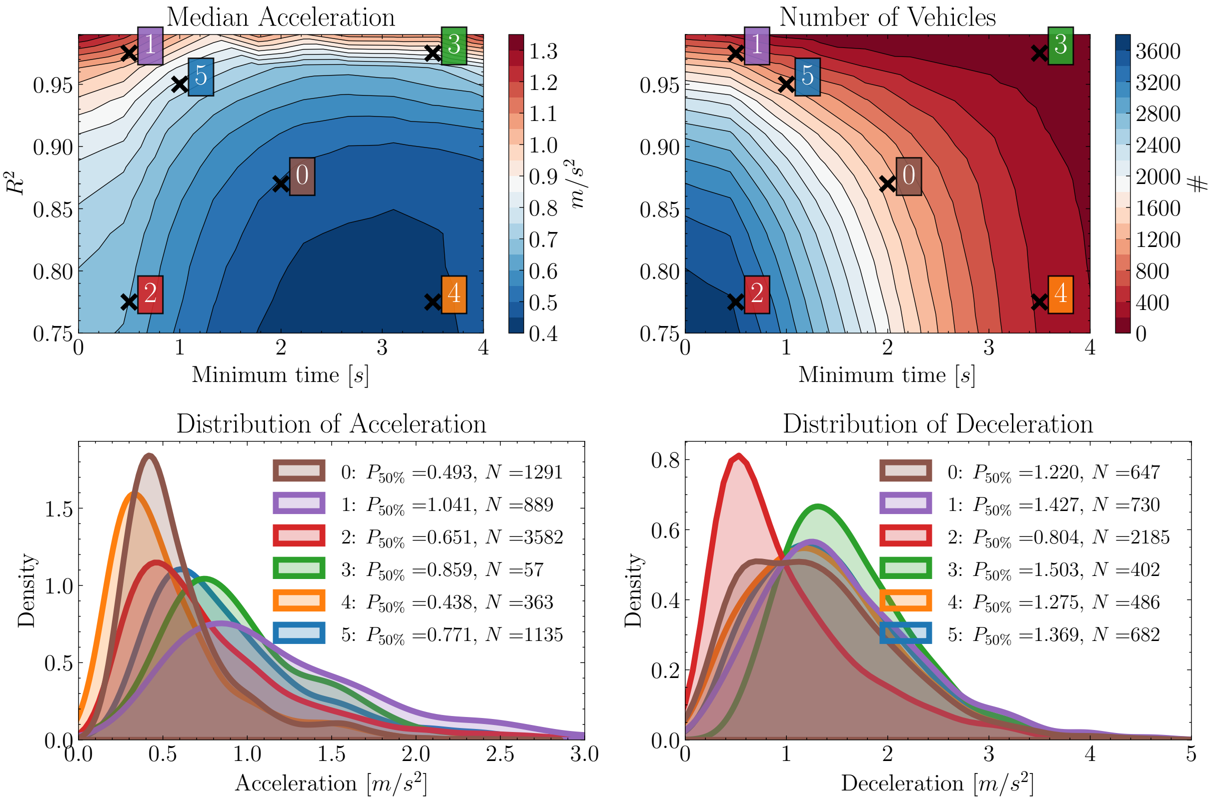


Figure 4. Highlights the sensitivity of the median ($P_{50\%}$) acceleration and number of vehicles to the R^2 and minimum time headway filter. The bottom row shows the measured acceleration and deceleration kernel density estimations for the 5 called out R^2 and minimum time pairs.

was considered as primary metric for evaluating threshold selection as it best represents central tendency of the acceleration's and deceleration's R^2 values.

3.3.2 Headway & Free-Flow Speed

To determine the headway of vehicles in the radar data, a mapping process was performed to assign each vehicle to its respective lane. This was achieved by dividing the box illustrated in Figure 1 into two separate segments, thereby enabling lane identification. The arrival time of each vehicle at distances of 100m, 60m, and 40m from the stop bar was then calculated using linear interpolation. Vehicles that switched lanes during the data collection period were filtered out. The remaining vehicles were sorted by time and leader-follower pairs were identified for each lane. The headway was calculated as the average time difference between the leader and follower at each of the three aforementioned distances. Headway times of longer than 5 seconds were deemed outside of the CF regime, inline with prior literature [10], [28], [29].

The derived acceleration and headway data was utilized to determine the free-flow speed of the network. However, it was noted that the measured range was signalized and a simple average would be significantly impacted by stopped vehicles. To address this issue, filtering was employed in a manner similar to prior literature [10]. The method considered only vehicles with time headway greater than 5 seconds and excluded vehicles with acceleration or deceleration greater than 1 m/s^2. As can be inferred from the sample of raw data in Fig. 3 in just a few minutes of data there are large numbers of free-flowing and leader/follower pairs from which to extract the desired behavioral distributions.

3.4 Vehicle Distribution Creation

In the context of SUMO simulation, vehicle CF attributes are defined by the vehicle type attribute, which can be assigned to each individual vehicle entering the network, or specified in a vehicle type distribution file[4] from which SUMO samples when generating routes. The presence of individual vehicle data in the radar representation provides the opportunity to construct a heterogeneous vehicle distribution through sampling, either correlated or uncorrelated.

The correlation of parameters in the simulation of traffic flow has garnered attention in the literature, as it has been shown to impact the realism of simulation results [30], [31]. To create a correlated distribution from the radar data, only vehicles with deceleration, acceleration, and headway less than 5 seconds are considered. These vehicles are used to create synthetic vehicles by combining the acceleration and deceleration events, resulting in a set of acceleration, deceleration, and headway values that all derive from a single vehicle, in addition to corresponding vehicle length, which is measured by the radar. However, since the vehicle is in the CF regime, its matching speed factor is not available. In this scenario, the speed factor is sampled from the overall distribution. There are 1135 vehicles in that radar dataset that makes 1290 accelerations (refer to Section 3.3.1 for further details). Similarly, 682 vehicles decelerate, resulting in a total of 1193 recorded decelerations. Additionally, there were 1831 vehicles with headways less than 5 seconds, out of which 48 vehicles were present in both the acceleration and deceleration datasets. Because of the difference, due largely to the dramatically reduced sample size, the correlated model parameters were not used fur-

[4] https://sumo.dlr.de/docs/Definition_of_Vehicles%2C_Vehicle_Types%2C_and_Routes.html#route_and_vehicle_type_distributions

ther in this work. Future work will expand raw data recording period and perhaps result in large enough size to apply correlated sampling for comparison. In this study however, only the uncorrelated case was considered in Section 4. In the uncorrelated case, the acceleration, tau, deceleration, speed factor, and length columns are independently sampled from the distributions of measured parameters.

The study also investigated the use of acceleration and deceleration measures that occur outside the CF regime. This is because both the Krauss and IDM model's representation of CF acceleration is intended to reflect the maximum follower acceleration, which could emerge in the absence of a leader vehicle. However, the measured distributions of parameters outside of the CF regime were found to not significantly differ from those within the CF regime according to the Mann-Whitney test ($U = 4.21\mathrm{e}5$, $p = 0.43$), and thus were not included in the evaluation.

4 Results

The constructed distributions of CF parameters were simulated in SUMO `Version 1.16.0` with a simulation step size of $0.1s$. Two experiments were conducted, one with the default CF model parameters and one with parameters sampled from the measured distributions. In each experiment, the models had an `actionStepLength` of $0.2s$[5]. All vehicles were simulated as passenger cars with the emissions and fuel consumption model `PHEMlight/PC_G_EU4` [32].

To ensure the robustness of the findings, each CF model discussed in Section 2 (IDM, EIDM, and Krauss) was simulated 30 times, only varying the random seed. Subsequently, the output of all 180 simulations was processed according to the methods described in Section 3.3.

4.1 Simulation Comparison

In Figure 5, the empirical cumulative distribution functions (eCDF's) obtained by simulating with both the default parameters of a CF model and the results obtained from sampled parameters are presented. The colors in the figure are paired such that the darker shade represents the sampled parameters and the lighter shade represents the default parameters. However, it is immediately apparent that neither the sampled nor default parameters accurately reproduce the empirical distributions of acceleration and deceleration. Nonetheless, the default models do perform reasonably well at modeling headway and free-flow speed, although they still under-predict the speed and fail to capture lower headway vehicles.

The results are summarized in Table 1, with the mean (μ) and 50th percentiles ($P_{50\%}$). The standard deviation of the distributions are not presented, as the majority cannot be approximated by a Gaussian distribution. Also included in Table 1 is the resulting fuel consumption per vehicle. In each row the bolded numbers represent the closest to the real world. The default parameters perform best for all mean and median cases, with the mean acceleration of Krauss being $0.05m/s^2$ away from the measured acceleration mean. Krauss also approximates the free-flow speed the best. For deceleration, the default IDM model has the closest mean and median ($0.14m/s^2$ and $0.03m/s^2$ respectively). The default IDM also performs the best at headway estimation. However, the default parameters fail to capture the tails of the distributions well, which is understandable given that default parameters are a homogeneous fleet, as every car that enters

[5] https://sumo.dlr.de/docs/Car-Following-Models.html#actionsteplength

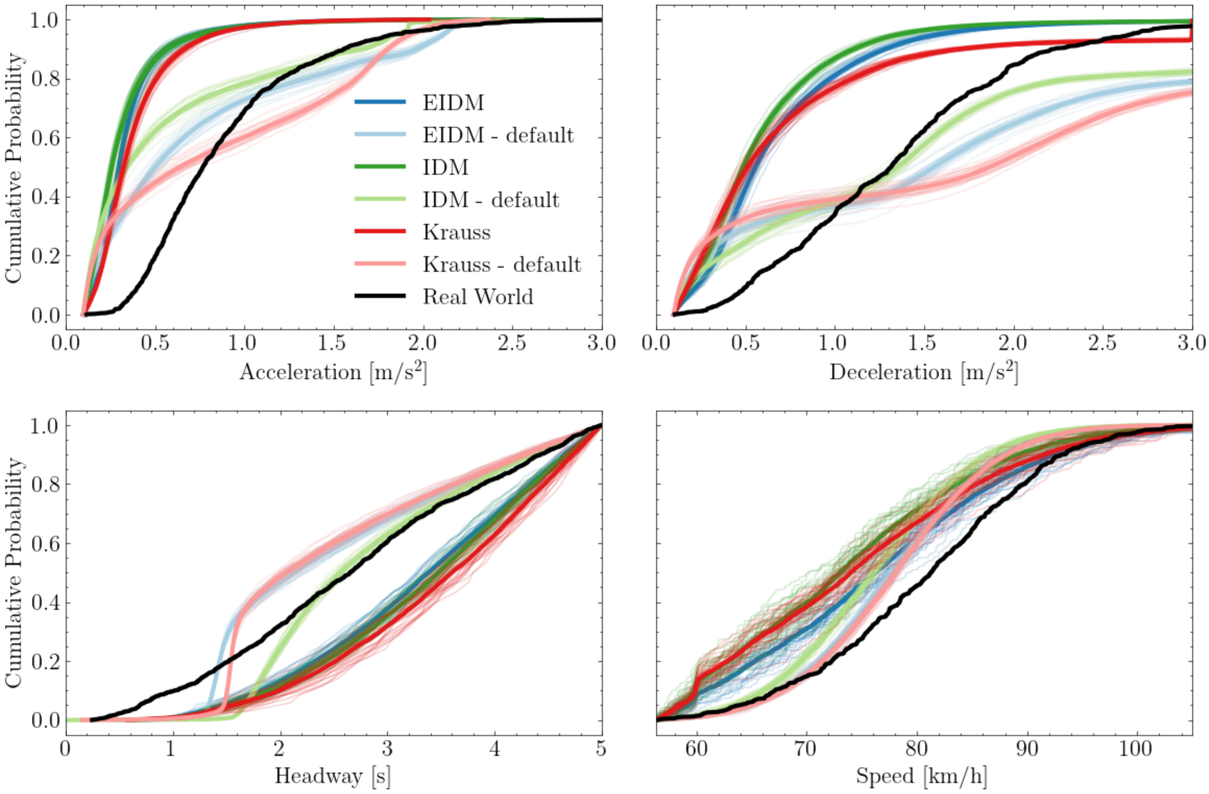


Figure 5. Empirical cumulative distribution functions for acceleration, deceleration, headway and free-flow speed. The colors are paired such that the darker shade represents the sampled parameters and the lighter represents the default. The thin individual lines represent the CDF from one simulation.

the network has the same set of parameters in the default case.

The explanation for the poor performance of the sampled parameters is two fold. For one, the fact that CF model parameters are not physical or cannot simply be found via macroscopic measurements has been discussed in literature [8], [11], [16]. Secondly and perhaps more importantly in the context of SUMO, the low acceleration measured in the real-world data ($P_{50\%} = 0.77 m/s^2$) does not result in realistic traffic flows when applied to simulation. Long queues develop in turn lanes, gridlock ensues, and GEH calibration fails in the west bound straight approach. This disruption of regular traffic flow causes elevated fuel consumption per vehicle in the sampled simulations. Due to the complexity of isolating the specific contributions of the parameters from the fuel consumption resulting from congestion, the findings of the fuel consumption analysis are not presented in this study. Future work must address this issue as the use of microsimulation tools such as SUMO to study traffic control optimization for energy reductions will rely on accurate predictions of driver behavior and driving trajectories beyond typical measures of vehicles per hour or average speed.

5 Summary, Conclusions & Future Work

This work presented the efficacy of using measured distributions of acceleration, deceleration, time headway and free-flow speed as their corresponding parameters in SUMO CF models. The distributions were acquired via processing of data from signal pole mounted radar units, which return both the position and velocity of individual vehicles as they approach a signalized intersection. The acceleration, deceleration

Table 1. Summarizes the eCDF's presented in Figure 5. The simulated value nearest the measured is presented in bold. The columns "Samp." and "Def." correspond to Sampled and Default parameters.

		EIDM		IDM		Krauss		**Measured**
	Stat.	Samp.	Def.	Samp.	Def.	Samp.	Def.	
Freeflow	μ	76.8	77.8	73.9	76.3	74.9	7**8.1**	**80.6**
Speed $[km/h]$	$P_{50\%}$	76.3	77.9	73.0	76.1	74.0	**78.1**	**81.3**
Accel. $[m/s^2]$	μ	0.4	0.8	0.3	0.6	0.4	**0.8**	**0.9**
	$P_{50\%}$	0.3	0.5	0.3	0.3	0.3	**0.6**	**0.8**
Decel. $[m/s^2]$	μ	0.7	1.8	0.6	**1.5**	0.8	1.9	**1.4**
	$P_{50\%}$	0.6	1.6	0.5	**1.4**	0.5	1.9	**1.3**
Headway $[s]$	μ	3.3	2.5	3.4	**2.8**	3.5	2.5	**2.7**
	$P_{50\%}$	3.4	2.1	3.5	**2.6**	3.6	2.1	**2.7**

and free-flow speed were found using piece-wise linear fit, and headway via linear interpolation. After aggregating the data into corresponding distributions, they were sampled and simulated in SUMO via vehicle distribution files. The SUMO demand was calibrated with the radar data, ensuring that simulation matched the volume of the measured day. Once simulated, the SUMO floating car data output was processed in an identical manner to the radar data, and the resulting SUMO distributions were compared to the empirical. In addition to the empirical distributions, the CF models in SUMO were also simulated using their default parameters to assess how well the defaults re-created the real distributions.

Based on the study's results, it can be concluded that using only measured accelerations and decelerations, independent of a leader-follower relationship, as a basis for CF model parameters is insufficient. While these measurements can replicate the heterogeneity of traffic, conducting SUMO simulations with a range of low accelerations ($P_{50\%} = 0.77 m/s^2$) leads to congested simulations that do not meet basic calibration standards. Due to the ensuing congestion and calibration failure, it becomes infeasible to compare the fuel consumption of simulations with sampled parameters to those with default settings. Consequently, initiating simulations with the default parameters provided by SUMO may prove more advantageous than using the presented measurements without further calibration of the car-following model.

However, there is a still a significant gap in distributions of primarily acceleration behavior which should be investigated. Moving forward, the next step is to close the loop between simulation distributions and their real-world counterparts by calibrating the CF model. This can be done through either fitting the simulation distributions to the measured data or analyzing the trajectories themselves. Both methods should be examined and evaluated for their impact particularly on fuel consumption. Additionally, it may be beneficial to assess how parameters change based on the time of day, volume of traffic, and at different locations in the network, as well as the impact of correlated versus uncorrelated behaviors on CF model parameters and compare their influence on simulation outputs.

CRediT authorship contribution statement

Maxwell Schrader: Conceptualization, Methodology, Software, Formal Analysis, Visualization, Writing - original draft, Writing - review & editing. **Mahdi Al Abdraboh:** Methodology, Software, Writing - original draft. **Joshua Bittle:** Conceptualization, Methodology, Resources, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Acknowledgements and Declarations

This material is based upon work supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the the Vehicle Technologies Office (VTO) and through Energy Efficient Mobility Systems (EEMS) program award number DE-EE0009210. The views expressed herein do not necessarily represent the views of the U.S. Department of Energy or the United States Government. This work was also supported by The University of Alabama's Center for Advanced Vehicle Technology (CAVT).

The authors have no conflicts of interests to declare that are relevant to the content of this article.

Currently data provided by request to corresponding author, though future plans include making this data and more from the dataset publicly available online.

References

[1] A. Reuschel, "Fahrzeugbewegungen in der kolonne," *Osterreichisches Ingenieur Archiv*, vol. 4, pp. 193–215, 1950.

[2] L. A. Pipes, "An operational analysis of traffic dynamics," *Journal of applied physics*, vol. 24, no. 3, pp. 274–281, 1953. DOI: 10.1063/1.1721265.

[3] Y. Zhang, X. Chen, J. Wang, Z. Zheng, and K. Wu, "A generative car-following model conditioned on driving styles," *Transportation research part C: emerging technologies*, vol. 145, p. 103 926, 2022. DOI: 10.1016/j.trc.2022.103926.

[4] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000. DOI: 10.1103/PhysRevE.62.1805.

[5] 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: https://elib.dlr.de/124092/.

[6] J. Barceló *et al.*, *Fundamentals of traffic simulation*. Springer, 2010, vol. 145. DOI: 10.1007/978-1-4419-6142-6.

[7] K. Wunderlich, M. Vasudevan, P. Wang, R. Dowling, A. Skabardonis, and V. Alexiadis, *Tat volume iii: Guidelines for applying traffic microsimulation modeling software 2019 update to the 2004 version (fhwa-hop-18-036)*, English, USDOT & Noblis, Washington, DC, 2019, 130 pp. [Online]. Available: https://trid.trb.org/view/1607216, released.

[8] T. V. da Rocha, L. Leclercq, M. Montanino, *et al.*, "Does traffic-related calibration of car-following models provide accurate estimations of vehicle emissions?" *Transportation research part D: Transport and Environment*, vol. 34, pp. 267–280, 2015. DOI: 10.1016/j.trd.2014.11.006.

[9] V. Punzo and F. Simonelli, "Analysis and comparison of microscopic traffic flow models with real traffic microscopic data," *Transportation Research Record*, vol. 1934, no. 1, pp. 53–63, 2005. DOI: 10.3141/1934-06.

[10] L. Jie, H. Van Zuylen, Y. Chen, F. Viti, and I. Wilmink, "Calibration of a microscopic simulation model for emission calculation," *Transportation Research Part C: Emerging Technologies*, vol. 31, pp. 172–184, 2013. DOI: 10.1016/j.trc.2012.04.008.

[11] J. Asamer, H. J. van Zuylen, and B. Heilmann, "Calibrating vissim to adverse weather conditions," in *2nd International Conference on Models and Technologies for Intelligent Transportation Systems*, 2011, pp. 22–24. [Online]. Available: https://research.tudelft.nl/en/publications/calibrating-vissim-to-adverse-weather-conditions.

[12] B. Notter, M. Keller, and B. Cox, "Handbook emission factors for road transport 4.2," *INFRAS, Bern*, 2022. [Online]. Available: https://www.hbefa.net/e/help/HBEFA42_help_en.pdf.

[13] A. Kesting and M. Treiber, "Calibrating car-following models by using trajectory data: Methodological study," *Transportation Research Record*, vol. 2088, no. 1, pp. 148–156, 2008. DOI: 10.3141/2088-16.

[14] V. G. Kovvali, V. Alexiadis, and L. Zhang PE, "Video-based vehicle trajectory data collection," Tech. Rep., 2007. [Online]. Available: https://trid.trb.org/view/801154.

[15] M. Treiber and A. Kesting, "Microscopic calibration and validation of car-following models–a systematic approach," *Procedia-Social and Behavioral Sciences*, vol. 80, pp. 922–939, 2013. DOI: 10.1016/j.sbspro.2013.05.050.

[16] L. Li, X. M. Chen, and L. Zhang, "A global optimization algorithm for trajectory data based car-following model calibration," *Transportation Research Part C: Emerging Technologies*, vol. 68, pp. 311–332, 2016. DOI: 10.1016/j.trc.2016.04.011.

[17] A. Sharma, Z. Zheng, and A. Bhaskar, "Is more always better? The impact of vehicular trajectory completeness on car-following model calibration and validation," *Transportation research part B: methodological*, vol. 120, pp. 49–75, 2019. DOI: 10.1016/j.trb.2018.12.016.

[18] S. Ossen and S. P. Hoogendoorn, "Validity of Trajectory-Based Calibration Approach of Car-Following Models in Presence of Measurement Errors," *Transportation Research Record*, vol. 2088, no. 1, pp. 117–125, 2008. DOI: 10.3141/2088-13.

[19] S. Krauß, "Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics," 1998. [Online]. Available: https://www.osti.gov/etdeweb/biblio/627062.

[20] S. Krauß, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Physical Review E*, vol. 55, no. 5, p. 5597, 1997. DOI: 10.1103/PhysRevE.55.5597.

[21] L. Bieker-Walz, M. Behrisch, M. Junghans, and K. Gimm, "Evaluation of car-following-models at controlled intersections," Tech. Rep., 2017. [Online]. Available: https://elib.dlr.de/115720.

[22] D. Salles, S. Kaufmann, and H.-C. Reuss, "Extending the intelligent driver model in sumo and verifying the drive off trajectories with aerial measurements," *SUMO Conference Proceedings*, vol. 1, pp. 1–25, Jul. 2022. DOI: 10.52825/scp.v1i.95.

[23] M. Treiber and A. Kesting, "The intelligent driver model with stochasticity-new insights into traffic flow oscillations," *Transportation research procedia*, vol. 23, pp. 174–187, 2017. DOI: 10.1016/j.trpro.2017.05.011.

[24] M. Treiber, A. Kesting, and D. Helbing, "Delays, inaccuracies and anticipation in microscopic traffic models," *Physica A: Statistical Mechanics and its Applications*, vol. 360, no. 1, pp. 71–88, 2006. DOI: 10.1016/j.physa.2005.05.001.

[25] M. Schrader, Q. Wang, and J. Bittle, "Extension and Validation of NEMA-Style Dual-Ring Controller in SUMO," in *SUMO User Conference*, 2022. DOI: 10.52825/scp.v3i.115.

[26] Deutsches Zentrum für Luft-und-Raumfahrt (DLR), *SUMO RouteSampler*, Aug. 2020. [Online]. Available: https://sumo.dlr.de/docs/Tools/Turns.html#routesamplerpy.

[27] H. Agency, *Design Manual for Roads and Bridges, Volume 12, Traffic Appraisal of Road Schemes, Section 2, Part I, Traffic Appraisal in Urban Areas*, The Stationery Office, London, 1996.

[28] D. R. King, A. Gold, K. B. Phillips, and D. A. Krauss, “Headway times on urban, multiple lane freeways,” in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, SAGE Publications Sage CA: Los Angeles, CA, vol. 66, 2022, pp. 1225–1229. DOI: 10.1177/1071181322661413.

[29] O. C. Puan, “Driver’s car following headway on single carriageway roads,” *Malaysian Journal of Civil Engineering*, vol. 16, no. 2, pp. 15–27, 2004. [Online]. Available: https://core.ac.uk/download/pdf/11784144.pdf.

[30] Q. Ge and M. Menendez, “Exploring the variance contributions of correlated model parameters: A sampling-based approach and its application in traffic simulation models,” *Applied Mathematical Modelling*, vol. 97, pp. 438–462, 2021. DOI: 10.1016/j.apm.2021.04.012.

[31] J. Kim and H. S. Mahmassani, “Correlated Parameters in Driving Behavior Models: Car-Following Example and Implications for Traffic Microsimulation,” *Transportation Research Record*, vol. 2249, no. 1, pp. 62–77, 2011. DOI: doi.org/10.3141/2249-09.

[32] S. Hausberger and D. Krajzewicz, “COLOMBO Deliverable 4.2: Extended Simulation Tool PHEM coupled to SUMO with User Guide,” German Aerospace Center (DLR), Tech. Rep., Feb. 2014. [Online]. Available: https://elib.dlr.de/98047/.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.215

Published: 29 June 2023

The State of Bicycle Modeling in SUMO

Aboozar Roosta[1][https://orcid.org/0000-0003-1023-8189], Heather Kaths[1][https://orcid.org/0000-0003-2554-8243], Mirko Barthauer[2][https://orcid.org/0000-0003-3177-3260], Jakob Erdmann[2][https://orcid.org/0000-0002-4195-4535], Yun-Pang Flötteröd[2][https://orcid.org/0000-0003-3620-2715], and Michael Behrisch[2][https://orcid.org/0000-0002-0032-7930]

[1] University of Wuppertal, Germany

[2] German Aerospace Center (DLR), Germany

Abstract. Microscopic traffic simulation tools provide ever-increasing value in the design and implementation of motor vehicle transport systems. Research and development of automated and intelligent technologies have highlighted the usefulness of simulation tools and development efforts have accelerated in recent years. However, the majority of traffic simulation software is developed with a focus on motor vehicle traffic and has limited capabilities in the simulation of bicycles and other micro-mobility modes. Bicycles, e-bikes and cargo bikes represent a non-negligible modal share in many urban areas and their impact on the operation, efficiency and safety of traffic systems must be considered in any comprehensive study. The Differentiation between different types of micro-mobility modes, including microcars, e-kick scooters, different types of bicycles and other personal mobility devices, has not yet attracted enough attention in the development of simulation software which creates difficulties in including these modes in simulation-based studies. On November 25th, 2022, members of the SUMO team at DLR organized a workshop to assess the state of bicycle simulation in SUMO, identify shortcomings and missing capabilities and prioritize the order in which bicycle traffic related features should be modified or implemented in the future. In this paper, different aspects of simulating bicycle traffic in SUMO are examined and an overview of the results of the workshop discussions is given. Some suggestions for the future development of SUMO emerging from this workshop, are presented as a conclusion.

Keywords: Microscopic Traffic Simulation, Bicycles, SUMO

1. Introduction

Bicycle traffic is distinct from car traffic in terms of the movement and interactions of individual road users and the aggregated traffic flow, which requires special consideration in microscopic traffic simulation. As of January 2023, the official documentation of SUMO suggests two methods for simulating bicycle traffic: modeling bicycles as “slow vehicles” or as “fast pedestrians”. The former option, simulating bicycles as slow vehicles, is the method that is widely used. With some modifications to the simulation environment, the same models that describe car traffic in SUMO are calibrated to simulate bicycle traffic. A desired speed and acceleration model captures the dynamics in free flow and a car-following model is used to simulate interactions with other road users on one-dimensional lanes. Lateral movement is simulated by dividing a single driving lane into multiple narrower sub-lanes in the longitudinal direction. The width of the road user and the sub-lanes dictates the number of sub-lanes that are “blocked” by a road user. Lane selection and lane change models are employed to determine the lateral position of the road user within one driving lane, making it possible to simulate passing within this lane. The addition of sub-lanes allows for much more realism in the simulation of bicycle and mixed traffic flows.

Although sub-lanes have vastly improved the simulation of bicycle traffic, it is still difficult to completely capture the unique dynamics [1], flexible movement [2], and less rule-based interactions of cyclists [3]. At the same time, in light of its low cost, low environmental impact, minimal space requirement and negligible noise production as well as increased public health through daily movement, the bicycle is emerging as a key to the "Verkehrswende" (English: transportation revolution). The modal split in many large German cities is already reaching 20% of the total number of trips [4] and there is a national goal of doubling the number of kilometers travelled by bicycle by 2030 in comparison to 2017 [5].

Given the increased need to simulate bicycle traffic, the SUMO development team recognized the need to address these shortcomings and identify opportunities to improve the simulation of bicycle traffic. To this end, an online workshop was held on November 25th, 2022 and past and current members of the SUMO development team, researchers, and SUMO users were invited to participate. The aim of the workshop was to analyze the status of bicycle modeling and simulation in SUMO, identify aspects of bicycle behavior that require improvement, and prioritize the development of new features to improve the simulation of bicycle traffic. In this paper, we present the results of the workshop and discuss the identified areas for improvement and proposed solutions.

At the beginning of the workshop, the SUMO team made a clear distinction between "qualitative" and "quantitative" features in the context of bicycle traffic modeling. Qualitative features refer to the aspects of bicycle traffic that should be accurately reflected in SUMO's modeling approach, such as turning behavior, following behavior, lateral movements and routing. On the other hand, the term quantitative refers to numerical validation, which assesses how well simulated bicycle traffic metrics align with real-world scenarios. To date, the SUMO team has not conducted any quantitative validation of the implemented features. It is worth noting that while writing this paper, the SUMO documentation [6] was frequently utilized to obtain more data about the issues discussed in the workshop.

In this paper, the terms "bicycle" and "cyclist" are used to refer to the simulated agent, depending on whether the emphasis is on the rider or the vehicle. Moreover, the term "car" is used to describe engine-powered vehicles, including cars, trucks, and buses.

The topics covered in the workshop are divided into four sections in this paper. In Section 2, methods for routing bicycle traffic through the simulated network are discussed. Section 3 examines the modeling and simulation of the longitudinal and lateral movements and interaction of cyclists. In Section 4, the simulation of traffic at intersections is examined. Section 5 focuses on the relevant topics in network design and road grade simulation. Finally, in Section 6, we present our concluding remarks and discuss proposed ideas for future feature development of SUMO, based on the outcomes of the workshop. Nevertheless, paragraphs following other text paragraphs are indented.

2. Routing

The SUMO package consists of several individual tools, each serving a specific purpose in the simulation. Among these tools are four routing algorithms: Dijkstra, A*, ALT and CH. Each of these algorithms is well-suited for certain scenarios. Routing for all road user types can be performed by travel time, effort, distance, or edge priority, offering flexible options to simulate the various factors that impact drivers' route choices. By default, routing in SUMO is done based on travel time minimization. "Effort" is a general term that refers to providing the routing algorithm with alternative weights or in other words optimization based on the alternative costs, such as pollutants (CO, CO_2, PM_X, HC and NO_X), fuel or electricity required to travel a given route, or noise generated in the process. These weights can be either constant or time-dependent.

In SUMO, bicycle routing uses the same routing algorithms and objectives as used for vehicle routing. However, there is an additional parameter called "--device.rerouting.bike-speeds" that allows the routing module to compute separate average speeds for bicycles. This parameter is disabled by default as it adds extra computing cycles and slows down the simulation. However, enabling it for scenarios where bicycle traffic is important is recommended as it results in more accurate routing of bicycle traffic. This feature can help account for the unique speed characteristics of bicycles, such as their lower speeds.

In addition to routing preferences and behavior, modeling bicycle traffic in SUMO should also take into account the specific characteristics of bicycle movement. Cyclists should be able to exhibit a preference for using dedicated bicycle infrastructure. This preference usually stems from safety concerns or enforced traffic laws. Besides using individual weights for routing, the only other possibility to affect the routing of bicycles is specifying individual speeds for bicycle lanes. Another frequently observed behavior that should be added to SUMO is the ability for cyclists to use adjacent lanes or edges under special conditions. This makes for a more flexible and realistic cyclist behavior modeling. However, practical implementation could be difficult due to the need to identify and avoid collisions in the course of these short lane/edge changes.

Other improvements could also be made to allow for the simulation of multimodal trips and more flexible bicyclist behavior. These include allowing for bicycle use in accessing public transport, to simulate pushing the bicycle across pedestrian crossings or carrying them on infrastructure where cycling is not allowed or possible, inventing new types of settings that allow for flexible switch between cycling and walking at any time during the trip and settings that allow for leaving a bicycle in some location and returning to pick it up at a later time. It should be noted that carrying certain bicycle categories like cargo bikes and bicycles with trailers may not be feasible thus the option to switch between walking and cycling has to be more fine-grained. These simulation settings will be in more demand as usage of shared bicycles grows and implementing these features will also facilitate incorporation of other personal mobility devices in SUMO in the future.

3. Following behavior and lateral movements

As discussed in the introduction, bicycle traffic is either modeled using adapted versions of car models for lateral and longitudinal movement and interactions (slow car option) or pedestrian models (fast pedestrian option). Because the former method is more frequently used by users of SUMO, the sub-lane modeling approach for following and lateral movement were focused on in the workshop. Although bicycle traffic tends to follow lanes in the intended direction of travel, cyclists are more flexible in this domain due to their smaller size and higher maneuverability in comparison to motorists. It is acknowledged that car-following, (sub-)lane selection and (sub-)lane changing models may not be able to fully capture the complexity of bicycle traffic behavior.

3.1 Car-following models

Car-following models determine the longitudinal acceleration in each simulation step based on the location, speed and other characteristics of individual vehicles by taking into account the vehicle directly ahead in the same (sub-)lane. Numerous car-following models have been formulated in the last 50 years and many of them are included in the SUMO package. The "carFollowModel" parameter specifies which car-following model is to be used in the simulation of the vehicle.

By default, SUMO utilizes the Krauß car-following model [7], which relies on three primary variables to determine a driver's behavior: the vehicle's own speed, the speed difference with the leading vehicle, and the distance to the leading vehicle. The Krauß model is designed to maintain a safe speed that ensures a minimum distance to the leading vehicle and prevents collisions. However, according to presentations from the SUMO team, this model produces noisy speed curves. To obtain smoother position and speed curves, it is recommended to use the Intelligent Driver Model (IDM) [8] instead of the Krauß model for bicycle traffic. In contrast to the Krauß model, the Intelligent Driver Model (IDM) takes into account the time headway, leading to smoother traffic flow curves. Furthermore, the extended IDM model (eIDM) [9] combines features of both the Krauß and IDM models to offer benefits of both models. When modeling bicycle traffic and car traffic in the same lanes, it is apparent that the two modes exhibit similar patterns, except that bicycle traffic densities are significantly higher (due to a lower minimum gap of 0.5 meters). Additionally, some lane-changing activities may alter the speed patterns of bicycles.

After discussions about the behavior and properties of the Krauß and IDM models, the question was raised as to whether these models provide an accurate representation of bicycle traffic, or whether entirely new models are needed to address their inaccuracies. The consensus was that these models are sufficient for the current stage of research, but modifications are necessary to account for distinct behaviors that are present in cycling but absent in motor vehicle traffic, such as side-by-side riding, which represents leisurely bicycle activities. It was suggested that quantitative efforts should be made to calibrate the Krauß and IDM models using real-world trajectory data and/or bicycle experiments. Currently, the following parameters are recommended in the SUMO User Documentation for modeling bicycle traffic. In order to provide a comparison, Table 1 displays a selection of significant default parameters for bicycles and cars in SUMO.

Table 1. Selected bicycle parameters defined in vClass=“bicycle” and “passenger”.

Parameter	vClass=“bicycle”	vClass=“passenger”
Minimum Gap	0.5 m	2.5 m
Maximum Acceleration	1.2 m/s2	2.6 m/s2
Maximum Deceleartion	3.0 m/s2	4.5 m/s2
Emergency Deceleration	7.0 m/s2	9.0 m/s2
Length	1.6 m	5.0 m
Maximum Speed	20.0 km/h	not limited (1000 km/h)

Observational and experimental data is needed to examine the qualitative properties of various car-following models, and to calibrate and validate them. During the calibration process, special attention should be given to reproducing exact macroscopic results to ensure better validity. Calibration and validation of the car-following models for bicycle traffic would be a first step in decoupling the models of car traffic and bicycle traffic. Ultimately, these models could be used to simulate other emerging micro-mobility modes, such as e-scooters. Although simulation of these modes can also be achieved through the utilization of the “slow pedestrian” approach, further enhancements to this approach are required. In particular, the integration of proper visualization techniques, as well as the inclusion of new movement models, would be imperative to improve the accuracy and comprehensiveness of the simulation.

3.2 (Sub-)Lane changing and lateral alignment

The lateral behavior and alignment of bicycles is controlled by lane selection and lane changing, both in terms of regular driving lanes and sub-lanes. When considering the motor vehicle simulation, vehicles may need to change lanes for various reasons, such as navigation or route-following, speed gain, cooperation, and following the rules. The lane changing model in SUMO determines lane choice on multi-lane roads and speed adjustments related to lane changing [10], and now supports four motives for lane changing: strategic, cooperative, and tactical lane changes, as well as the obligation to clear the overtaking lane. Changing lanes could also be triggered remotely by the TraCI interface.

In order to simulate more flexible lateral movements and differences in lateral positioning within a lane (bicycle traffic keeping to the right), and enable passing in one traffic lane, it is possible to divide the lanes into multiple sub-lanes. The sub-lane model is used to govern movement between multiple sub-lanes. The introduction of this model has made it possible to simulate common scenarios, such as cars overtaking two-wheeled vehicles in a single lane and multiple two-wheeled vehicles driving in parallel. This is particularly useful in scenarios where a significant amount of urban traffic consists of scooters and/or bicycles. For a full list, refer to the SUMO documentation on sub-lanes [11]. To ensure accurate simulation of behavior when the sub-lane model is activated, certain parameters must be set properly. In the sub-lane model, the car-following algorithm is adjusted to consider all vehicles occupying at least one sub-lane of the lane in which the subject vehicle is located. Additionally, the lane-changing model accounts for lateral alignment and safe lateral gaps, in addition to the four motivations for lane changes mentioned previously. However, the activation of the sub-lane model can significantly increase computation costs, and is therefore disabled by default to prevent slowing down simulations in cases where high resolution of lateral movements is unnecessary. An example of a mixed traffic link simulated using the sub-lane model approach is shown in Figure 1.

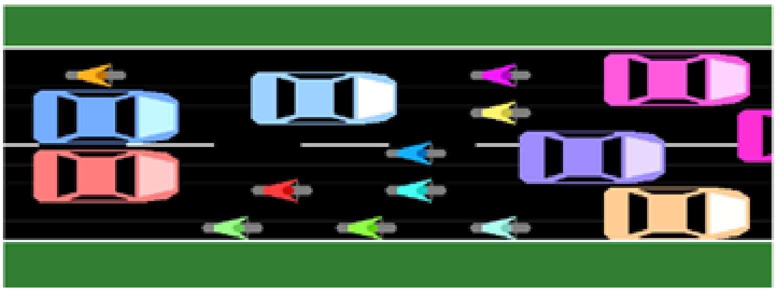


Figure 1. Application of sublane model to mixed traffic simulation [12].

In addition to the lane-change and sub-lane models, there is also a continuous lane-changing model available, which allows for more realistic lane-changing behavior by specifying the time it takes to complete a lane-change action. Compared to the sub-lane model, the continuous lane-changing model has significantly lower computation times. By default, without the sub-lane model, a single lane-change operation takes one simulation time step, which may not be entirely realistic depending on the vehicle's speed. The use of this model may be beneficial in simulating very narrow lanes where sub-lanes cannot be utilized. Incorporating a transition time during lane changes can make the maneuver more realistic.

The current implementation of the sub-lane model in SUMO has a limitation in that it cannot be enabled only for bicycle traffic while remaining disabled for car traffic. Having the sub-lane model enabled for car traffic creates more computation cost and adds little to the realistic representation of cars in the achieved flexibility of movement, except that they can move laterally to pass in the same driving lane. Nonetheless, the SUMO team has highlighted that the current implementation of the sub-lane model is beneficial for achieving smooth lane changes. Then, as for the car-following behavior, the adequacy of SUMO's lane change and sub-lane models for simulating bicycle traffic has been discussed. The consensus among the participants was that the simulated bicycle traffic in SUMO is too regular and lacks sufficient stochasticity. During the discussions, one proposal to enhance the realism of bicycle traffic simulation in SUMO was to introduce variations into the parameters that model bicycle traffic, such as incorporating a variable minimum gap for bicycles. The idea was well received by the SUMO team, who suggested that incorporating realistic variation bounds into the parameters, based on studies or publicly available data, would be preferable. It was also suggested that the regularity in the simulated traffic may be due to the vehicles, including bicycles, all being modeled with rectangular boxes instead of diamond shapes which are used in other simulation tools [12]. While the discussion favored adding variations to the parameters, the SUMO team also mentioned the possibility of adding a “diamond” shape to the bicycles’ parameters by a mathematical transformation. This transformation would be simple to implement and would not incur any additional computational costs. Diamond shape will lead to a more accurate simulation of more unconventional bicycle geometries of different types of cargo bikes and bicycles with a trailer. Another suggestion was to model the safety distance based on impatience or frustration of the cyclist which involves psychological factors and therefore requires more data.

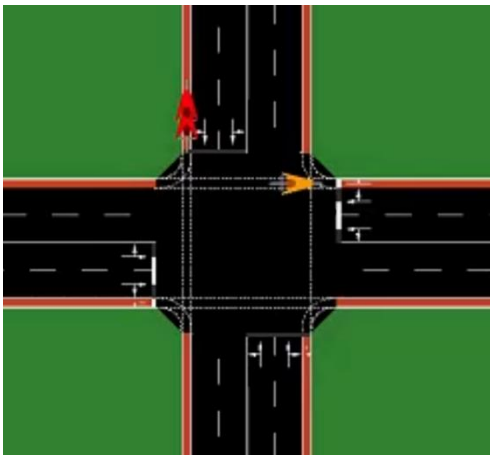


Figure 2. Demonstration of perceived conflict area by the red cyclists represented by SUMO team at the workshop.

4. Intersections

Intersections are crucial elements of the traffic network, connecting links and requiring conflicting streams of vehicles, bicycles, and pedestrians to interact. As such, simulating the behavior of road users at intersections requires special attention. Traffic signals, right-of-way rules, and the need to consider multiple types of vehicles and road users make intersections more complex than links and other segments of the road network. At intersections, the points/areas of conflict must be carefully simulated to accurately represent the behavior of all road users. This requires consideration of factors such as internal links that connect incoming and outgoing lanes, the speed at which vehicles approach and traverse the intersection, waiting times before entering and within the intersection, and outgoing flows that must be managed to avoid blocking the junction. Proper simulation of these factors can help ensure a more realistic and accurate representation of intersection behavior for all road users. Issues like direct and indirect turns for cyclists [13], adherence of cyclists to the internal links, and behavior in conflicting areas are important. For example, cyclists may be more likely to make indirect turns to navigate through crosswalks, and they may have different preferences at the intersection depending on their position relative to other road users.

The SUMO team emphasized the need for improved modeling of conflict areas, particularly in regards to bicycles. Currently, if a cyclist with right of way enters an intersection, other cyclists intending to cross the internal link must wait until the cyclist with right of way has completely left the intersection, even if there is sufficient time and space to safely cross. Figure 2 illustrates this issue, with the yellow cyclists representing the flow with right of way and red cyclists waiting due to the fact that all of the intersection is being considered a conflict area. It was suggested to alter conflict points to not be applicable to cyclists’ interactions with other cyclists, instead having cyclists slightly alter their speed and/or path to pass the conflict point without needing to wait. It was agreed that this phenomenon should be implemented as a qualitative feature. The quantitative aspect has to be further studied but there was an internal DLR study at Braunschweig which confirms that virtually no cyclist stops at a pedestrian crossing.

An additional discrepancy between actual bicycle traffic behavior at intersections and the simulation outputs produced by SUMO has been observed. While in SUMO cyclists adhere to the internal links very accurately, in reality cyclists roam more freely at intersections. One of the participants presented their work about the result of data collections at intersections in Munich and a comparison with simulated bicycle traffic in SUMO. Figure 3 compares the heatmap of speed and occupancy of bicycle traffic at one of the measured intersections.

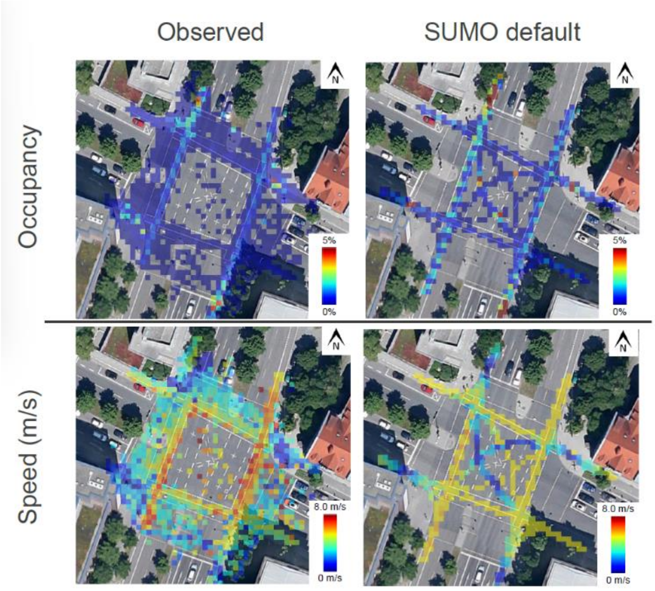


Figure 3. Comparison of real-world bicycle traffic occupancy and speeds with SUMO simulation [14].

The observed strict adherence to internal links in the simulated bicycle traffic may be partially attributed to the constant and unvarying nature of the bicycle traffic parameters, which was discussed in the previous section. Moreover, it is plausible that some cyclists may dismount from their bicycles while performing indirect left turns. Indirect left turns refer to the maneuver where cyclists continue straight at an intersection, subsequently making a 90-degree turn, before continuing in a straight direction again. Similar results have also been observed in other smaller intersections with one lane car traffic in this research. Various static and dynamic characteristics of the intersection likely influence cyclists' behavior. Traffic signals, car traffic volume and speed, physical separation between bicycle lanes and the roadway, whether islands are available at the intersection, and last but not least the geometry of the intersection are among these influencing factors.

A number of other issues were discussed briefly. Distinct signalization for bicycles and cars on the same connection should be allowed. Implementation of bicycles waiting ahead of motorized traffic could be further enhanced by making bicycles and motorcycles ignore the minimum gap of stationary cars when changing lanes. In scenarios where there are cars as well as bicycles, bicycle boxes do not work reliably. Furthermore, it is beneficial that connections with more than two internal lanes be allowed to accommodate complex junctions with multiple islands or two indirect left turns.

5. Network

Over the past two years, the SUMO team has placed emphasis on the development of network-related features, with significant advancements being made. Much of SUMO's code base had been originally developed with car-only networks in mind. As a response to this limitation, the team has been actively working on the implementation of multimodal network import capabilities, although there are challenges with regard to availability of openly available data.

The implementation of indirect left turns is now possible in SUMO. Indirect left turn is particularly favorable in larger intersections where the perceived risk associated with direct left turns for cyclists is higher. By default, direct turns are currently the default setting for bicycles. It was discussed whether indirect left turns should be the default action, and ultimately, it was decided to change the default to indirect left turns at intersections with bicycle lanes. This topic requires more observational research. A proposed idea is to enhance SUMO's network generation/import tools by advancing beyond the default option architecture and instead, making decisions based on the infrastructure's geometry and parameters. The rationale behind this suggestion is to address the limitations of open data, such as OpenStreetMap (OSM), which may contain incomplete infrastructure information. There was a discussion about the feasibility of predicting the existence of bicycle lanes based on available infrastructure data, like total road width and availability of parking lanes from OSM. However, implementing such an approach would require the development of models and studies on the correlation of different design parameters, in order to make more accurate estimations. It was decided that this feature is not a priority for development at this moment.

SUMO enables the inclusion of elevation data in the network. Currently, this information is used natively for electric vehicles, calculating emissions and in the extended Krauss car-following model. In the extended version of the Krauss car-following model, the maximum acceleration in each time step is reduced, based on the gradient of the road.

SUMO can import road network data from major data formats, some of which include elevation data. However, creating networks based on OSM is usually preferred as it offers its data under a free license. One of the downsides of using OSM is that there are not enough elevation data points in OSM to allow for reliable simulation of networks with varying road grades. Network grade and its variability has a considerable impact on mode choice and route choice, especially of active mobility road users [14].

SUMO includes a method for modeling and simulating electric powered vehicles, which includes additional variables that are not normally considered in a microscopic traffic simulation software such as the vehicle mass, the coefficient of drag, and the frontal surface area. This feature was developed in order to test different charging scenarios and technologies. As the physical relationships that describe the power required to move a bicycle are quite similar to those describing the power needed to move a car, this existing model will prove very valuable in developing an improved model for e-bikes in SUMO. The main difference lies in the combination of power supplied by the person and the electric motor. According to European law, an electric motor on a e-bikes can supply a maximum of 250W and can only provide power up to a speed to 25 km/h [15]. If a cyclist exceeds this speed or stops pedaling, the electric motor must immediately or gradually stop providing power to the bicycle.

6. Conclusions and suggestions

This workshop provided an opportunity for discussion on important aspects of bicycle traffic modeling and simulation in SUMO. During the workshop, researchers, users, and other interested parties contributed to discussions regarding the identification of critical features and priorities for future development. As a result of these discussions, certain issues were identified as requiring less effort to be implemented, indicating the possibility for their prioritization in the development process. These include the addition of diamond shapes to bicycle models in order to enhance their realism and add more variety to bicycle traffic simulations. Additionally, the possibility of assigning weights to different types of infrastructure to enable infrastructure-aware routing for bicycles was discussed. Such a system would give preference to certain infrastructure types, such as bicycle lanes, by assigning them lower weights in the routing process. Other low effort improvements include making indirect left turns the default behavior for cyclists, and enhancing bicycle boxes positioned in front of car traffic prior to intersections. In addition to the aforementioned topics, several other issues were identified as important and requiring additional attention. These include the development of concepts such as conflict areas for bicycles and pedestrians, as well as improvement of shared space simulations capabilities. There was also a recognition of the need to improve slope and elevation modeling and to enhance lateral movements in bicycle traffic simulations. Another general concern was isolating the cyclist behavior characteristics from sufficiently large empirical datasets and acquiring appropriate datasets needed for more accurate implementation of some proposed changes.

This workshop highlighted the need for models that can simulate unique bicycle traffic flow behavior. More improvements on top of the issues discussed in the workshop can be envisioned. It would be beneficial to researchers if SUMO allowed for easy implementation of force models and physics-based models. Software like NetLogo [16] provide this opportunity for simulating different models with subject agents. However, testing models in more realistic scenarios in the context of urban traffic networks is also required to accelerate research and development of these models. There is already a partnership with Jülich research center to integrate the JuPedSim pedestrian model [17] into SUMO and this suggestion could be considered during this integration. Integrating such models in SUMO could be an opportunity to create flexible frameworks that can accommodate similar models.

Much of the focus on modeling bicycle traffic has gone into the option “slow car”, meaning that modeling approaches for car traffic have been adapted, calibrated and applied to bicycle traffic. Far less attention has been placed on the option “fast pedestrian” and the use of social force like models in recreating bicycle traffic, both in this workshop and by researchers and developers. The formulation, calibration and validation of social force models for bicycle traffic could offer an important way forward in including the flexible behavior and the fluid interactions of cyclists in SUMO.

One possible enhancement would be augmenting the intended implementation of a diamond shape transformation for bicycle models with additional shapes that better reflect the wide range of cargo bicycles and bicycles with trailers that are currently available. As new types of micro-mobility vehicles continue to emerge, it is crucial to anticipate their use cases and adapt the SUMO software accordingly, in order to minimize the need for extensive reworking of the codebase in the future. Another suggestion is the implementation of shared mobility in the form of stations for shared bicycle providers and bicycle parking with high capacity to improve simulation of seamless pedestrian-bicycle trips.

Data availability statement

This paper is not based on data.

Author contributions

The authors contributed to this paper in the following ways: All authors contributed to the conceptualization and methodology. A. R. was responsible for writing the original draft of the manuscript. H. K., Y. P., M.B. and J.E. contributed to reviewing and editing the manuscript.

Competing interests

The authors declare that they have no competing interests.

Acknowledgement

The content of this paper is based on a workshop that was organized by DLR and held online on November 25th, 2022.

References

[1] A.-S. Karakaya, K. Köhler, J. Heinovski, F. Dressler and D. Bermbach, "A Realistic Cyclist Model for SUMO Based on the SimRa Dataset," in 2022 20th Mediterranean Communication and Computer Networking Conference (MedComNet), 2022.

[2] H. Twaddle, T. Schendzielorz and O. Fakler, "Bicycles in urban areas: Review of existing methods for modeling behavior," Transportation research record, vol. 2434, p. 140–146, 2014.

[3] H. Kaths, "Cyclists' interactions with other road users from a safety perspective," Cycling, p. 187, 2022.

[4] C. Nobis, "Mobilität in Deutschland- MiD: Analysen zum Radverkehr und Fußverkehr," 2019.

[5] Bundesministerium für Verkehr und digitale Infrastruktur (BMVI), Nationaler Radverkehrsplan 3.0: Fahrradland Deutschland 2030, Berlin, 2021.

[6] "SUMO User Documenation," 10th Feb. 2023. [Online]. Available: https://sumo.dlr.de/docs/index.html.

[7] S. Krauß, "Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics," 1998.

[8] A. Kesting, M. Treiber and D. Helbing, "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity," Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 368, p. 4585–4605, 2010.

[9] D. Salles, S. Kaufmann and H.-C. Reuss, "Extending the intelligent driver model in SUMO and verifying the drive off trajectories with aerial measurements," in SUMO Conference Proceedings, 2020.

[10] J. Erdmann, "SUMO's lane-changing model," in Modeling Mobility with Open Data: 2nd SUMO Conference 2014 Berlin, Germany, May 15-16, 2014, 2015.

[11] "SublaneModel," 10th Feb. 2023. [Online]. Available: https://sumo.dlr.de/docs/Simulation/SublaneModel.html.

[12] G. Falkenberg, A. Blase, T. Bonfranchi, L. Cosse, W. Draeger, P. Vortisch, L. Kautzsch, H. Stapf and A. Zimmermann, "Bemessung von Radverkehrsanlagen unter verkehrstechnischen Gesichtspunkten," Berichte Der Bundesanstalt Fuer Strassenwesen. Unterreihe Verkehrstechnik, 2003.

[13] S. Amini, H. Twaddle and A. Leonhardt, "Modelling of the tactical path selection of bicyclists at signalized intersections," in Transportation Research Board 95th Annual Meeting, 2016.

[14] A. Meister, K. W. Axhausen, M. Felder and B. Schmid, "Route choice modelling for cyclists on dense urban networks," Available at SSRN 4267767, 2022.

[15] Regulation (EU) No 168/2013, 2013.

[16] U. Wilensky, "NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Compter-Based Modeling, Northwestern University, Evanston, IL.," 1999.

[17] A. U. Kemloh Wagoum, M. Chraibi and G. Lämmel, "JuPedSim: an open framework for simulating and analyzing the dynamics of pedestrians," 2015.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.216

Published: 29 June 2023

Coping with Randomness in Highly Complex Systems Using the Example of Quantum-Inspired Traffic Flow Optimization

Maria Haberland[1] [https://orcid.org/0009-0009-4383-8632] and Lars Hohmuth[1][https://orcid.org/0009-0004-0360-7606]

[1] Fujitsu, Germany

Abstract. Developing new solutions to complicated large-scale problems typically requires large-scale numerical simulation. Therefore, traffic simulations often run against randomized simulations instead of real-world traffic situations. This paper demonstrates a method to calculate the statistical significance of numerical simulations and optimizations in the presence of numerous random variables in complex systems using one-sided paired t-tests. While the paper covers a specific Fujitsu traffic-optimization project which uses SUMO for simulating the traffic situation, the method can be applied to many similar projects where a complete investigation of the solution space is not feasible due to the size of the solution space.

Keywords: Statistics, Traffic simulation, Optimisation

1. Introduction

In 2020 the Hamburg Port Authority (HPA) initiated a traffic innovation project called MOZART. Its goal was to improve the car and heavy goods vehicle (HGV) traffic flow throughout the 30km road network of the port region by using an advanced digital twin, microscopic traffic simulations, and the Fujitsu Digital Annealer Unit (DAU) [1] to create a globally optimized signal plan for all 35 intersections once a minute. This signal plan will help the Port of Hamburg to achieve their part in the UN Sustainable Development Goals, specifically regarding the subgoals of “climate action by reducing pollution”, “responsible production and consumption by streamlining transport of goods”, “creating sustainable cities and communities by reducing traffic induces stressors”.

Fujitsu developed a solution concept that uses real time traffic simulation combined with multiple computationally generated alternative signal plans for each intersection. Using these the solution simulates the traffic between the intersections in parallel for all possible combinations of signal plans between adjacent intersections to calculate the coefficients of a stress function. This function is then written as a polynomial of quadratic order in binary variables called a Quadratic Unconstrained Binary Optimization (QUBO) suitable as input for the Digital Annealer. The solution then uses the Digital Annealer to find a global optimum of this stress function. For more details on the approach, see Traffic management through traffic signal control by Quantum-Inspired optimization. [2]

After validating the basic viability of the approach, this project was turned over to our team to develop a solution which can be run 24/7/365 in cities.

2. The TraFO System

The production application we are developing is called TraFO (Traffic Flow Optimization) and consists of a scalable ensemble of containers:

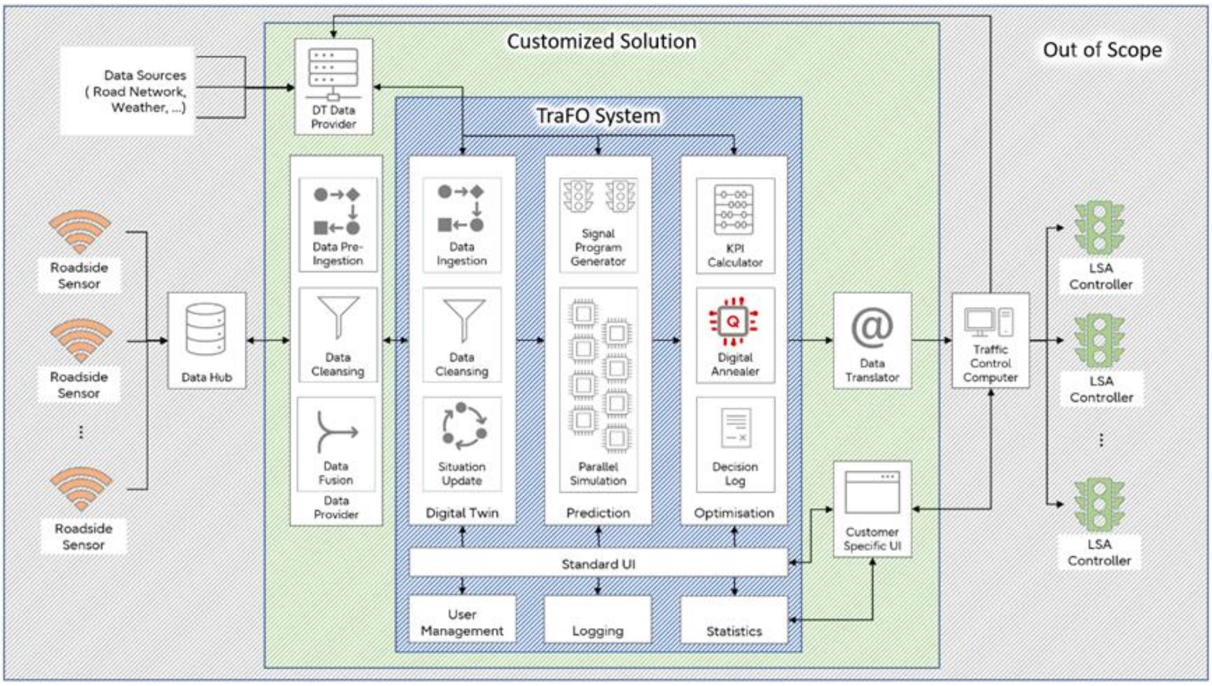


Figure 1. Structure of TraFO Optimization Platform

- The TraFO container sits at the center of the application. It performs the setup of the optimization, calculates the QUBO, calculates the KPIs, orchestrates the other containers, and provides the UI.
- The digital twin represents the real-world traffic. The digital twin is constructed using Vissim or SUMO networks and run in an instance of Vissim and SUMO [3] using traffic flow data for different time slots. We are using the available interface of these applications to collect the needed data for evaluation.
- Functions to do data ingestion and data cleaning for sensor data is also included into the digital twin to ensure a high data quality for the traffic simulations.
- Multiple instances can be used to run alternative scenarios in individually configured digital twins.
- The Signal Program Generator container calculates and preselects alternative traffic light programs to the TraFO for use by the short-term simulations. It also checks the compliance of the traffic light programs with legal and regulatory requirements. Currently, the Signal Program Generator implements the German regulatory requirements laid out in the Richtlinien für Lichtsignalanlagen (RiLSA), Edition 2015 [4]. This can be expanded by other regulatory requirements like MUTCD for the United States of America.

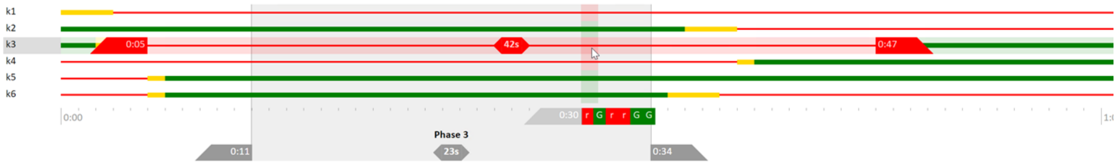
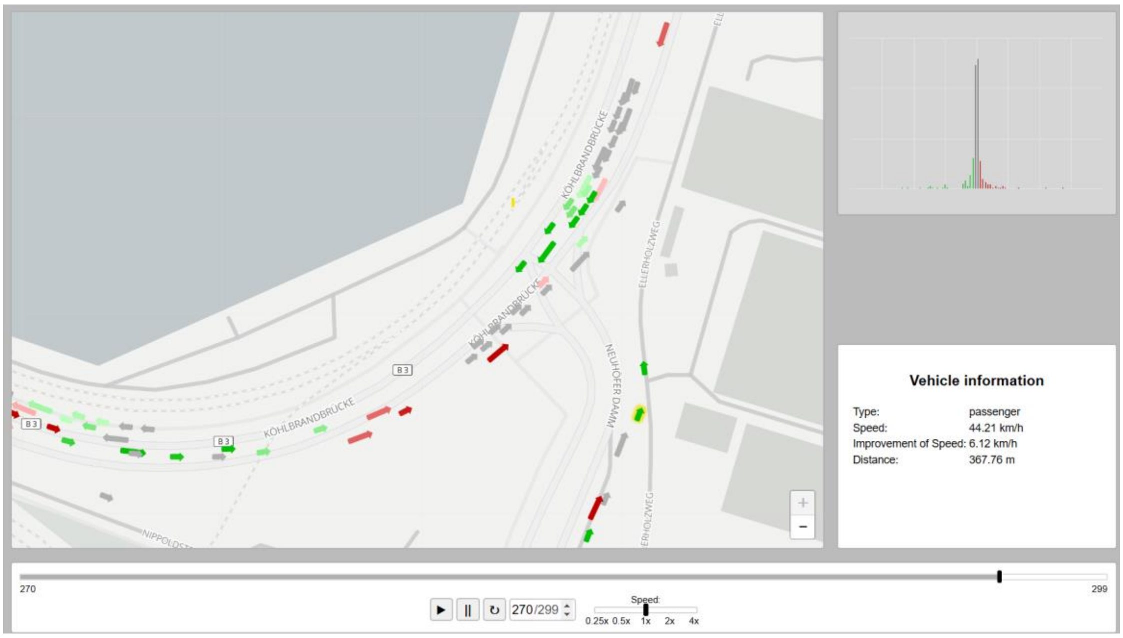


Figure 2. Visualization of Traffic Controller Programs

- To archive the required performance, the 125 short term simulations for each step are distributed across multiple instances, at least 10, running SUMO using libsumo in a containerized environment. The ideal number of instances depends on the available hardware, typically one per available core.
- The Digital Annealer calculates the optimization result.
- A standard UI is provided to consolidate the results and configure the different components. Customer specific UI, for example to display customer specific KPIs or add external data, can be added



Vehicle information

Type:	passenger
Speed:	44.21 km/h
Improvement of Speed:	6.12 km/h
Distance:	367.76 m

Figure 3. UI for Digital Twin Visualisation

- We are using a containerized MongoDB to store the road network and other input data as well as the simulation results, vehicle trajectories and KPIs.

In each optimization cycle, the TraFO container:

1. Collects the up-to-date traffic situation and signal program from the Digital Twin.
2. Passes them to the short-term simulations together with the possible signal programs from the signal program generator and starts the short-term simulations.
3. Builds the QUBO with the results of the short-term simulations.
4. Passes the QUBO to the DAU.
5. Collects the optimal signal programs for each signal head from the DAU.
6. And passes them to the long-term simulation at the start of the next cycle.

2.1 Randomness in TraFO

Since we cannot develop an application in live traffic, we must rely on traffic simulation tools to create a digital twin. In our project we use two different simulation tools: SUMO and PTV Vissim. Both tools use random number generators (RNG) to place vehicles and simulate driver behavior in traffic. Both also let users specify random seeds to reproduce simulation runs.

Additionally, in TraFO, the generation of signal program alternatives and the short-term simulation use multiple random number generator instances to decouple different simulation aspects, including

- randomness when loading vehicles (vehicle type distributions, speed deviations, ...)
- probabilistic flows
- vehicle driving dynamics

and more.

Changes to the random seeds can lead to widely varying traffic flows as shown below. Depending on the traffic flow, one and the same alternate signal plan can be highly effective, or highly detrimental to the flow of traffic.

In our project, we came across the following types of random number generators:

1. Pseudo Random Number Generators in SUMO
   Sumo uses the Mersenne Twister [5] is a general-purpose pseudorandom number generator developed in 1997 by Makoto Matsumoto (松本 眞) and Takuji Nishimura (西村拓士). This algorithm is widely used by commercial software like Microsoft Excel, SAS, SPSS, and Matlab as well as standard libraries like the standard C++ library, CUDA, and the NAG Numerical Library.
2. Pseudo Random Number Generators in Vissim
   PTV Vissim is a proprietary commercial software, so while it uses multiple random number generators to vary the patterns of stochastic assignments and traffic signals, the exact type could not be determined from the technical documentation.

3. How Big is the Influence of Random Numbers?

The number of possible combinations of random seeds between the long-term simulation and optimization is so large (~ $3.4*10^{38}$), that exhaustive sampling is impossible (~10^{28} years at 1000 simulations per second). Additionally, we are using about 10 parameters to tune the short-term simulation, which further inflates the search space.

3.1 Measuring Traffic Quality

To score the simulation runs we chose four key performance indications (KPI) for comparing the different runs:

- Average Speed: at every simulation step, the average speed of all vehicles in the network is calculated and then averaged again over the run time of the simulation. A higher value is considered better.
- Number-Of-Vehicles: at every simulation step the number of vehicles in the network is counted. The KPI is the average of all these sums. A lower value is considered better, because when having the same number of vehicles getting into the network that means more vehicles already leaving the network earlier.
- Deceleration: at every simulation step, the average deceleration of all vehicles in the network is calculated. Deceleration in this case means, that only negative acceleration values were considered, using 0 for positive accelerations. This is used, to get value more fluent traffic, as stop and go creates much more fuel consumption and CO_2 production than a slower but steady traffic. At the end the values were averaged again over the run time of the simulation. A lower value is considered better.
- Traffic-Jam: at every simulation step, the vehicles which drive at less than 5 km/h are counted. The Traffic-Jam KPI is the average of all these sums. A lower value is considered better.

All KPIs are only calculated after a 900-second start-up phase without optimizations which is needed to populate each part of the network with enough vehicles to generate valid data.

To understand the impact the randomness in simulating traffic, we started with an analysis of baseline simulations of the same situation using different random seeds.

Further KPIs can be added to the list as long as they are numerical and metric.

3.2 Baseline Comparisons

To get an impression of the difference between simulations of the same initial situation using different random seeds, we have selected two simulations in our application, showing their KPIs and the values for average speed over simulation time: once initialized with 9299 as the random seed and once with 9340.

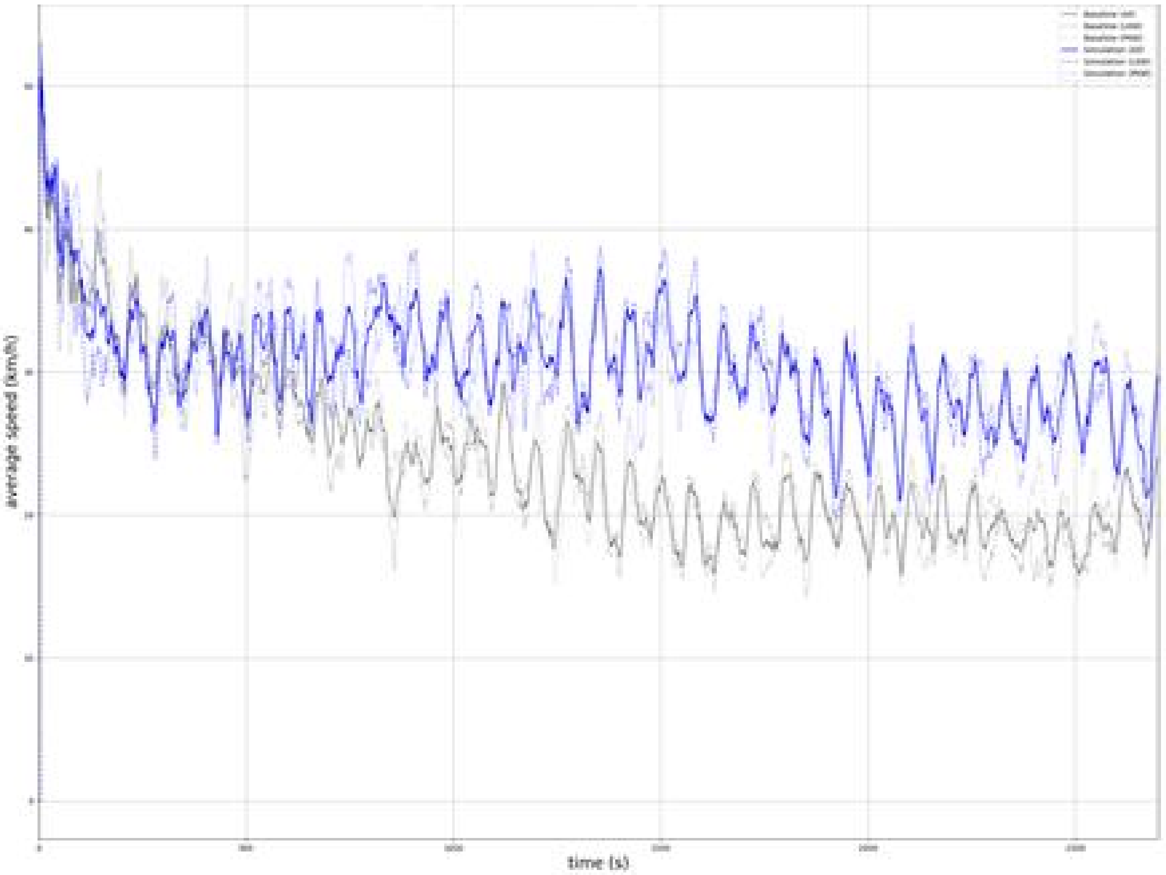


Figure 4. Comparison of average speed over simulation time of two simulations

The average speed of these two simulations has a ratio of almost 2:3 just because of the randomness of the input data for the simulation. Given that an average improvement of about 10% would be considered a huge success in the real world, any optimization effect would be buried by the differences created by the choice of the random seed.

Table 1. KPI of baseline 9299 and 9340.

Simulation Seed	Average Speed	Number Of Vehicles	Deceleration	Traffic Jam
9299	29.36	203.2	-0.791	65.1
9340	20.51	289.7	-0.680	143.5

To get a better impression, we created 50 baseline simulations and generated a boxplot (see Fig. 5) from the generated data.

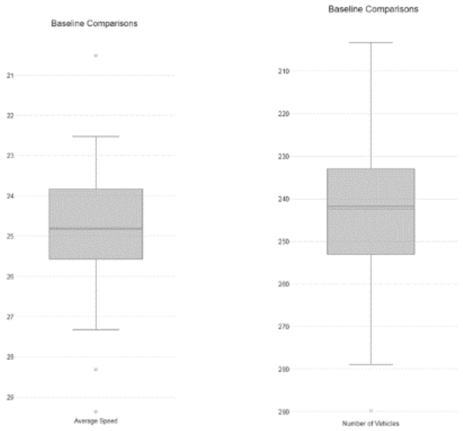


Figure 5. Boxplot for baseline simulations

As one can see, the measured KPIs vary widely even though we haven't used the digital annealer and haven't generated any optimization or changed any traffic light program. The big question is now: If we already have such big differences in the baseline simulations, how could we prove that our optimization creates better results? Just running some optimizations for some randomly picked random seeds wouldn't prove anything, even though one might get a decent hunch from running a few thousand simulations.

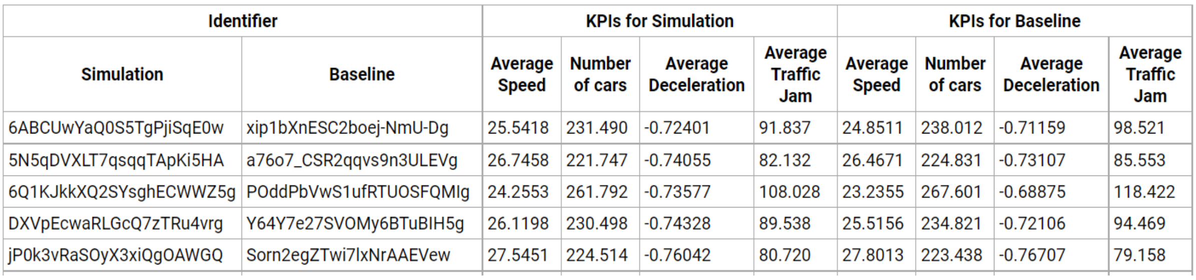
4. Are We Really Improving Traffic?

It would already be a huge success if we could improve the traffic flow by about 10%. As shown above, two baseline simulation can already have a bigger difference using the same pool of traffic light programs, caused just by variations in simulating the behavior of drivers. As each round of optimization creates a change in the overall simulated system, the vehicle behavior will be different over time. For example, if without optimization a car would have been waiting at a crossing, it is possible that the optimization has it now already driving past the intersection. This will influence the later simulation of other vehicles as well.

Therefor it is not sufficient to just pick a few simulations and compare the baseline with the optimized one, but instead use a more sophisticated approach.

4.1 Medical Science to the Rescue

Our solution for this dilemma was to look at other discipline in sciences. How do they handle such uncertainties? After reviewing some approaches, we selected methods used in medical sciences and pharmaceutical testing [6]. These seem suitable, because the traffic simulation



Identifier		KPIs for Simulation				KPIs for Baseline			
Simulation	Baseline	Average Speed	Number of cars	Average Deceleration	Average Traffic Jam	Average Speed	Number of cars	Average Deceleration	Average Traffic Jam
6ABCUwYaQ0S5TgPjiSqE0w	xip1bXnESC2boej-NmU-Dg	25.5418	231.490	-0.72401	91.837	24.8511	238.012	-0.71159	98.521
5N5qDVXLT7qsqqTApKi5HA	a76o7_CSR2qqvs9n3ULEVg	26.7458	221.747	-0.74055	82.132	26.4671	224.831	-0.73107	85.553
6Q1KJkkXQ2SYsghECWWZ5g	POddPbVwS1ufRTUOSFQMlg	24.2553	261.792	-0.73577	108.028	23.2355	267.601	-0.68875	118.422
DXVpEcwaRLGcQ7zTRu4vrg	Y64Y7e27SVOMy6BTuBIH5g	26.1198	230.498	-0.74328	89.538	25.5156	234.821	-0.72106	94.469
jP0k3vRaSOyX3xiQgOAWGQ	Sorn2egZTwi7lxNrAAEVew	27.5451	224.514	-0.76042	80.720	27.8013	223.438	-0.76707	79.158

Figure 6. Raw statistical data generated by simulation

programs essentially model human behavior and must deal with large solution spaces and incomplete knowledge of the "participants" in large scale medical studies as well.

When doing new drug development or studies for new therapies, biometric statistical methods are used. Similar statistics are also used in human studies in psychology, for example when evaluating therapies or clustering human behavior. So, we took a detailed look at how they solve the problem of distributed data in randomized testing.

Any study normally consists of three steps: the design of the study, the execution, and the evaluation. We will use the same three steps for our problem.

4.2 Design of the Study

In the design phase, one answers the following questions:

- Which hypothesis(es) do we want to prove?
- Which type of study do we conduct?
- What aspects will we measure during execution?
- How will we evaluate the generated data?
- How many "participants" do we need?

The hypothesis we want to prove is that using the optimization with the QUBO and the digital annealer results in better traffic flow. In other words, we want to check whether the traffic in the optimized simulation is more fluid than the baseline simulation. In statistics, one uses an inverted null hypothesis, which is: The traffic flow after optimization is not more fluid than the traffic flow in the baseline simulation.

For the other prescribed steps, we build an analogy to medical studies. When we look at our generated data, we see that we can run a baseline simulation and an optimization simulation using the same random seeds. This is reminiscent of "twin studies" in medical sciences. So, we ran baseline simulations and optimization simulations using the same random seed for our testbed and a fixed set of parameters for our optimization and calculated the above mentioned KPIs.

To test whether one set of results were better than the other, we chose a one-sided paired t-test [6], a popular method to compare data of twin studies in medical biometry.

One initial step is to calculate how many "participants" are needed for a statistically significant result. Using standard online tools, a Cohens d of 0.2 (the minimal feasible value) required 156 "participants"[7], [8]. This tells us that we had to run at least 312 (156 * 2) simulations to achieve a useful result. Due to the parallel execution of the simulations in TraFO, this could be done in an acceptable length of time.

4.3 Execution of the Study

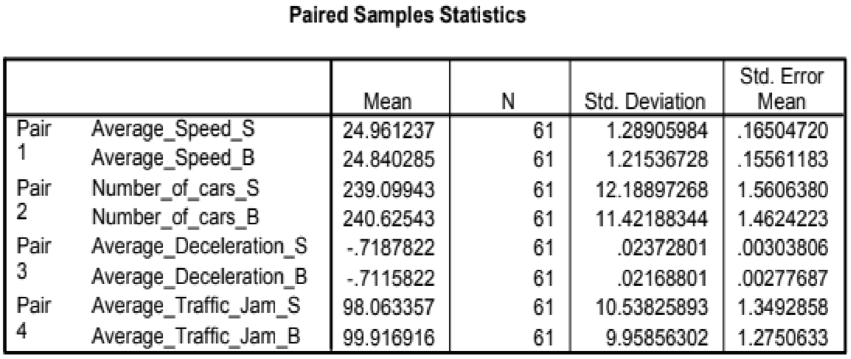
For the execution we had to run a lot of simulations and collect the data (see Figure 6). We had previously implemented a batch mode TraFO, which allows us to run a defined number of pairs of simulations (baseline and optimization) for a specific situation using different randomly generated random seeds. All data is saved in a MongoDB database and can be processed later.

Because running simulations in our optimized version of SUMO is much faster than in Vissim, we focused our first evaluation runs on SUMO simulations. Using SUMO, we can run a single simulation of a 45-minute traffic flow in about 5 minutes with 11 parallel simulation threads on a Fujitsu CELSIUS J5010 (Intel Core i7-10700, 32GB RAM) so we would have needed about one day to run all the simulations we needed.

To save time, we separated the simulation runs into subsets, so that we could start evaluating data even while additional simulations were still running. For our first run we chose a set of optimization parameters which caused some improvements in traffic flow in randomly selected trial runs.

4.4 Statistical Evaluation of the Study

For the first evaluation we use SPSS to calculate the statistical results.

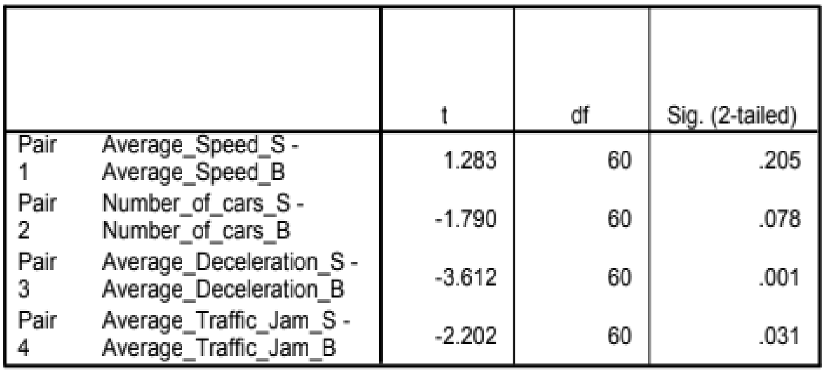


Paired Samples Statistics

		Mean	N	Std. Deviation	Std. Error Mean
Pair 1	Average_Speed_S	24.961237	61	1.28905984	.16504720
	Average_Speed_B	24.840285	61	1.21536728	.15561183
Pair 2	Number_of_cars_S	239.09943	61	12.18897268	1.5606380
	Number_of_cars_B	240.62543	61	11.42188344	1.4624223
Pair 3	Average_Deceleration_S	-.7187822	61	.02372801	.00303806
	Average_Deceleration_B	-.7115822	61	.02168801	.00277687
Pair 4	Average_Traffic_Jam_S	98.063357	61	10.53825893	1.3492858
	Average_Traffic_Jam_B	99.916916	61	9.95856302	1.2750633

Figure 7: Mean and standard derivation for each data set

When running our first evaluation using SPSS, we were surprised to see that all averages were slightly better in the optimization simulations than in the baseline simulations already (see Figure 7). Furthermore, we already got statistically significant metrics (Sig < 0.05) for the Traffic-Jam KPI and a statistical trend (Sig < 0.1) for the Number-of-Cars KPI (see Figure 8). The results can be reproduced using the Excel implementation for the t-Test [9].



		t	df	Sig. (2-tailed)
Pair 1	Average_Speed_S - Average_Speed_B	1.283	60	.205
Pair 2	Number_of_cars_S - Number_of_cars_B	-1.790	60	.078
Pair 3	Average_Deceleration_S - Average_Deceleration_B	-3.612	60	.001
Pair 4	Average_Traffic_Jam_S - Average_Traffic_Jam_B	-2.202	60	.031

Figure 8. t-test results

This shows that even with the first selected optimization configuration and our simple KPIs, we can show that our algorithm improves the traffic flow. Calculating Cohens d we can even show that the results are not only significant but also show a statistical effect. We expect that we will find other optimizations with even larger effects in the future.

Because we want to run these steps with more optimization configurations in future, we added an implementation of the one-sided pair-test to TraFO using the SciPy library for Python. This will allow us to run more tests faster and displaying the results inside TraFO.

5. Conclusion and Outlook

Because of the strong dependency of complex systems on pseudo random number generator seeds and initial conditions, it is often hard to separate effects of deliberate optimizations in digital simulations from spurious changes due to varying initial conditions. This becomes a real issue if the simulation of the real-world system relies on multiple independent simulations using random number generators. Traffic flow simulations often shows this behavior.

This paper shows that using biometric statistical evaluation of a large number of simulations for a set of simulation parameters can help prove that a specific computational approach is working and can also eliminate unsuitable combinations of parameters. Compared to the common approach to verify the effectiveness of new computational approaches in traffic simulation by “running a few experiments” or “eyeballing it”, it increases the confidence in the accuracy of the simulations and optimizations and makes it easier to explain the approach and its benefits to future users with hard statistical evidence.

Additionally, these measures - for example Cohens d - will help identify the best set of optimization parameters early in research and simulation projects, so researchers and traffic engineers can home in on strategies that provide the highest added value.

We will use one-sided paired t-tests in more simulations for our scenario with different configurations to evaluate the effects of different tweaks to the optimization algorithm. Our experiments already showed that it is possible to get a statistically significant improvement in our scenario at the port of Hamburg using the digital annealer. We are already using this approach to find the best parameter sets for Hamburg as well as for other situations.

For example, we have now a tool kit to check quickly, whether more complex scoring algorithms or new optimization algorithm leads to better solutions. We already have a set of different methods for generating QUBOs for the digital annealer which we now can compare to each other quantitatively.

In the future we want to use this approach in other projects, which also simulate scenarios based on human behavior or pseudo random number generators, including large scale SUMO simulations. We also want to encourage other SUMO users to use it as well.

We also will continue to explore additional statistical tests for more detailed analysis with the Wilcoxon signed-rank test as the next candidate.

Data availability statement

Due to the nature of the research, due to commercial supporting data is not available.

Author contributions

Maria Haberland: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Visualization, Writing. **Lars Hohmuth**: Conceptualization, Project administration, Validation, Visualization, Writing.

Competing interests

The authors declare that they have no competing interests.

Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

Acknowledgement

We would like to thank the whole development team for the TraFO solution for building up this wonderful platform, that allows us to create this work. Especially we would like to thank Peter Marschke for his work providing services to use the SUMO interfaces and his support using them. Furthermore, we would like to thank Frank Hambach for his work on the traffic light program generation, Hannah Berlau for her help creating beautiful UIs and Mike Maaßen for his development of further functions.

References

1. Aramon, M., Rosenberg, G., Valiante, E., Miyazawa, T., Tamura, H., & Katzgraber, H., Physics-Inspired Optimization for Quadratic Unconstrained Problems Using a Digital Annealer. Front. Phys., 05 April 2019, doi: https://doi.org/10.3389/fphy.2019.00048.
2. F. Schinkel, I. Schwende, R. Schade, E. Cerny, M. Fellendorf, Traffic management through traffic signal control by Quantum-Inspired optimization. 27th ITS World Congress, Hamburg, Germany, 2021.
3. Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, Evamarie Wießner, Microscopic Traffic Simulation using SUMO. IEEE Intelligent Transportation Systems Conference (ITSC), 2018, doi: https://doi.org/10.1109/ITSC.2018.8569938.
4. Forschungsgesellschaft für Straßen- und Verkehrswesen. Richtlinien für Lichtsignalanlagen - Lichtzeichenanlagen für den Straßenverkehr. Berlin: FGSV. 2015.
5. M. Matsumoto, T. Nishimura, Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Transactions on Modeling and Computer Simulation, 1998, doi: https://doi.org/10.1145/272991.272995.
6. J. Hedderich, L. Sachs, Hypothesentest, Angewandte Statistik. Berlin, Heidelberg: Springer Spektrum, 2018.
7. J. Frost, Cohens D: Definition, Using & Examples, Statistics by Jim: https://statisticsbyjim.com/basics/cohens-d/, (04/2023).
8. Hemmerich, W., Statistik Guru: Cohen's d für den gepaarten t-Test berechnen, Statistics Guru: https://statistikguru.de/rechner/cohens-d-gepaarter-t-test.html, (04/2023).
9. B. Walther, T-Test bei abhängigen Stichproben in Excel durchführen, https://bjoern-walther.com/t-test-bei-abhaengigen-stichproben-in-excel/, (04/2023).

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.217

Published: 29 June 2023

The Effects of Route Randomization on Urban Emissions

Giuliano Cornacchia[1 2], Mirco Nanni[2], Dino Pedreschi[1], and Luca Pappalardo[2]

[1]University of Pisa, Pisa, Italy
[2]ISTI-CNR, Pisa, Italy

Abstract: Routing algorithms typically suggest the fastest path or slight variation to reach a user's desired destination. Although this suggestion at the individual level is undoubtedly advantageous for the user, from a collective point of view, the aggregation of all single suggested paths may result in an increasing impact (e.g., in terms of emissions). In this study, we use SUMO to simulate the effects of incorporating randomness into routing algorithms on emissions, their distribution, and travel time in the urban area of Milan (Italy). Our results reveal that, given the common practice of routing towards the fastest path, a certain level of randomness in routes reduces emissions and travel time. In other words, the stronger the random component in the routes, the more pronounced the benefits upon a certain threshold. Our research provides insight into the potential advantages of considering collective outcomes in routing decisions and highlights the need to explore further the relationship between route randomization and sustainability in urban transportation.

Keywords: Routing, Route Randomization, Traffic Simulation, Urban Emissions

1 Introduction

Vehicular mobility is pivotal in global greenhouse gas emissions and determining the urban environment's sustainability [1]. Emissions of CO_2 from road vehicles were 1.57 billion metric tons in 2012, accounting for 28% of US fossil fuel CO_2 emissions [2], causing climate changes, heat islands [3] and health-related risks [4]. Traffic congestion, a significant source of CO_2 emissions in urban environments [5], may arise due to (unintended) drivers' miscoordination, which may be exacerbated nowadays by the massive use of GPS navigation systems. Typically delivered as phone apps, these systems suggest the fastest path to reach a user's desired destination. Although this suggestion is undoubtedly advantageous for the user, especially when exploring an unfamiliar city, the aggregation of all single suggested paths may result in an increasing urban impact (e.g., in terms of emissions). Indeed, a recent work shows that the higher the fraction of vehicles following these apps' suggestion, the higher the urban emissions [6]. Several alternative routing algorithms have been introduced, which typically slightly randomize the fastest path to increase route diversity [7]–[11]. However, it still needs to be determined to what extent route diversification can help reduce emissions and traffic congestion in urban environments.

This paper provides a method to assess the impact of route randomization on the urban environment using the mobility simulator SUMO and duarouter. We investigate the impact of randomized routes on CO_2 emissions and travel time in Milan, Italy. By changing the fraction of randomized vehicles for different degrees of randomization, we examine how the distribution of emissions across the roads and the vehicles' travel time change. We find that an optimal randomization degree exists, leading to a 15% reduction in CO_2 emissions and an 18% reduction in travel time, compared to the baseline case in which there is no path randomization. In particular, the CO_2 distribution entropy increases with the degree of randomization, leading to a more evenly distributed emission on the road network. Our study provides valuable insights into the potential benefits of incorporating randomness into route recommendations as it may increase sustainability in transportation networks. We provide the code and the link to the data to reproduce our study at https://bit.ly/route_randomization_sumo.

2 Related Work

Computing the shortest (or fastest) path between two given locations in a road network is a largely addressed problem in mobility research [12]. The fastest path is the one that minimizes the travel time to reach a desired destination. Although this suggestion at the individual level is undoubtedly advantageous for the user, from a collective point of view, the aggregation of all single suggested paths may result in an increasing impact (e.g., CO_2 emissions) [6].

Different works have focused on alternative routing [7], typically formalized as the k-shortest path problem [13], [14], which aims to find the $k > 0$ shortest paths between an origin and a destination in a network. Cheng et al. [8] demonstrate how, in most practical cases, path diversification is crucial to solving the k-shortest path problem since the generated paths have 99% overlap in terms of road edges. Suurballe [15] proposes another method to generate k-shortest disjointed paths, in which the route appears considerably diverse from the optimal path and the travel time and path length increase considerably. In between the k-shortest path and k-shortest disjoint paths lie several approaches that are a good tradeoff between the two approaches.

Liu et al. [9] propose the k-Shortest Paths with Diversity (kSPD) problem, defined as top-k shortest paths that are the most dissimilar with each other and minimize the paths' total length. Given the kSPD problem, Chondrogiannis et al. [10] propose an implementation and a study of the k-Shortest Paths with Limited Overlap (kSPLO), seeking to recommend k-alternative paths that are as short as possible and sufficiently dissimilar based on a similarity threshold defined by the user.

Chondrogiannis et al. in [11] formalize the Dissimilar Paths with Minimum Collective Length (kDPML) problem based on the definition proposed by Liu et al. in [9]. Given two locations on a road network, they compute a set of k paths containing sufficiently dissimilar routes and the lowest collective path length among all sets of k sufficiently different paths.

Cheng et al. [8] generate alternative routes by considering the road network as a weighted graph and distorting the edge weights. They iteratively compute the optimal path, applying a penalty on each edge of the optimal path found in the previous iteration.

Another technique used to generate alternative routes is the plateau method [16]: it builds two shortest-path trees, one from the source and one from the target, and then joins the two trees to obtain the branches in common. These common branches are termed plateaus. Top-k plateaus are selected based on their lengths, and each plateau is used to generate an alternative path by appending the shortest paths from the source to the first edge of the plateau and from the last edge of the plateau to the target.

All existing works validate and evaluate the goodness of their proposals from a theoretical and algorithmic point of view. None of them investigates the impact of route diversification on urban welfare, for example, traffic congestion and pollution. In this paper, we fill this gap and assess the collective impact of alternative routing as randomization of the fastest path on emissions using the mobility simulator SUMO.

3 Simulation Framework

Our simulation framework is based on SUMO (Simulation of Urban MObility), an agent-based tool that allows for intermodal traffic simulation, including road vehicles, public transport, and pedestrians [17]. SUMO models each vehicle's physics and dynamics, supporting various route choice methods and routing strategies [18].

SUMO requires two elements to simulate traffic: a road network and a traffic demand. The road network describes the virtual road infrastructure where the simulated vehicles move during the simulation. It is a directed graph $G = (V, E)$ in which V represents intersections and E represents roads. The traffic demand describes the vehicles' movement on the road network. A vehicle path may be either a trip or a route. The origin edge, the destination edge, and the departure time define a trip. A route also contains all edges the vehicle passes through.

We control our SUMO simulations through TraCI[1] (Traffic Control Interface) [19], a Python controller that allows retrieving simulated objects' values that are useful for analyzing the simulation, such as the vehicle's trajectory, its speed and acceleration, total CO$_2$ emissions, and fuel consumption.

3.1 Mobility Demand

The mobility demand $\mathcal{D} = \{T_1, \dots, T_N\}$ is a collection of N trips (one per each vehicle) within a city. A single trip $T_v = (o, d)$ is defined by its origin location o and destination location d. To compute $\mathcal{D}$, we first divide the area of interest into a grid with squared tiles of a given side. Then, we use real mobility data to compute the flows between the tiles obtaining an origin-destination matrix M where an element $m_{o,d} \in M$ describes the number of vehicles' trips that start in tile o and end in tile d. Finally, we iterate N times the following procedure: we choose a vehicle v's trip $T_v = (e_o, e_d)$ selecting at random a matrix element $m_{o,d} \in M$ with a probability $p_{o,d} \propto m_{o,d}$ and uniformly at random two edges $e_o, e_d \in E$ within tiles o and d.

3.2 Randomized Fastest Path

In graph theory, the shortest path between two nodes is the path that minimizes the sum of the weights of the path's edges. The fastest path is the shortest path considering travel time as the edge cost. We define a *randomized fastest path* as a non-deterministic distortion of the fastest path. The resulting path should not deviate considerably from the optimal path in terms of length and duration.

[1] https://sumo.dlr.de/docs/TraCI.html

We compute the randomized fastest paths using the SUMO tool duarouter[2], which allows us to compute vehicle routes using different algorithms (e.g., CHWrapper, A*, and Dijkstra) and specify the degree of path randomization $w \in [1, +\infty)$.

If $w > 1$, duarouter uses an edge weight randomization method [8] to dynamically distorts edge weights (i.e., travel time) by a chosen random factor drawn uniformly in $[1, w)$. The edge cost distortion is performed every time duarouter computes the fastest path for a vehicle; hence, two vehicles with the exact origin and destination may be assigned to two different randomized fastest paths (see Figure 1). The edge weight considered by duarouter is the expected travel time, estimated for each edge as its length divided by the maximum speed allowed on that edge. Duarouter randomizes the edge weight $f(e)$ of an edge e using a function $f_{dua}(e)$ defined as:

$$f_{dua}(e) = f(e) \cdot \mathcal{U}(1, w)$$

where w is the degree of randomization and $\mathcal{U}(1, w)$ is a random variable drawn uniformly in $[1, w)$. Note that for $w = 1$ there is no randomization. Furthermore, the higher w, the more randomness is introduced into calculating the fastest path, and the more (on average) the path deviates from the fastest path (see Figure 2). Given an origin location o, a destination location d, and random weight factor w, we define the sequence of SUMO edges computed with duarouter as $\mathcal{DR}((o, d), w)$.

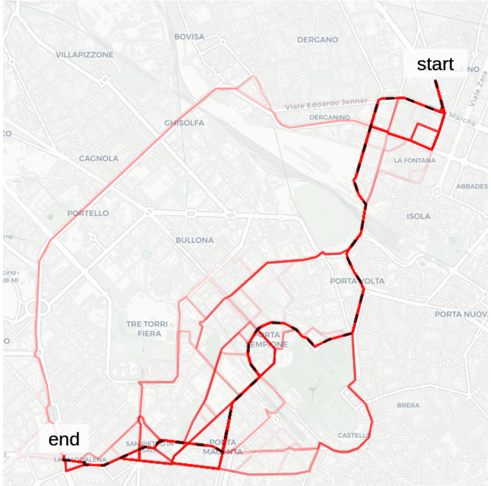


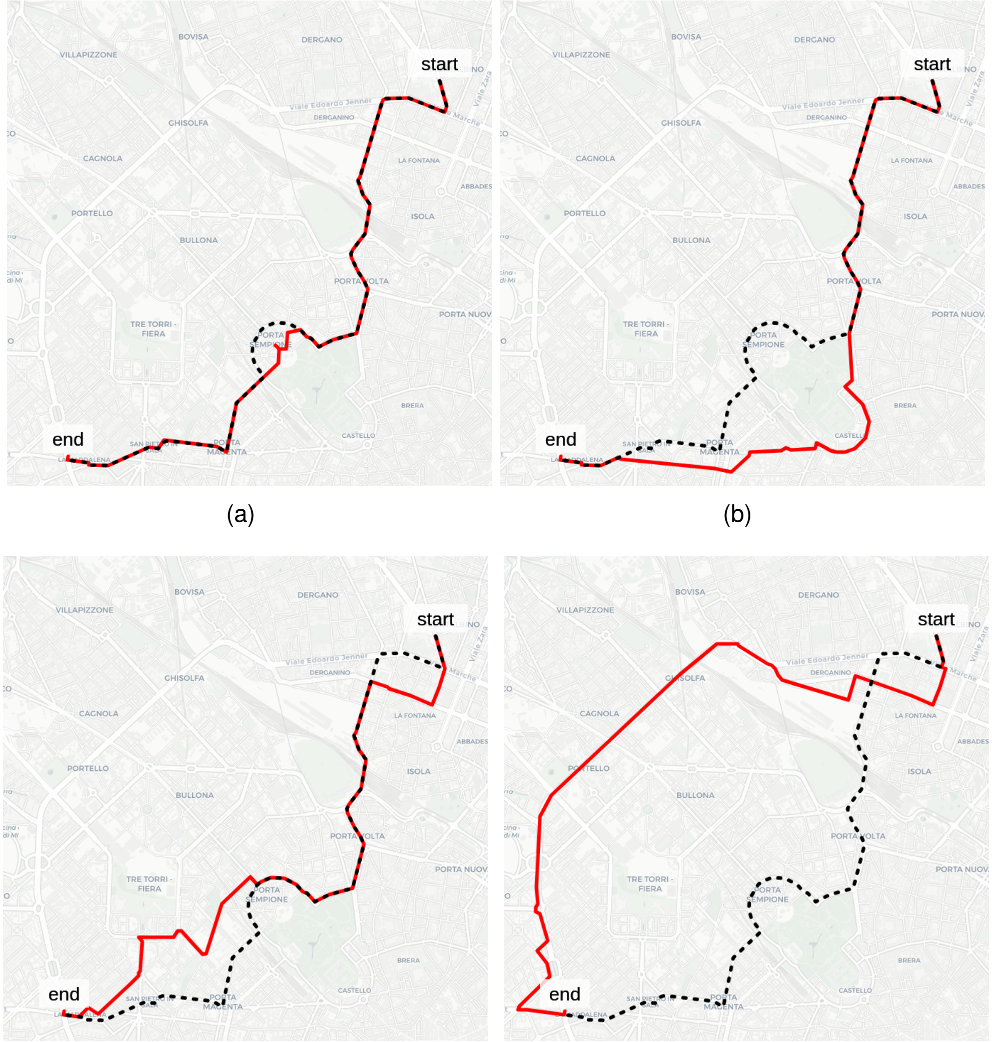
Figure 1. The non-deterministic nature of the fastest path randomization. The original fastest path is the black dashed line. We visualize the routes for 1000 randomizations of the fastest path, using $w = 5$, between the exact origin and destination edges. The intensity of the colour on a road edge is proportional to its likelihood of being selected as part of the randomized fastest path.

3.3 Non-randomized and Randomized Traffic Demands

We derive two types of traffic demands based on a given mobility demand $\mathcal{D}$: the non-randomized traffic demand and the randomized demand.

The non-randomized traffic demand, $\mathcal{NR}$, is a collection of N routes that link the origin to the destination of each trip in $\mathcal{D}$ using the fastest path (i.e., using duarouter with $w = 1$):

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



(a)

(b)

(c)

(d)

Figure 2. Randomizations (red lines) of the fastest path (black dashed line) for w=5 (a), w=10 (b), w=15 (c), and w=20 (d). Increasing w produces routes that diverge more from the fastest path.

$$\mathcal{NR}_{\mathcal{D}} = \bigcup_{T_i=(o_i,d_i)}^{\mathcal{D}} \mathcal{DR}((o_i, d_i), 1)$$

The randomized demand, $\mathcal{R}$, is the collection of N routes connecting the origins to the destinations of each trip in $\mathcal{D}$ using randomized fastest paths (i.e., using duarouter with $w > 1$):

$$\mathcal{R}_{\mathcal{D},w} = \bigcup_{T_i=(o_i,d_i)}^{\mathcal{D}} \mathcal{DR}((o_i, d_i), w)$$

Given a mobility demand $\mathcal{D}$, we compute $\mathcal{NR}_{\mathcal{D}}$ and $\mathcal{R}_{\mathcal{D},w}$ for each $w \in W$, where W is the set of randomization factors to study.

3.4 Traffic Simulation

We use TraCI to collect edge and vehicle-related measures such as total travel time, emissions (CO_2, PM, and NOx), and fuel consumption. We use the HBEFA3/PC_G_EU4 emission model [20], which estimates the vehicle's instantaneous emissions at a trajectory point j as [21]:

$$\mathcal{E}(j) = c_0 + c_1 sa + c_2 sa^2 + c_3 s + c_4 s^2 + c_5 s^3$$

where s and a are the vehicle's speed, and acceleration in point j, respectively, and $c_0, \ldots, c_5$ are parameters changing per emission type and vehicle taken from the HBEFA database.

We compute the total quantity for each pollutant on each edge $e \in E$ by summing all the emissions corresponding to any vehicle v's trajectory point that fall on e. Finally, we construct a weighted road network $\overline{G} = (V, \overline{E})$ where each edge $\overline{e} \in \overline{E}$ is associated with the amount of emissions on it.

4 Experimental Settings

We simulate the effect of route randomization into a 45 km^2 area in the city center of Milan, Italy, for which we have GPS data[3] describing 17,000 private vehicles traveling between April 2nd and 8th, 2007 (114k GPS points). Previous works demonstrate that the portion of vehicles in the dataset is representative of the real fleet of vehicles [22]. We discretize the urban area of Milan by splitting it into a grid of squared tiles (side of 1 km), and we detect the origin and destination tile of each vehicle's trip to compute the origin-destination matrix M of vehicles' flows [23], [24].

We obtain the road network $G = (V, E)$ of Milan using OSMWebWizard[4], included in the SUMO suite. Before conducting the simulations, we perform a preprocessing step on the road network to correct inaccuracies that may negatively affect the simulations. This preprocessing phase includes correcting lane number inaccuracies, addressing road continuity disruptions, and modifying turns to align with real-world conditions. Since the pre-computed traffic lights' programs often differ from those in reality, we set the traffic lights' program to *actuated*, as suggested in the SUMO documentation. The preprocessing steps are based on the methodology outlined in [18]. We use the following `netconvert` options (recommended in the `netconvert` documentation):

```
--no-turnarounds true --geometry.remove --roundabouts.guess --ramps.guess
--junctions.join --tls.guess-signals --tls.discard-simple --tls.join
--output.original-names --junctions.corner-detail 5 --output.street-names
```

After the preprocessing, the road network includes 5,551 intersections (nodes) and 36,945 road segments (edges).

Given the preprocessed road network G and the OD matrix M, we compute the mobility demand $\mathcal{D}$ with $N = 15,000$ trips. This value of N minimizes the difference between the average travel time of actual trajectories and simulated ones, a standard way to assess a realistic estimation of the number of vehicles to simulate [6], [18]. We associate at each vehicle's trip in $\mathcal{D}$ a departure time assigned uniformly at random between 0 and 3600 seconds.

[3] https://ckan-sobigdata.d4science.org/dataset/gps_track_milan_italy
[4] https://sumo.dlr.de/docs/Tutorials/OSMWebWizard.html

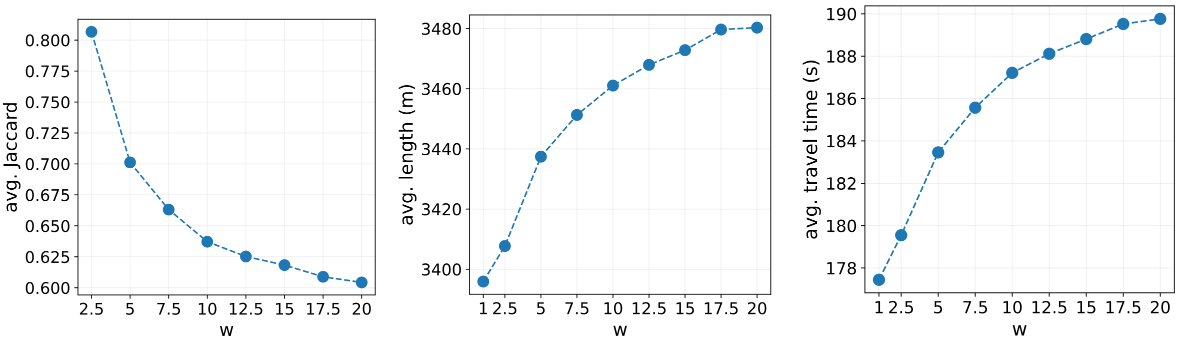
First, we compute the non-randomized traffic demand $\mathcal{NR}_{\mathcal{D}}$: we connect each vehicle's origin and destination through the fastest path ($w = 1$). Second, we build the randomized traffic demands $\mathcal{R}_{\mathcal{D},w}$ for several randomization values $w \in \{2.5, 5, 7.5, 10, 12.5, 15, 17.5, 20\}$. Third, we create a mixed demand $\mathcal{MP}_{p,w}$ specifying the fraction p of the randomized fastest paths. In each $\mathcal{MP}_{p,w}$, a fraction p of the N vehicles chosen uniformly at random are assigned to their randomized paths computed with the randomization value w. In contrast, the vehicles' remaining fraction $(1 - p)$ is assigned to their non-randomized paths. We consider $p \in [0, 1]$ at step of 0.1. The mixed demand allows us to study the impact of the percentage of vehicles that follow a randomized fastest path on the urban environment.

To make simulations more robust, for each value of p and w, we generate $\mathcal{MP}_{p,w}$ ten times, each with a different choice of randomized vehicles that are chosen uniformly at random. Finally, we simulate each $\mathcal{MP}_{p,w}$ in SUMO, and through the Python controller TraCI we collect the emissions on each edge and the vehicles' total travel time.

Finally, to confirm that the randomization of a path from an origin to a destination grows with w, we take 15,000 paths for each value of $w \in W$ (generated starting from the trips in $\mathcal{D}$). We measure the randomization of a path as the normalized Jaccard coefficient, defined between two sets A and B as:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

between the edges of the randomized paths ($w > 1$) and edges of the fastest path ($w = 1$) (Figure 3a), computed for the same origin and destination. We also measure, for different values of $w \in W$, the average path length (Figure 3b) and the average expected travel time (Figure 3c). Figure 3 shows how increasing the value of w results in randomized paths with a lower average Jaccard coefficient, higher length, and higher expected travel time than the fastest path. Therefore, path randomized grows with increasing w.



(a) (b) (c)

Figure 3. The average normalized Jaccard coefficient computed between the randomized paths and the fastest path (a), the average path length (b) and the average travel time (c) for different values of w.

5 Results

We study how the distribution of CO_2 emissions and travel time across Milan's roads change by changing with the p of randomized vehicles for different values of $w \in W$, where $W = \{2.5, 5, 7.5, 10, 12.5, 15, 17.5, 20\}$.

We find that the fraction p of randomized paths *does* impact the total CO_2: introducing randomization in the fastest path reduces the total CO_2 emissions. When $p > 0.1$, the emissions are lower than the baseline case ($w = 1$) and decrease with p, assuming their minimum value for $p = 1$ (Figure 4a). This relationship is consistent across different values of w.

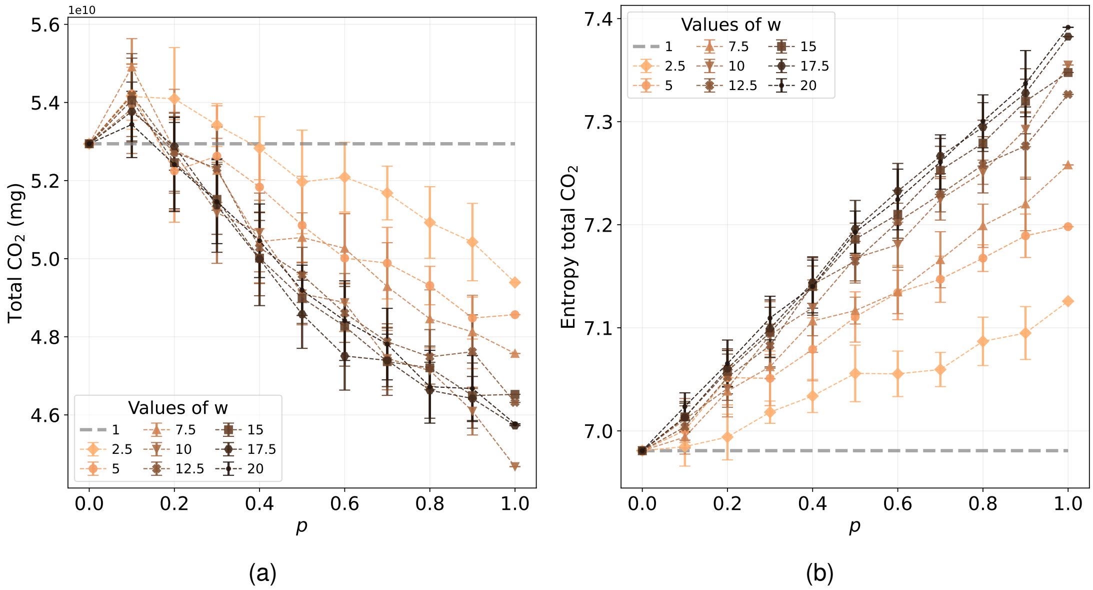


Figure 4. Total CO_2 emissions (a) and Shannon entropy (b) of the CO_2 distribution varying the fraction p of randomized vehicles for different values of w. Points indicate the average of total CO_2 emissions (a) and average Shannon entropy (b) over ten simulations with different choices of randomized vehicles (chosen uniformly at random). Vertical bars indicate the standard deviation, and the grey dashed line represents the baseline case (no randomization, w=1).

As Figure 4a shows, the configuration $p = 1$ and $w = 10$ corresponds to the minimum value of total CO_2 emissions, with a total emissions savings with respect to the baseline of 15.61$\%$ (Figure 6a). We also compute the Shannon entropy of the total CO_2 distribution to capture the inequality of the distribution on the road network's edges, defined as:

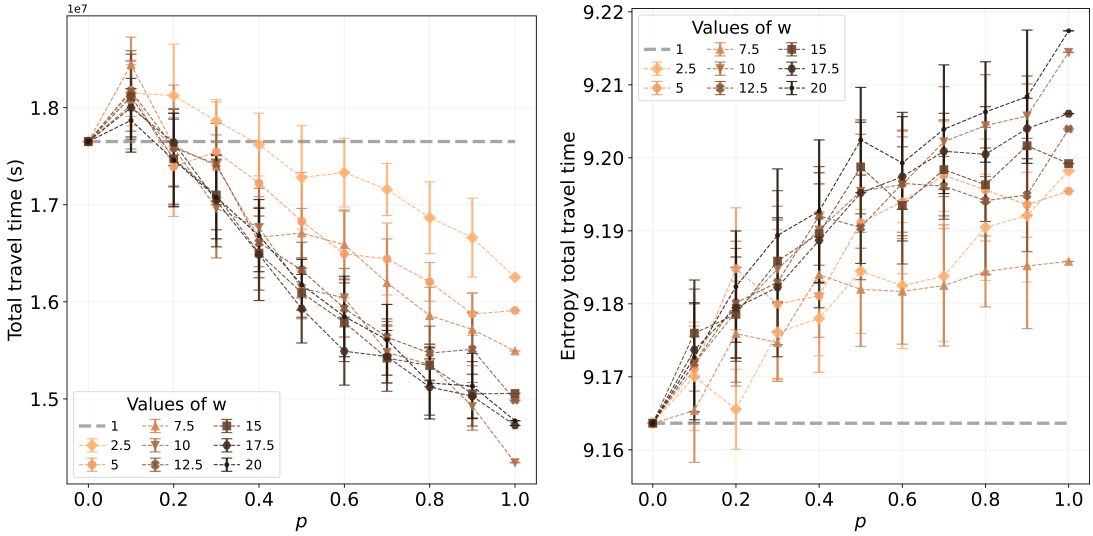
$$H(X) = -\sum_{x \in X} p(x)\ log\, p(x)$$

where X is a random variable. We find that the higher w, the more evenly the emissions are distributed on the road network (Figure 4b). In particular, the distribution is the most equal when all the vehicles follow a randomized fastest path, $\forall w \in W$.

The results for the travel time are in agreement with those of CO_2 emissions: the higher p, the lower the vehicles' travel times (Figure 5a). Travel times are minimized when $p = 1$ and $w = 10$, with an improvement of 18.74$\%$ with respect to the baseline scenario (Figure 6b). The entropy associated with the total travel time is more variable than the entropy of the total CO_2; this may arise from the stochastic nature of each simulation.

6 Discussion and Future Works

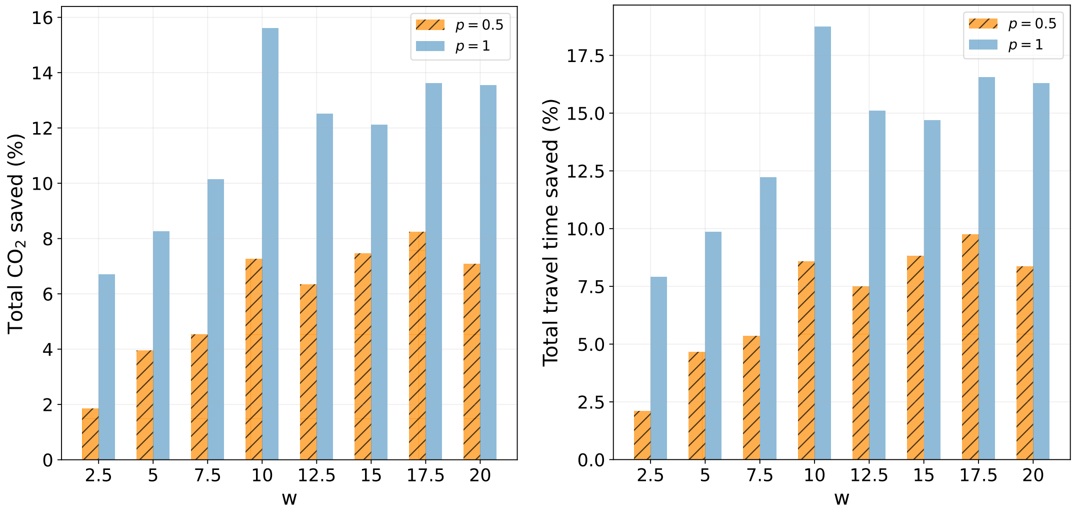
Our study investigates the effects of route randomization on CO_2 emissions and travel time in an urban environment. We find that the injection of randomness into the fastest



(a)

(b)

Figure 5. Total travel time (a) and Shannon entropy of the travel time distribution (b) varying the fraction p of randomized vehicles, different values of w. Points indicate the average total travel time (a) and average Shannon entropy (b) over ten simulations with different choices of randomized vehicles (chosen uniformly at random). Vertical bars indicate the standard deviation, and the grey dashed line represents the baseline case (no randomization, hence w=1).



(a)

(b)

Figure 6. The percentage of savings to the baseline case ($p = 0$), in terms of total CO_2 (a) and total travel time (b) for $p = 0.5$ (blue) and $p = 1$ (orange) for different values of w.

paths, which can be interpreted as an increasing “diversity” of paths on the road network, is beneficial for reducing CO_2 emissions and travel time. Presumably, path randomization helps distribute the traffic more evenly among the different (non-fastest) routes, preventing the emergence of detrimental and counterintuitive effects such as the Braess paradox [25], [26].

Path randomization with $w = 10$ leads to the best improvement over the baseline, with savings in CO_2 emission and travel time of 15.61$\%$ and 18.74$\%$, respectively. However, increasing the random component in the randomization of the fastest path is beneficial up to a certain threshold: when $w > 10$, CO_2 emissions increase again. This result suggests that more effort should be devoted on finding the dependence of the optimal degree of randomization on the road network structure and the number of circulating vehicles. In future works, we plan to explore the potential for scaling these results to other cities and integrating them into real-world transportation planning and management.

Our findings have practical implications for real-world transportation systems. Implementing our approach could significantly reduce traffic congestion and pollution, thus improving the overall efficiency of the transportation network. The approach is also easy to implement and can be integrated into existing navigation systems without significant modifications. Further research could investigate the impact of several diversification methods and other transportation efficiency measures, such as fuel consumption.

A further improvement would be to consider a stable user equilibrium (UE) [27] as a baseline scenario instead of assigning the fastest path for each trip in the travel demand. User equilibrium (UE) describes the condition in which each driver chooses their route based on their individual preferences, resulting in a network-wide equilibrium where no individual driver can reduce their travel cost (e.g., travel time) by unilaterally using a different route. In other words, UE represents a state of traffic flow where all drivers have chosen the shortest or fastest paths, given the prevailing traffic conditions and their preferences or constraints.

In the meantime, our work is a first step towards designing next-generation routing algorithms that, as our results suggest, should consider some degree of path randomization to increase urban well-being while still satisfying individual needs.

Underlying and related material

The code and the link to the dataset to fully reproduce the analysis presented in this work is available on a GitHub repository at https://bit.ly/route_randomization_sumo.

Author contributions

G.C. and L.P. did the conceptualization and wrote the paper. G.C. did the data curation, visualization, and methodology of this work. L.P., D.P., and M.N. supervised this work.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

This work has been partially supported by EU project H2020 Humane-AI-net G.A. 952026, EU project H2020 SoBigData++ G.A. 871042, ERC-2018-ADG G.A. 834756 “XAI: Science and technology for the eXplanation of AI decision making”, the CHIST-ERA grant CHIST-ERA-19-XAI-010, by MUR (grant No. not yet available), FWF (grant No. I 5205), EPSRC (grant No. EP/V055712/1), NCN (grant No. 2020/02/Y/ST6/00064), ETAg (grant No. SLTAT21096), BNSF (grant No. KП-06-AOO2/5), and by the PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - ”FAIR - Future Artificial Intelligence Research” - Spoke 1 ”Human-centered AI”, funded by the European Commission under the NextGeneration EU programme.

References

[1] M. Böhm, M. Nanni, and L. Pappalardo, “Gross polluters and vehicle emissions reduction,” *Nature Sustainability*, vol. 5, no. 8, pp. 699–707, 2022.

[2] C. K. Gately, L. R. Hutyra, and I. Sue Wing, “Cities, traffic, and co2: A multidecadal assessment of trends, drivers, and scaling relationships,” *Proceedings of the National Academy of Sciences*, vol. 112, no. 16, pp. 4999–5004, 2015, ISSN: 0027-8424. DOI: 10.1073/pnas.1421723112. eprint: https://www.pnas.org/content/112/16/4999.full.pdf. [Online]. Available: https://www.pnas.org/content/112/16/4999.

[3] R. Zhu, M. Wong, E. Guilbert, and P. Chan, “Understanding heat patterns produced by vehicular flows in urban areas,” *Scientific Reports*, vol. 7, Nov. 2017. DOI: 10.1038/s41598-017-15869-6.

[4] B. Zhou, D. Rybski, and J. Kropp, “The role of city size and urban form in the surface urban heat island,” *Scientific Reports*, vol. 7, p. 4791, Jul. 2017. DOI: 10.1038/s41598-017-04242-2.

[5] M. Barth and K. Boriboonsomsin, “Real-world carbon dioxide impacts of traffic congestion,” *Transportation research record*, vol. 2058, no. 1, pp. 163–171, 2008.

[6] G. Cornacchia, M. Böhm, G. Mauro, M. Nanni, D. Pedreschi, and L. Pappalardo, “How routing strategies impact urban emissions,” in *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, ser. SIGSPATIAL ’22, Seattle, Washington: Association for Computing Machinery, 2022, ISBN: 9781450395298. DOI: 10.1145/3557915.3560977. [Online]. Available: https://doi.org/10.1145/3557915.3560977.

[7] L. Li, M. Cheema, H. Lu, M. Ali, and A. N. Toosi, “Comparing alternative route planning techniques: A comparative user study on melbourne, dhaka and copenhagen road networks,” *IEEE Transactions on Knowledge & Data Engineering*, vol. 34, no. 11, pp. 5552–5557, Nov. 2022, ISSN: 1558-2191. DOI: 10.1109/TKDE.2021.3063717.

[8] D. Cheng, O. Gkountouna, A. Züfle, D. Pfoser, and C. Wenk, “Shortest-path diversification through network penalization: A washington dc area case study,” in *Proceedings of the 12th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, ser. IWCTS’19, Chicago, IL, USA: Association for Computing Machinery, 2019, ISBN: 9781450369671. DOI: 10.1145/3357000.3366137. [Online]. Available: https://doi.org/10.1145/3357000.3366137.

[9] H. Liu, C. Jin, B. Yang, and A. Zhou, “Finding top-k shortest paths with diversity,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 3, pp. 488–502, 2018. DOI: 10.1109/TKDE.2017.2773492.

[10] T. Chondrogiannis, P. Bouros, J. Gamper, and U. Leser, "Alternative routing: K-shortest paths with limited overlap," ser. SIGSPATIAL '15, Seattle, Washington: Association for Computing Machinery, 2015, ISBN: 9781450339674. DOI: 10.1145/2820783.2820858. [Online]. Available: https://doi.org/10.1145/2820783.2820858.

[11] T. Chondrogiannis, P. Bouros, J. Gamper, U. Leser, and D. B. Blumenthal, "Finding k-dissimilar paths with minimum collective length," ser. SIGSPATIAL '18, Seattle, Washington: Association for Computing Machinery, 2018, pp. 404–407, ISBN: 9781450358897. DOI: 10.1145/3274895.3274903. [Online]. Available: https://doi.org/10.1145/3274895.3274903.

[12] L. Wu, X. Xiao, D. Deng, G. Cong, A. D. Zhu, and S. Zhou, "Shortest path and distance queries on road networks: An experimental evaluation," 2012. DOI: 10.48550/ARXIV.1201.6564. [Online]. Available: https://arxiv.org/abs/1201.6564.

[13] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971. DOI: 10.1287/mnsc.17.11.712. eprint: https://doi.org/10.1287/mnsc.17.11.712. [Online]. Available: https://doi.org/10.1287/mnsc.17.11.712.

[14] H. Aljazzar and S. Leue, "K ⋆: A heuristic search algorithm for finding the k shortest paths," *Artificial Intelligence - AI*, vol. 175, pp. 2129–2154, Dec. 2011. DOI: 10.1016/j.artint.2011.07.003.

[15] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974. DOI: https://doi.org/10.1002/net.3230040204. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230040204. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230040204.

[16] C. V. I. T. Ltd., *Choice routing*, http://www.camvit.com, 2005.

[17] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582. DOI: 10.1109/ITSC.2018.8569938.

[18] J. Argota Sánchez-Vaquerizo, "Getting real: The challenge of building and validating a large-scale digital twin of barcelona's traffic with empirical data," *ISPRS International Journal of Geo-Information*, vol. 11, no. 1, 2022, ISSN: 2220-9964. DOI: 10.3390/ijgi11010024. [Online]. Available: https://www.mdpi.com/2220-9964/11/1/24.

[19] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "Traci: An interface for coupling road traffic and network simulators," in *Proceedings of the 11th Communications and Networking Simulation Symposium*, ser. CNS '08, Ottawa, Canada: Association for Computing Machinery, 2008, pp. 155–163, ISBN: 1565553187. DOI: 10.1145/1400713.1400740. [Online]. Available: https://doi.org/10.1145/1400713.1400740.

[20] INFRAS, *Handbuch für emissionsfaktoren*, http://www.hbefa.net/, 2013.

[21] D. Krajzewicz, M. Behrisch, P. Wagner, R. Luz, and M. Krumnow, "Second generation of pollutant emission models for sumo," in *Modeling Mobility with Open Data*, M. Behrisch and M. Weber, Eds., Cham: Springer International Publishing, 2015, pp. 203–221.

[22] L. Pappalardo, S. Rinzivillo, Z. Qu, D. Pedreschi, and F. Giannotti, "Understanding the patterns of car travel," *The European Physical Journal Special Topics*, vol. 215, pp. 61–73, 2013.

[23] R. Hariharan and K. Toyama, "Project lachesis: Parsing and modeling location histories," vol. 3234, Oct. 2004, pp. 106–124, ISBN: 978-3-540-23558-3. DOI: 10.1007/978-3-540-30231-5_8.

[24] L. Pappalardo, F. Simini, G. Barlacchi, and R. Pellungrini, *Scikit-mobility: A python library for the analysis, generation and risk assessment of mobility data*, 2019. DOI: 10.48550/ARXIV.1907.07062. [Online]. Available: https://arxiv.org/abs/1907.07062.

[25] D. Braess, A. Nagurney, and T. Wakolbinger, "On a paradox of traffic planning," *Transportation Science*, vol. 39, no. 4, pp. 446–450, Nov. 2005, ISSN: 1526-5447. DOI: 10.1287/trsc.1050.0127. [Online]. Available: https://doi.org/10.1287/trsc.1050.0127.

[26] D. Braess, "Über ein paradoxon aus der verkehrsplanung," *Unternehmensforschung*, vol. 12, pp. 258–268, 1968.

[27] J. Wardrop, "Some theoretical aspects of road traffic research," *Proceedings of the Institution of Civil Engineers*, vol. 1, pp. 325–378, 1952.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.218

Published: 29 June 2023

A Technical Concept for sensor-based Traffic Flow Optimization on connected real-world intersections via a SUMO Feature Gap Analysis

Ingo Trautwein[1] [https://orcid.org/0009-0006-5092-3999], Andreas Freymann[1] [https://orcid.org/0000-0002-3735-4545], Emanuel Reichsoellner[2], Jessica Schoeps Kraus[1], Sonntag Mirco[2], and Thomas Schrodi[1]

[1] Fraunhofer IAO, Stuttgart, Germany
[2] University of Applied Sciences Esslingen, Germany

Abstract: Traffic within cities has increased in the last decades due to increasing mobility, changing mobility behavior and new mobility offerings. These accelerating changes make it increasingly difficult for responsible authorities or other stakeholders to predict mobility behavior, to configure traffic rules or to size roads, bridges and parking lots. Traffic simulations are a powerful tool for estimating and evaluating current and future mobility, upcoming traffic services and automated functionalities in the domain of *traffic management*. For being able to simulate a complex real-world traffic environment and traffic incidents, the simulation environment needs to fulfill requirements from real-world scenarios related to sensor-based data processing. In addition, it must be possible to include latest advancements of technology in the simulation environment, for instance, (1) connected intersections that communicate with each other, (2) a complex and flexible set of rules for *traffic sign control* and *traffic management* or a well-defined data processing of relevant sensor data. In this paper we therefore define requirements for a traffic simulation based on a complex real-world scenario in Germany. The project addresses major urban challenges and aims at demonstrating the contribution that the upcoming 5G mobile generation can make to solving real-time traffic flow optimization. In a second step, we investigate in detail if the simulation environment SUMO (Simulation of Urban Mobility) fulfills the postulated requirements. Thirdly, we propose a *technical concept* to close the gap of the uncovered requirements for later implementation.

Keywords: Traffic flow optimization, connected intersections, traffic light systems

1 Introduction

In the last decade, traffic density has increased worldwide by the growing amount of vehicles making the management of the traffic more challenging [1], [2]. This leads to more traffic jams, accidents [3] or transportation problems such as costly delays [4]. Reasons for this are, e.g., the growing car ownership and the behaviour of road users [1], [5]. Thus, solutions for traffic flow optimization is important [4]. Especially with respect to multi-lane intersections [5], different traffic flows crossing each other and these crossings have to be regulated. The regulation can take place via traffic light systems and they thus have an effect on the traffic flow. Developing and testing such optimizations in the real world is time-consuming, expensive, hardly reproducible and a potential safety risk. Thus, traffic experiments should be performed in a virtual environment using traffic simulations [4]. Mapping the real world into a virtual environment and simulation which are also being studied intensively in the research community [2], [4] is of high priority for the traffic management [1]. Simulations require models which contains roads, buildings, traffic lights or traffic demands to reflect the reality in those aspects that are needed to find responses for posed research questions [6]. SUMO offers functions and tools for applying traffic simulations such as traffic lights or road data [6]. Traffic demand represents a basic information for running a SUMO simulation that is provided in different modes by SUMO [6]. Additionally, SUMO offers functionalities to understand the traffic state [6]. However, it has to be validated which requirements of the real-world scenario can be fulfilled with SUMO built-in features and which components have to be developed further. This paper addresses this issue by identifying project specific requirements, derived from a concrete real-world project called *5G-trAAffic* [1] supported by the German Federal Ministry of Transport and Digital Infrastructure, by comparing it with the current SUMO functionalities. One goal of the *5G-trAAffic* project is to optimize the *traffic flow* by tweaking the way traffic lights are managed at various multi-lane intersections. This is going to be realized by connected the intersections by a cooperative behaviour of the *traffic signal systems*.

The main goal of this work is to identify gaps between the project related requirements and the current SUMO functionalities ending up with a technical concept for a later implementation of new extensions or software plugins for SUMO. In the core, as Fig. 1 shows, we, firstly, analyze which requirements are demanded for SUMO to be able to closely represent the project reality within the virtual environment. Secondly, the concrete requirements are sorted within categories such as the geographical environment, the traffic logic such as a traffic management, and the data creation of relevant data sources such as traffic sensors as a traffic optimization is related to a sensor-based data processing. In addition, the identified gaps between the demanded requirements and the SUMO functionality resulted within a *gap analysis*, which transparently shows required SUMO enhancements to close the gaps. Thirdly, in the last step we use the gap analysis to suppose a *technical concept* which provides an overview of the solution approach.

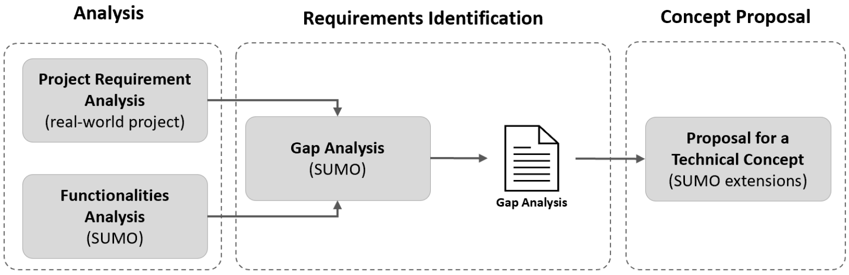

Analysis
Requirements Identification
Concept Proposal
Project Requirement Analysis (real-world project)
Functionalities Analysis (SUMO)
Gap Analysis (SUMO)
Gap Analysis
Proposal for a Technical Concept (SUMO extensions)

Figure 1. Overview over the work content

The structure of the paper is organized as follows: Section 2 provides a review of relevant literature. Section 3 presents an overview of the considered road intersections in the project and their specific conditions. It also includes the identified requirements on the simulation environment. In Section 4 there will be shown which requirements can already be fulfilled by SUMO and which requirements needs to be integrated to

[1] https://www.keim.iao.fraunhofer.de/de/projekte/5G.html

meet all requirements. Subsequently, Section 5 presents the technical concept with a diagram and description about the approach to extend SUMO. Finally, Section 6 provides an overall evaluation and reflection of the results.

2 Related Work

Our work strongly deals with several fields with respect to the traffic flow optimization. Jin et al. [7] generated an intelligent control system for traffic signal applications called Fuzzy Intelligent Traffic Signal (FITS) control. The system is designed to improve existing signal control systems without changing the fundamental traffic management infrastructure. It uses fuzzy logic for decision making and communicates with signal control hardware and simulated traffic environment through a M2M connectivity protocol.

Traffic monitoring is required to minimize traffic congestion and accidents and to develop *intelligent transportation systems* (ITS). Mega et al. [8] analyze a realistic simulation of vehicle mobility based on traffic surveys of a city using SUMO. The traffic information was obtained by integrating *OpenStreetMap* with request data and filtering with a Kalman Filter method. Muzamil et al. [9] designed an adaptive traffic signal control based on fuzzy logic with Webster and modified Webster's formula considering the average waiting time, average travel time, and average speed as performance indices. Data from three, four and five intersections were collected and by using SUMO it turned out that the two proposed adaptive traffic light control algorithms showed better results in terms of performance indices at a four-access intersection. In the *KI4LSA* project phase [10], a solution for real-time data-based traffic flow optimization of existing traffic light systems with artificial intelligence (AI) was proposed. They used SUMO and collected data from different sensors such as radar sensors or video-based sensors of the city of Lemgo. They showed a positive impact on the traffic flow by applying linear optimization. In comparison, our focus is on collecting other kinds of data like statistical and historical data sensor data, which later in the project will result in a set of rules that will provide the *traffic light systems* with suggestions for the next *traffic light* phase. Tomar et al. [11] address the problem to update traffic signal timing and synchronize the traffic signal at intersections based on real-time traffic information. They provide an overview of traffic light synchronization techniques for intelligent vehicles, a cutting-edge technology in ITS. They proposed a traffic signal synchronization framework which divides the system in two levels of synchronization to reduce the congestion, working with any technology to collect traffic density at intersection. Pandey and Jalan [12] use image and video processing to gather information on traffic density at an intersection and develop a strategy for allocating time periods for traffic signals. With their suggestion they want to reduce the waiting times for vehicles and reduce the average waiting queues.

The work of Lobo et al. shows the concept, model and validation to presents a realistic traffic scenario for *Ingolstadt*. For this purpose, observations from the real world were taken over, such as the average green phases at traffic intersections [13]. In contrast to our approach, we want to access the actual sensor data to provide an accurate representation of the traffic light phases and intersection situation in the simulation environment. In Luow at al. [14] the queue length at intersections is to be optimized. For this purpose, the two approaches of Machine Learning: Q-learning and policy optimization are faced. Here, the agents have access to the state of the traffic light as well as the number of vehicles. This approach excludes environmental factors to be considered in this work. Halbach and Erdmann [15] focus on the setup of intersections with SUMO and the implementation of traffic light codes of a manufacturer, which are transmitted in a fixed interval. The transmission is realized by *Traffic Control Interface* (TraCI).

3 Requirements Analysis

Deriving requirements from the real-world project *5G-trAAffic* for virtualization, the real-world project needs to be analyzed in detail including the core characteristics such as the traffic flow or the logic of traffic management. Thus, in the following the project is described in detail followed by the the requirements analysis.

3.1 Overview of the 5G-trAAffic Project

The main goal of the *5G-trAAffic* project is to examine potential applications of the 5G mobile communications technology in cities. The project addresses major urban challenges and aims at demonstrating how the upcoming 5G mobile communications generation can contribute as a key technology to the solution of real-time *traffic flow optimization*.

The project is located in Aalen, Germany. Fig. 2 shows the three traffic intersections in Aalen that are taken as concrete scenario for the traffic flow optimization.

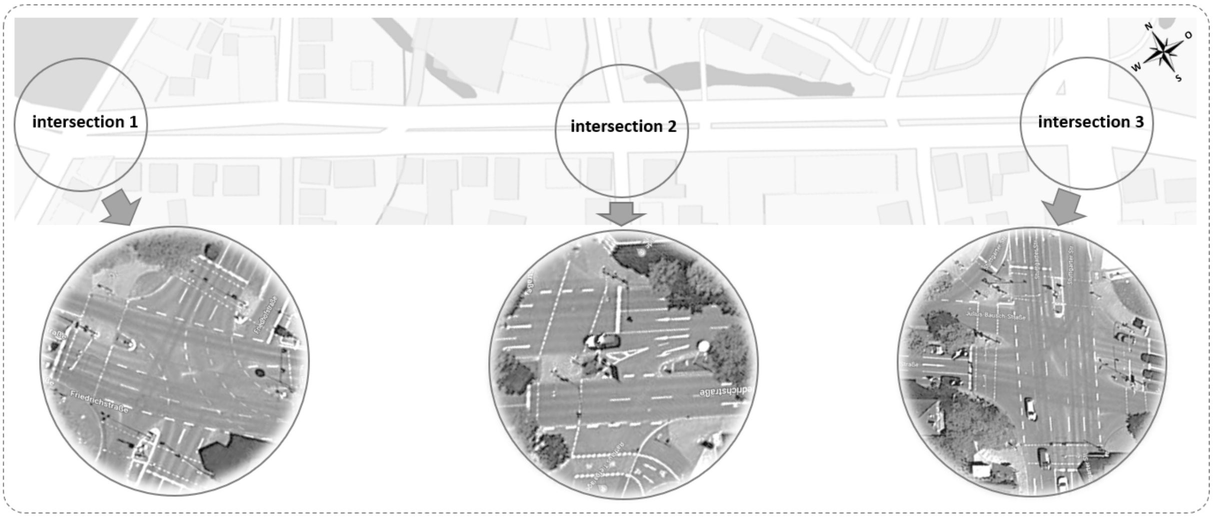


Figure 2. Overview about the intersections within the 5G-trAAffic project

The two streets *Friedrichsstraße* and *Friedhofstraße* represent *Intersection 1*. The middle intersection *Intersection 2* contains the both crossing streets *Gartenstraße* and *Friedrichstraße* and connects the *Intersection 1* and *Intersection 3*. The last intersection *Intersection 3* contains the *Stuttgarter Str.* and the *Julius-Bausch-Straße*. The intersections are connected by multi-lane roads and build major traffic axes. The traffic is characterized by motorized traffic including bicycle lanes, cars, trucks, buses or motorcycles. Pedestrians can request green phases via a request button on the traffic light pole. Induction loops and basic video detection to identify the amount of road users are provided at all intersections. The existing traffic light systems have a static traffic signal phasing program and have no knowledge about the current or next phase of the other intersections. The selection of the next traffic light phase can be influenced by the request buttons (pedestrians), induction loops and video detection. Several additional modern sensor devices will be added to the intersections and roads comprising, e.g., road-side units or weather sensors. These sensors can enable better optimization of traffic flow by providing additional relevant data for optimization. Furthermore, the driving areas will be equipped with modern AI assisted camera systems that can establish the fluctuation of the travel. This includes to detect the amount of vehicles per time as well as the class of vehicles (e.g., bus or car). Camera systems will enable tracking in which direction a vehicle is travelling. Thus, the camera systems will exchange data with each other. The current basic operating principle of the traffic light systems is to receive internal suggestions from the traffic system to decide which traffic light phase should be selected next. This should enable the traffic light system to react to local traffic conditions, but also to achieve macroscopic overall effects. Fig. 3 shows the *5G-trAAffic* project and its basic solution approach in order to optimize the traffic flow.

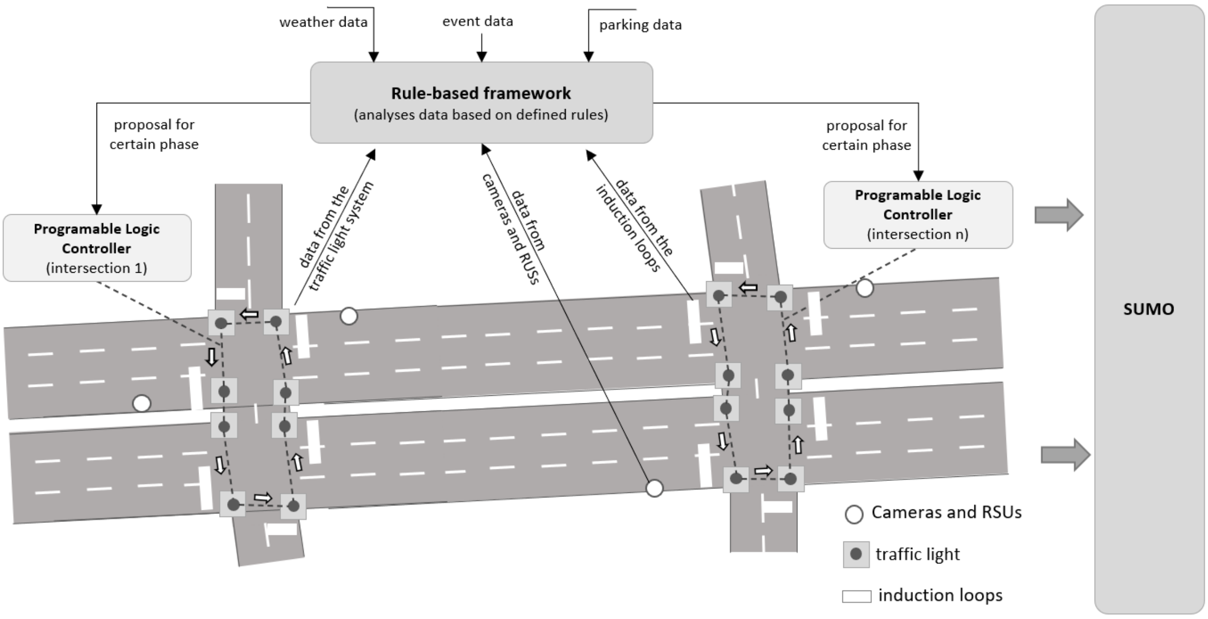

weather data
event data
parking data
Rule-based framework
(analyses data based on defined rules)
proposal for certain phase
proposal for certain phase
data from the traffic light system
data from cameras and RUSs
data from the induction loops
Programable Logic Controller (intersection 1)
Programable Logic Controller (intersection n)
SUMO
○ Cameras and RSUs
● traffic light
▭ induction loops

Figure 3. Overview over the traffic flow optimization using the rules based framework

The core of the traffic flow optimization are based on a *rules-based framework* which optimizes the traffic lights by sending a proposal for the next phases based on multiple data sources. Algorithms are being evaluated that calculate a recommendation for the next phase, based on the input signals from the sensors. The characteristics of the traffic light of the considered intersections *5G-trAAffic* is that they have a built-in logic to accept or reject the proposal. In real world this is performed by a hardware unit called a programmable logic controller (PLC). For simulation purpose in the running project *5G-trAAffic*, the PLC has to be emulated by a software module. This means that the behavior of the hardware unit (PLC) from the real world has to be modeled with a self developed software module. The PLC serves as an interface between the rule-based framework and the traffic signal system. The incoming proposal for the next traffic light phase is received and evaluated by the PLC. By accepting the suggestion, it sends the suggested phase to the traffic signal. Which requirements the SUMO environment needs to fulfill is part of this paper. In the next Section the requirements are identified that have to be fulfilled by the simulation environment.

3.2 Requirements Analysis

The requirements for the *5G-trAAffic* project focus all on traffic flow optimization and not on traffic safety issues. Pedestrians will just be considered to represent how pedestrian requests affect the vehicle traffic flow. The requirements are structured in several groups. The first group addresses the requirements for the *traffic signal systems*. These are central to the objective of *traffic flow optimization*. The second requirement group is made up of the sensor systems which are used with the 5G technology, called the *5G sensor systems*. The described *5G-trAAffic* project uses already installed sensor systems such as induction loops or specific sensors (e.g., camera systems or 5G components) which provide the input sensor-data for the SUMO simulation. The third requirement group focuses the traffic simulation itself, which combine the *traffic signal systems* and the *5G sensor systems*. The fourth requirement group represents the *traffic flow optimization*. Table 1 shows the identified requirements for the *traffic light system*.

Table 1. Traffic Light System Requirements

Requirements - Traffic Light System	
RID	Description
1.1	Traffic signals need to be manageable via phase transitions.
1.2	Selecting the next phase, incoming signals from sensor devices need be considered.
1.3	An interface is needed to request the next phase based on the current input signals.
1.4	An interface is needed to change the current phase to the transmitted phase.
1.5	A PLC is needed to decide if a suggestion of the next phase is accepted or rejected.

Controlling traffic signals via phases within a traffic signal program is a standardized approach. Thus, the requirement *RID 1.1* is of particular importance. Furthermore, a traffic light program consists of multiple phases and its transitions. A 5G sensor system should deliver relevant information to determine the next phase (*RID 1.2*). For the development optimization of the traffic flow, the basic information about current state of the traffic light must be known by transmitting relevant data via a proper interface, defined in requirement *RID 1.3*. The traffic signal system requires an interface to change to another phase. This interface is necessary because the generated phase proposal of the rule-based framework need to to be transmit to the traffic light system.*RID 1.4*. PLC are installed in the traffic signal systems relevant to the project. The phase proposals are sent to the PLC. One task of the PLC is to check whether external signals such as the proposal for the next phase are accepted or not. The PLC decides on the basis of defined criteria whether a phase proposal is accepted or not. *RID 1.5*

In Table 2, the identified requirements for the *5G sensor system* are presented. In the project also pedestrians are considered and they can use request buttons at the intersections. Thus, the simulation needs to consider such functionality for the traffic flow optimization *RID 2.1*. In addition, the project requires detectors to distinguish between different types of vehicles such as cars, trucks of busses, which is set in *RID 2.2*. The used cameras must be connected with each other by a camera system, because they share information about the traffic comprising, etc, detected vehicles and their driving directions, described in *RID 2.3*. Finally, further information from vehicles such as speed or the destination lane is also required for the later optimization. Thus, road side units are installed which needs to be considered *RID 2.4*.

Table 3 shows the identified requirements for the *traffic simulation*. Creating a realistic model from real world data, it must be possible to reproduce the road layout as accurately as possible (i.e., *RID 3.1*). In addition, the traffic simulation needs to consider different road users for cars, buses and pedestrians *i.e., RID 3.2*. For the data collec-

Table 2. 5G Sensor System Requirements

Requirements - 5G Sensor System	
RID	Description
2.1	Request buttons need to be considered used by pedestrians influencing the next phase.
2.2	Induction loops need to be considered counting the traffic and making a distinction between cars, busses or trucks.
2.3	The traffic cameras need to be connected to an intelligent camera system to track each vehicle or pedestrian from entering until leaving the intersection.
2.4	Road side units needs to be included that can communicate with vehicles.

tion, 5G sensor systems need to be adaptable to the changing settings, meaning that properties of the sensor systems must be customizable (i.e., *RID 3.3*). During the simulation the behaviour of vehicles, the pedestrians and cyclists needs to be considered based on statistical information (i.e., *RID 3.4*).

Table 3. Traffic Simulation Requirements

Requirements - Traffic Simulation	
RID	Description
3.1	The intersections need to be modelled closely to the real-world project.
3.2	Different types of lanes need to be distinguished including, pedestrians, bicycles, cars/trucks and busses.
3.3	The 5G sensor systems need to be configurable within the traffic simulation in order to match the real-world project characteristics such as angle of the cameras.
3.4	Behavior pedestrians, cyclists or vehicles needs to be considered within the simulation.

Table 4 shows the identified requirements for the *traffic flow optimization*. The algorithm for determining the next traffic light phase requires the current status of all sensors. This is specified in requirement *RID 4.1*. Requirement *RID 4.2* states that an interface must exist at the traffic signal system for the transmission of a suggestion for the next traffic light phase. For the evaluation of the developed traffic flow optimization, a feedback from the traffic signal system is necessary to see if a proposal for the next phase has been accepted or rejected (i.e., *RID 4.3*). The identified requirements will be used in Section 4 to check which requirements can be fulfilled by SUMO and to identify the gaps.

Table 4. Traffic Flow Optimization Requirements

Requirements - Traffic Flow Optimization	
RID	Description
4.1	The simulation need to request the current state of each sensor via an interface.
4.2	Suggestions or next phases of traffic light systems need to be interface-requestable.
4.3	A feedback mechanism is needed to get information of the suggested phase acceptance.

4 The SUMO Traffic Simulation Environment

In this Section, the requirements from Section 3.2 are compared to the available features in SUMO by performing a gap analysis based on the official documentation of SUMO [2] as well on Lopez et al. [6]. The results of the gap analysis are presented and discussed in Section 4.2. By mentioning SUMO, the whole SUMO package, including all official components, which is available at the official SUMO webpage[3] is meant. By mentioning TraCI, which stands for Traffic Control Interface, the Python[4] based TraCI library[5] is meant. TraCI is an interface to interact with a running simulation, to retrieve current values and to modify simulation parameters and properties of simulated objects. This allows to implement highly customizable traffic simulation models.

4.1 Existing built-in Features

4.1.1 Traffic Light Systems

SUMO offers sophisticated standard functionalities for traffic light system controls, such as fixed-timing control and adaptive traffic control based on detected time gaps [16]. The integration of *traffic light systems* (TLS) into SUMO models can be realized by the following SUMO tools: *netconvert* [17] or the OSMWebWizard[18] to work with real world maps and *netgenerate* [19] to work with generic traffic networks (e.g., grid or spider web shaped). These tools automatically create *traffic light systems* and *traffic light programs* with default settings, which are both represented in XML files. The cycle length is by default 90 seconds and the green time is equally split between the main phases. The created XML files can be customized, by modifying, adding or removing parameters, *traffic light programs* or whole *traffic light systems* to the corresponding XML files. By default, the latest assigned program will be used. Switching between predefined programs can be performed by TraCI or by WAUT switches [16], which can switch between different traffic light programs for different timestamps [16]. To simplify the import of TLS programs from real world TLS, SUMO provides Python scripts such as *tls_csv2SUMO.py* and *tls_csvSignalGroup.py* [16]. Furthermore, it is possible to model induction loop detectors to detect time gaps between vehicles, which affects the phase duration of TLS. Customizing different behaviour for traffic light systems can be implemented with TraCI, which provides various TLS functions [20] like *setPhase*, *setPhaseDuration*, *setProgram* or *setProgramLogic*.

4.1.2 5G Sensor Systems

5G Sensor Systems can be modeled by the offered types of detectors provided by SUMO. There are different types of detectors to retrieve data from intersections, which can be used to optimize traffic light programs. They comprise the induction loop detector (named as E1 [16]), lane-area detectors (named as E2 [16]) as well as the multi-entry-exit detectors (named as E3 [16]). The induction loop detectors are simple detectors to retrieve the number of passed vehicles, the time gap between vehicles, the length and the speed of the passed vehicles. Lane-area detectors represent vehicle tracking cameras for certain areas. Compared to simple induction loop detectors, lane-area detectors track each vehicle that enters the supervised area and measures queues of jammed vehicles, jam length and halting duration. This information can be

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

[3] https://www.eclipse.org/sumo/

[4] https://www.python.org/

[5] https://sumo.dlr.de/docs/TraCI/Interfacing_TraCI_from_Python.html

coupled with *traffic light systems* to optimize their programs. With multi-entry-exit detectors (named as E3 [16]) the entering vehicles and exit positions can be supervised and the information can be processed via TraCI [21].

4.1.3 Traffic Simulation

SUMO provides various tools like the OSMWebWizard[18] and the netconvert[17] to create simulation models from real-world map data including road networks that consist of different lane types for cars, trucks, buses, bicycles and pedestrians. The infrastructure, the vehicles, the pedestrians, the routes, the traffic light programs and other relevant information and parameters like simulation time, speed limits and overtake behaviour are defined with a set of XML files. The traffic simulation can be performed in two possible modes: a static mode and a dynamic mode. The static mode uses static settings based on XML files including, e.g., fixed defined routes or speeds. The dynamic mode enables interactive behaviour during the runtime based on TraCI. Additionally, TraCI provides data for each entity of the simulation and the associated parameters can be modified. This allows, e.g., changing routes, speed limits and vehicle parameters during the runtime.

4.1.4 Traffic Flow Optimization

Using TraCI allows the implementation of highly customizable traffic flow optimization modules, which allows to retrieve different kinds of detector data. Furthermore, that modules can also switch between traffic light phases and evaluate traffic flow data (e.g., number of vehicles per lane or number of entering and exiting vehicles per lane). The core task of the a traffic flow optimization is to find an optimized next phase for the traffic light system, for example to reduce waiting times. This information about the next traffic light phase need to be sent as a proposal to the traffic light system by the *rule-based framework*. Also further data sources, like weather data or event data can be taken into account to adjust the simulation parameters according to the real traffic behaviour.

4.2 Gap Analysis

In the following, a gap analysis is performed and presented in Table 5. It shows the requirements, described in Section 3.2, and its degree of fulfillment by SUMO to model the real world scenario described in Section 3.1. The fulfillment differs between completely fulfilled, partly fulfilled, and not fulfilled. The requirements are structured in Table 5 and in the description by the aforementioned topics, comprising traffic light system, 5G sensor system, traffic simulation and traffic flow optimization.

4.2.1 Traffic Light System

RIDs 1.1 to 1.4: The requirements for modeling traffic light systems are completely fulfilled. SUMO provides sophisticated functions to define the phase transitions via different TLS Programs, to retrieve signals from traffic sensors and to change between predefined TLS programs during runtime.
RID 1.5: The requirement is not fulfilled, because the programmable logic controller to accept or reject proposals, which are sent to the traffic light systems for changing their current traffic light program is not part of the SUMO standard functionality and must be

Table 5. Overview of the gap analysis.

Gap Analysis - Traffic Light System Requirements		
RID	Fulfilled	Description
1.1	Yes	SUMO provides customizable TLS-Programs
1.2	Yes	The TLS can be actuated by detectors
1.3	Yes	TLS phases can be switched with TraCI
1.4	Yes	For one TLS several interchangeable Programs can be defined
1.5	No	PLCs are not represented in SUMO
Gap Analysis - 5G Sensor Systems Requirements		
RID	Fulfilled	Description
2.1	No	There is no TLS request button, but this can be emulated
2.2	Yes	E1 Induction Loop Detectors count and measures time gaps
2.3	Partly	Camera interaction can be modeled with TraCI
2.4	No	But 5G Sensors can be represented with TraCI
Gap Analysis - Traffic Simulation Requirements		
RID	Fulfilled	Description
3.1	Yes	SUMO provides tools to import OpenStreetMap data
3.2	Yes	SUMO allows to model different lane types
3.3	Yes	The Sensor Types E1, E2 and E3 are customizable
3.4	Yes	Each vehicle / person can be accessed individually
Gap Analysis - Traffic Flow Optimization Requirements		
RID	Fulfilled	Description
4.1	Yes	TraCI allows sensor value retrieval
4.2	Yes	TLS phases can be changed by TraCI
4.3	Partly	A feedback mechanism can be implemented

extended by a custom implementation. For the integration into the SUMO environment, TraCI can be utilized.

4.2.2 5G Sensor Systems

RID 2.1: This requirement is not fulfilled, because SUMO does not provide traffic light request buttons for pedestrians, but such buttons can be emulated via a custom software implementation by interacting with the simulation by TraCI according to an already existing tutorial[22].
RID 2.2: This requirement is completely fulfilled, because induction loops can be modeled with induction loop detectors (names as E1[16]) and customized with a set of parameters.
RID 2.3: This requirement is partly fulfilled, because traffic camera behaviour can be modeled with multi-entry-exit detectors (named as E3 [16]) to detect source sink relations for each lane, but not for tracking individual vehicles over a wider range (e.g., several intersections). Therefore the behaviour of the AI assisted cameras, described in section 3.1 needs to be implemented, by connecting several multi-entry-exit detectors (named as E3 [16]) via TraCI.
RID 2.4: This requirement for 5G Sensor Systems is not fulfilled. The reason for this can be found in the very specialized use case described in Section 3.1 and in the fact that 5G sensors are not in the scope of the SUMO environment. But their behaviour can be emulated and integrated via TraCI.

4.2.3 Traffic Simulation

RIDs 3.1 to 3.4: The requirements for the traffic simulation are completely fulfilled. SUMO provides all required functions and tools to set up simulation models and to perform sophisticated traffic simulations. The OpenStreetMap[6] Data is imported, processed and a set of XML files are created as output, as shown in Table 6. Additionally to the predefined definitions from the XML files, TraCI allows clients to interact with the simulation during runtime, e.g., to adjust parameters (like routes, traffic light states and speed limits), track entities and retrieve detector values. When the simulation is finished, SUMO retrieves the overall simulation results and writes result files as configured in the *osm.sumocfg* file.

Table 6. Overview of basic XML files for SUMO

Overview of basic XML files for SUMO	
xml file	Description
osm.net.xml	street network with edges lanes and traffic lights
osm.[x].trips.xml	predefined trips for different vehicles / pedestrians
routes.xml	route definition for the vehicles / pedestrians
osm.sumocfg	master file which imports all the other files
osm.view.xml	GUI view settings
osm.poly	polygons (areas and buildings) and points of interest

4.2.4 Traffic Flow Optimization

RID 4.1 and RID 4.2: The requirements on performing traffic flow optimizations are completely fulfilled. The current state of the sensors can be queried and simulation-parameters can be modified at each time-step during the run-time of the simulation via TraCI. **RID 4.3**: The light signal channel does not have a feedback mechanism, but this can be modelled via the PLC. Thus, the rule-based framework initially communicates with the PLC, which serves as the interface to the traffic light system.

5 Concept for SUMO Extension

in the next Section, the *technical concept* is presented that meets the described requirements of the *5G-trAAffic* project. The concept comprises components and interfaces that are required to transfer the reality from the real world project to the virtuality of SUMO. Figure 4 shows the *technical concept*, which is divided into three perspectives: the *reality*, the *modeling* and the *virtuality*. Finally, for rounding off, the fulfilled or not fulfilled requirements are mapped to the technical concept.

5.1 Extension Points

There are two possible extension points in SUMO shown in Figure 5. The first extension point is to use self-provided applications by SUMO and the second extension point is to connect self-developed components via *TraCI*. The two applications *osmWebWizard* and *netconvert* are provided by SUMO. Both can be used to import map materials into SUMO and fulfills *RID 1*. In the progress of the project it may turn out that other applications can be useful, this is represented by the three dots in *Applications*. For importing the map material *osmWebWizard* will be used in the *technical concept* to

[6] https://www.openstreetmap.org

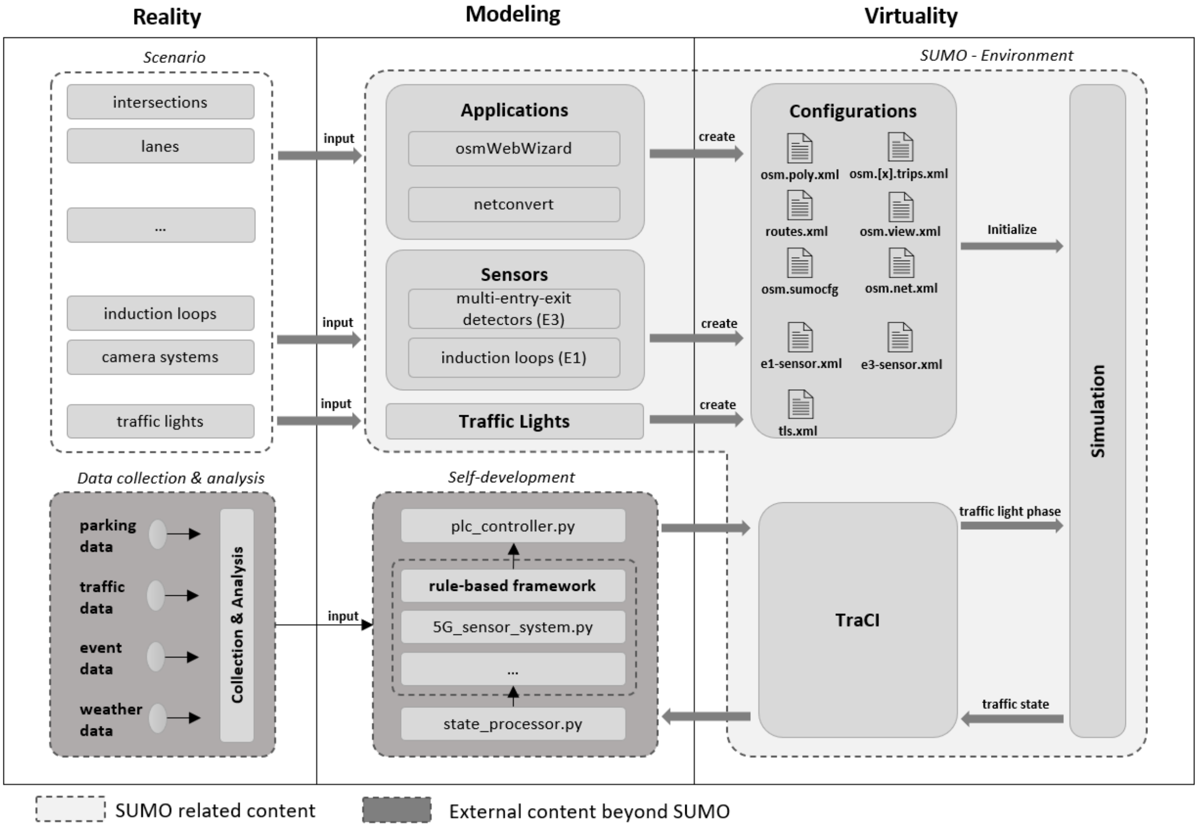


Figure 4. Technical Concept

import, e.g., streets or intersections. The second extension point TraCI which is provided by SUMO is to integrate third-party (external) applications. With respect to TraCI, it allows to interact with a traffic simulation at runtime via programming scripts. Python scripts, for example, can be used to emulate a PLC or the 5G sensor system. The PLC from the real intersection environment is not linked so an own implementation is needed. Furthermore, also states of traffic intersections can be retrieved and suggestions of a rule-based framework can be send back to a traffic light system within the simulation.

5.2 Technical Concept

The two extension points are used to close the identified gaps from chapter 4.2 and convert them to a technical concept.

With reference to Figure 4, for understanding an existing real scenario which needs to be modelled, the *reality* perspective helps to analyse the real-world with its infras-

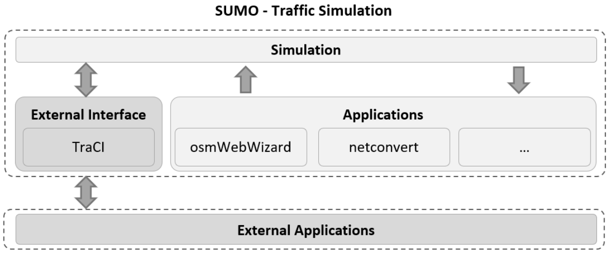


Figure 5. Overview of SUMO’s extension points

tructure such as streets, intersections and sensor systems (e.g, traffic signals or other detectors). Furthermore, this perspective gives also insights into relevant data that can be collected from different sources such as vehicles, pedestrians, parking sensors or weather sensors.

After the analysis of the reality, it is possible to create a model from the reality. For this, the *modeling* perspective offers different possibilities to map the reality into the virtual world using different applications or SUMO build-in detectors. This perspective shows two different areas for possible modeling approaches that are necessary to meet our defined requirements of the real-world scenario. The first area refers to the build-in applications, like sensors and the traffic lights that SUMO offers for modelling. To meet the defined requirements, we used the following different modeling approaches. Beside the *osmWebWizard* to integrate roads or intersections, we modelled different lanes and induction loops by using the *multi-entry-exit detectors* and *induction loops*. The existing project's related traffic light programs are, finally, modelled by the SUMO *Traffic Lights* tool including the traffic light programs itself, the traffic light phases, and the state transitions. The second area is related to the *Self-development* of external components which are not or partly included within SUMO.

Finally, the *virtuality* perspective represents the virtual area where different configuration files map the reality and TraCI provides to interact with the the simulation during runtime via an interface between the *Self-Development* and the *Simulation*. The configuration files contain the modeling aspects from the perspecitve *modeling* that arise from the individual applications such as *osmWebWizard*. For instance, they describe the traffic light programs, phases and state transitions for the *Traffic Lights*. The configuration files are XML files that are imported when the simulation is generated.

5.3 Mapped Requirements

In the following, it is summarized which and how the defined requirements are mapped and assigned to the presented technical concept which need to be implemented into the *5G-trAAffic* project.
Traffic Light System Requirements: The requirements RID 1.1 - 1.4 are already fulfilled by SUMO by default. By configuring the XML file, the traffic light programs and phase transitions can be modeled. At runtime, phase changes can be modified via *TraCI*. For the fulfillment of the requirement RID 1.5 an own development is planned. For this purpose, a *plc_controller.py* will be developed to interface the *rule-based framework* and interact with the simulation via *TraCI*.
5G Sensor Systems Requirements: The *Request Button* from the RID 2.1 requirement must be emulated by a self-development, based on an already existing tutorial[22], which shows in detail, how this emulation is performed with *TraCI*. The requirement RID 2.2 is fulfilled and all relevant functions for the induction loop are available. The camera system, which is described in RID 2.3, can on the one hand be mapped via the *multi-entry-exit detectors*, but must be extended by the in-house development. The emulation of the extension is located in the *Self-Development* block in *5G_sensor_system.py*. In addition, the RID 2.4 requirement is fulfilled here, the emulation of the *road side units* as well as the communication between the vehicles.
Traffic Simulation Requirements: The applications provided by *SUMO* meet the requirements RID 3.1 and 3.2. Via *osmWebWizard* the map material can be imported and provided to the simulation as a configuration file to initialize the environment. Requirement RID 3.3 is represented by the *multi-entry-exit detector*, but the extended functionality is represented in the file *5G_sensor_system.py*.

Traffic Flow Optimization Requirements: With *TraCI* the requirements RID 4.1 to 4.3 can be fulfilled. This allows the current state to be retrieved in each time step and individual states to be manipulated.
Traffic Flow Optimization - Planned Process: Starting from the *rule-based framework*, the generated data from the *reality* is included and additionally the emulated data from the *5G_sensor_system.py*. Based on this data a proposal for the next traffic light phase is generated. This proposal is forwarded to the *plc_controller.py*. Here it is decided whether the proposal is accepted or rejected. If the proposal is accepted, the new traffic light phase is sent to the *Simulation* via *TraCI*. The *Simulation* sends via *TraCI* the state of the simulation, which is passed to the *state_processor.py*. The received state is processed and serves as additional input for the *rule-based framework*.

6 Conclusion

In this work, a technical concept was shown that represents an approach how to transfer a running real-world project comprising different types of complex traffic questions into SUMO in order to implement a later traffic flow optimization using a rule-based framework. The described real-world project comprises sensor systems, roads, intersections, lanes and a traffic light system. Based on this, concrete project-based requirements were derived and compared with the current available functions and tools of SUMO. This requirement analysis was based on the official SUMO documentation. It revealed some gaps which need to be closed in order to make the project a success regarding the traffic optimization goals. The result of the gap analysis was then used as input to define a technical concept showing how SUMO can be extended by self-developed extension in order to meet the unfulfilled or only partly fulfilled requirements. The technical concept reveals different perspectives (i.e., reality, modeling and virtuality) with different aspects that are relevant to meet the defined project requirements. The technical concept combines the usage of SUMO's built-in applications as well as the integration of self-developed programs by using TraCI. At the beginning of the work, we expected a higher demand of self-development. We intensively studied the available features of the SUMO environment and were positively surprised that most of the requirements are already fulfilled by SUMO. In our future work we are going to implement the described concept. A later implementation will show how the integration of the self-developed components works in combination with the processing of real-time data from the real-world scenario. A functional and evaluated rule-based framework will be created in the running project *5G-trAAffic* and the results will be presented as follow-up. To complete the picture, SUMO already offers different tools and applications to model many aspects with respect to traffic flow optimization. Therefore, many derived requirements from the real-world scenario were already covered by SUMO. Additional features that include current and future developments of technologies that include new protocols (e.g., a MQTT), more complex traffic logic, or more complex communication functions or associated logic (e.g., Vehicle-to-X) should be considered. However, this work has shown that SUMO enables and makes it possible to model and simulate complex traffic aspects and offers TraCI to apply and integrate self-developed traffic logic, functions and tools.

7 Acknowledgements

This work is supported by the German Federal Ministry of Transport and Digital Infrastructure (BMVI) within the *5G-Umsetzungsförderung im Rahmen des 5G-Innovations-*

programms funding program under grant no. 45FGU116_C with a total funding amount of 2.600.000,00 €.

References

[1] C.-j. Liu, Z. Liu, Y.-j. Chai, T.-t. Liu, and D. F. Llorca, "Review of virtual traffic simulation and its applications," *Journal of Advanced Transportation*, vol. 2020, p. 8 237 649, 2020, ISSN: 0197-6729. DOI: 10.1155/2020/8237649.

[2] R. Aleksander and C. Paweł, "Recent advances in traffic optimisation: Systematic literature review of modern models, methods and algorithms," *IET Intelligent Transport Systems*, vol. 14, no. 13, pp. 1740–1758, 2020, ISSN: 1751-956X. DOI: 10.1049/iet-its.2020.0328.

[3] N. K. Jain, R. K. Saini, and P. Mittal, "A review on traffic monitoring system techniques," in *Soft Computing: Theories and Applications*, K. Ray, T. K. Sharma, S. Rawat, R. K. Saini, and A. Bandyopadhyay, Eds., Springer Singapore, 2019, pp. 569–577, ISBN: 978-981-13-0589-4.

[4] S. Dorokhin, A. Artemov, D. Likhachev, A. Novikov, and E. Starkov, "Traffic simulation: An analytical review," *IOP Conference Series: Materials Science and Engineering*, vol. 918, no. 1, p. 012 058, 2020, ISSN: 1757-8981. DOI: 10.1088/1757-899X/918/1/012058.

[5] Q. Ma, H. Zeng, Q. Wang, and S. ullah, "Traffic optimization methods of urban multi-leg intersections," *International Journal of Intelligent Transportation Systems Research*, vol. 19, no. 2, pp. 417–428, 2021, ISSN: 1868-8659. DOI: 10.1007/s13177-020-00245-y.

[6] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582. DOI: 10.1109/ITSC.2018.8569938.

[7] J. Jin, X. Ma, and I. Kosonen, "An intelligent control system for traffic lights with simulation-based evaluation," *Control Engineering Practice*, vol. 58, pp. 24–33, 2017, ISSN: 09670661. DOI: DOI:10.1016/j.conengprac.2016.09.009.

[8] M. A. Dian Khumara, L. Fauziyyah, and P. Kristalina, "Estimation of urban traffic state using simulation of urban mobility(sumo) to optimize intelligent transport system in smart city," in *2018 International Electronics Symposium on Engineering Technology and Applications (IES-ETA)*, 2018, pp. 163–169. DOI: 10.1109/ELECSYM.2018.8615508.

[9] M. E. M. Ali, A. Durdu, S. A. Celtek, and A. Yilmaz, "An adaptive method for traffic signal control based on fuzzy logic with webster and modified webster formula using sumo traffic simulator," *IEEE Access*, vol. 9, pp. 102 985–102 997, 2021. DOI: 10.1109/ACCESS.2021.3094270.

[10] F. IOSB-INA, *Datenbasierte Verkehrsflussoptimierung*, URL: https://lemgo-digital.de/index.php/de/mobilitaet-projekte/89-verkehrsoptimierung, accessed 2023-02-04, 2019.

[11] T. Ishu, S. Indu, and P. Neeta, "Review of traffic light synchronization for intelligent vehicles: Current status, challenges, and emerging trends," *Electronics 2022*, vol. 11(3), no. 465, 2022. DOI: 10.3390/electronics11030465.

[12] K. Pandey and P. Jalan, "An approach for optimizing the average waiting time for vehicles at the traffic intersection," in *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2018, pp. 30–35. DOI: 10.1109/PDGC.2018.8745757.

[13] S. C. Lobo, S. Neumeier, E. M. G. Fernandez, and C. Facchi, "Intas – the ingolstadt traffic scenario for sumo," 2020. DOI: 10.48550/ARXIV.2011.11995. [Online]. Available: %5Chref%7Bhttps://arxiv.org/abs/2011.11995%7D%7Bhttps://arxiv.org/abs/2011.11995%7D.

[14] L. L. Jacobus Louw and T. Woodley, “A Comparison of Reinforcement Learning Agents Applied to Traffic Signal Optimisation,” *SUMO Conference Proceedings*, vol. 3, pp. 16–43, 2022. DOI: 10.52825/scp.v3i.116.

[15] M. Halbach and J. Erdmann, “High fidelity modelling of traffic light control with xml logic representation,” *SUMO Conference Proceedings*, vol. 3, pp. 45–68, 2022. DOI: 10.52825/scp.v3i.114.

[16] German Aerospace Center (DLR) and others, *TrafficLights in SUMO*, URL: https://sumo.dlr.de/docs/Simulation/Traffic_Lights.html, accessed 2023-02-03, 2023.

[17] German Aerospace Center (DLR) and others, *netconvert*, URL: https://sumo.dlr.de/docs/netconvert.html, accessed 2023-02-03, 2023.

[18] German Aerospace Center (DLR) and others, *OSMWebWizard*, URL: https://sumo.dlr.de/docs/Tutorials/OSMWebWizard.html, accessed 2023-02-08, 2022.

[19] German Aerospace Center (DLR) and others, *netgenerate*, URL: https://sumo.dlr.de/docs/netgenerate.html, accessed 2023-02-03, 2023.

[20] German Aerospace Center (DLR) and others, *TrafficLights State*, URL: https://sumo.dlr.de/docs/TraCI/Change_Traffic_Lights_State.html, accessed 2023-02-08, 2023.

[21] German Aerospace Center (DLR) and others, *MultiEntryExitDetectorsE3*, URL: https://sumo.dlr.de/docs/Simulation/Output/Multi-Entry-Exit_Detectors_(E3).html, accessed 2023-02-03, 2023.

[22] German Aerospace Center (DLR) and others, *Pedestrian Crossing with TraCI*, URL: https://sumo.dlr.de/docs/Tutorials/TraCIPedCrossing.html, accessed 2023-04-12, 2023.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.219

Published: 29 June 2023

A Framework for Simulating Cyclists in SUMO

Heather Kaths[1][https://orcid.org/0000-0003-2554-8243] and Aboozar Roosta[1][https://orcid.org/0000-0003-1023-8189]

[1] Bergische Universität Wuppertal, Wuppertal, Germany

Abstract. Cyclists pose an interesting challenge in the microscopic modeling and simulation of urban traffic. Like motorists, cyclists can move on roadways, tend to have one main axis of movement (longitudinal), and cannot change their velocity instantaneously. However, like pedestrians, cyclists are less bound by lane discipline and are often less rule-oriented than motorists. They flexibly adjust their lateral position within a lane, fluidly move between different types of infrastructure (bicycle lane, sidewalk, roadway), and tactically select their pathways across intersections. Their interactions with other road users are more intuitive and less defined by the lane markings. How should the behavior of such adaptable road users be modeled? In SUMO, modifications to the simulation environment enable the application of car-based models to cyclists. A driving lane is divided into multiple sub-lanes along the longitudinal axis. Lane change and car-following models can be calibrated and applied to simulate realistic bicycle and mixed traffic using this approach. However, the flexible nature of cyclists, particularly at intersections or when switching between different types of infrastructure, is difficult to simulate. A modeling framework for linking the paradigms used to simulate motor vehicle traffic (one-dimensional lane-based models) and pedestrian traffic (two-dimensional social force type models) is presented. Guidelines are used to lead each cyclist through the network while they move freely on a two-dimensional plane, their movement and interactions governed by an adapted social force model. The conceptual framework and an openly available Python package *CyclistModel* are introduced, and advantages and possible use cases are discussed.

Keywords: Bicycle Traffic, Mixed Traffic, Non-Lane-Based Traffic, Social Force

1. Introduction

The earliest microscopic simulation tools, which emerged in the 1960's, were comprised of cars that progressed through road networks by traveling single-file along lanes, initially by hopping from one cell to the next [1]. Despite the tremendous advances in the field of traffic simulation since then, most simulation software still operate around the concept of the driving lane [2], [3]. In lane-based tools, motion is simulated using vehicle dynamics models, preference models (e.g. desired speed and acceleration), models of traffic rules and regulations and traffic signal control models. Interactions with other road users are reduced to one-dimensional, single-file following behavior, lane selection, (cooperative) lane changing, and yielding at points of conflict.

This relatively simple paradigm is very powerful for designing and evaluating road infrastructure, planning traffic signal control, developing Cooperative Intelligent Transport Systems (C-ITS), and investigating automated vehicle traffic, to name a few fields of application. Unfortunately, lane-based simulation tools are inherently bad at simulating other types of road users with different, more fluid ways of moving and interacting, such as pedestrians and cyclists.

As simulation became a more accepted tool for traffic analysis in the 1990s, approaches for simulating pedestrians emerged [1]. In contrast to motorists, pedestrian traffic is not always characterized by a common direction of travel. Pedestrians (typically) do not move single-file but rather exhibit free motion on a two-dimensional plane, changing their velocity almost instantaneously. In consideration of these major differences, a new type of model based on internal or social forces was proposed. The original social force model for pedestrian dynamics was introduced by Helbing and Molnar [4] and conceptualized the motion of a pedestrian as the result of multiple internal motivations. These internal motivations or forces include a driving force towards the (interim) destination, repulsive forces from other pedestrians, obstacles, and boundaries, as well as attractive forces to friends and points of interest. In each simulation step, an acceleration vector is calculated based on these attractive and repulsive forces that move the pedestrian through a continuous two-dimensional space. Over the 30 years since the original formulation of this model, there have been many further developments, extensions, and divergent models [5], [6]. Researchers have formulated social force models specifically for bicycle traffic [7], [8], the most notable of which is the model developed by Schönauer et al. [9]. In the Schönauer et al. model, operational behavior is modeled using an adapted social force model in which the degrees of freedom are limited based on the single-track model for car dynamics. Road users' pathways are determined using an infrastructure force field model.

Cyclists' behavior falls on a spectrum between the movement and interaction patterns of motor vehicle traffic and those of pedestrians. They can adjust their behavior along this spectrum to adapt to the current situation. At one end, cyclists ride quickly in one direction with little lateral movement while sharing a roadway with motor vehicle traffic. On the other end, cyclists move slowly and are ready to deviate from their pathways in spaces shared with many pedestrians. The SUMO user documentation indicates that there are no bicycle-specific models available and cyclists are to be simulated as 'fast pedestrians' or 'slow vehicles' [10].

Most of the extensions and further developments to microscopic traffic simulation software since the 2010s, including SUMO, have focused on the bicycle as a 'slow vehicle' option to enable the more realistic inclusion of cyclists. These extensions are also relevant for other types of road users, such as users of micromobility modes (e.g. e-kick-scooters), who are less bound by lane discipline. In SUMO, the main extension is the sub-lane model, which allows for the longitudinal division of driving lanes into multiple sub-lanes, the width of which can be defined by the user. The width of each road user relative to the width of the sub-lanes determines the number of sub-lanes occupied. In Figure 1, the original lane-based (left) and sub-lane approaches for simulating mixed-traffic streams are depicted.

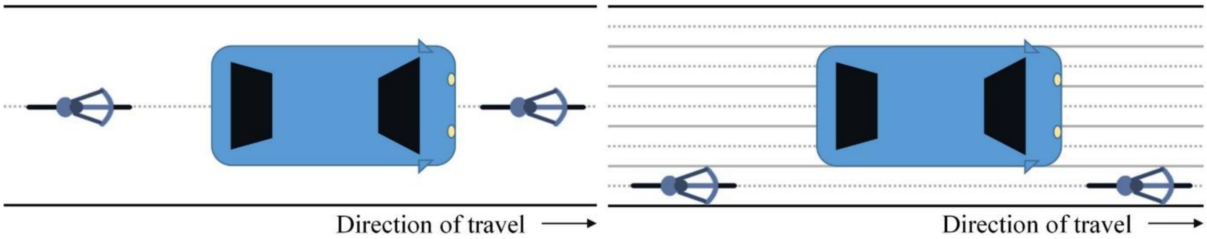


Figure 1. Original lane-based (left) and sub-lane (right) approach for simulating mixed traffic in SUMO.

Essentially, in the sub-lane approach, car-following models are used to simulate the reactions of a following road user to the speed and position of a leading road user in the same sub-lane (or one of the same sub-lanes if the width of the road user covers more than one sub-lane). Interactions between road users in adjacent sub-lanes are modeled using lane selection and lane change models. This approach enables narrower road users to pass and be passed in a single driving lane and makes it possible to simulate traffic in situations with little lane discipline, even amongst motorists.

Although the sub-lane approach permits modeling bicycle and mixed traffic flows with much more realism than the conventional single lane approach (see example in Figure 1, left), it is still difficult to capture the inherent flexibility of cyclists and users of micromobility modes in simulations.

Why is it important to simulate the movement and interactions of cyclists realistically? First, in reaction to the persisting and worsening effects of overdependence on private motor vehicles, cities and regions are strongly promoting and building for more sustainable modes of transport, including cycling and micromobility. The number of cyclists, e-kick-scooter users, and e-moped users is increasing as a result. For example, in German metropolitan regions, the modal share of cycling increased from 9 % in 2002 to 17 % in 2017 [11]. Considering this trend and the goal of the German government to double the number of kilometers traveled by bicycle by 2030 compared to 2017 [12], the volume of bicycle traffic is expected to rise drastically in urban areas. In this future, it will no longer be defendable to develop any C-ITS, automated driving, or future technological system for urban (motor vehicle) traffic without including bicycles in the research and development process. Not only will it be (more) necessary to consider bicycles in the development of future technologies for motor vehicle traffic, but it is likely (and hoped for) that the focus of technological development will shift largely to active and micromobility. In this future scenario, research questions concerning the dimensioning and design of road infrastructure for enormously high volumes of mixed bicycle traffic, the necessity and programming of traffic signal control as well as road user safety could be investigated using microscopic traffic simulation. This will only be possible if microscopic traffic simulation tools and the underlying behavior models accurately simulate bicycle and mixed traffic flows.

2. Conceptual framework

A straightforward approach for establishing compatibility between the one-dimensional lane-based and the two-dimensional force-based simulation paradigms and thus enabling the more realistic simulation of non-lane-based traffic is presented in this paper. The term *guideline* is used to describe the intended pathways of a cyclist or other non-lane-based road user and was coined with this specific meaning in the doctoral thesis *Development of tactical and operational behaviour models for bicyclists based on automated video data analysis* by Heather Twaddle (now Heather Kaths). A guideline is a polyline that a road user intends to follow to reach their interim destination, congruent in form to the center of a driving (sub-)lane.

Conceptually, guidelines are the result of conscious decisions made by a cyclist or other non-lane-based road user at the tactical behavior level. At this level, road users decide how to act in order to best cope with the current situation [13]. This includes making movement and interaction plans on a time horizon of seconds to minutes under the consideration of the speed and position of nearby road users, the form and state of the road infrastructure, and the phase of traffic signals, as well as many other factors. Based on this conceptual definition, an entire route through a network is comprised of numerous successive guidelines, again congruent to the centerlines of lanes in lane-based simulation tools. Indeed, if the non-lane-based road user simulated using the proposed approach does not encounter any impedance from other road users or obstacles, they will proceed along the guideline as they would the centerline of a (sub-)lane.

So far, the congruity with the (sub-)lane-based simulation paradigm is clear. The presented model diverges from the lane-based approach in that the road user is not bound to the guideline and does not travel along the line in one-dimension. Rather the guideline is used to determine the interim destination in the next simulation step to use as input for a social force-like behavior model. The point on the guideline closest to the front of the road user $l(t)$ is located. Then, a point along the guideline in the direction of travel $l(t+1) = l(t) + V^0$ is selected, where V^0 is a scalar value that is at least as large as the distance that will be travelled in the next simulation step. The magnitude of V^0 determines how closely the road user follows the guideline. An example of using the guideline to determine the interim destination in the next simulation step is shown in Figure 2.

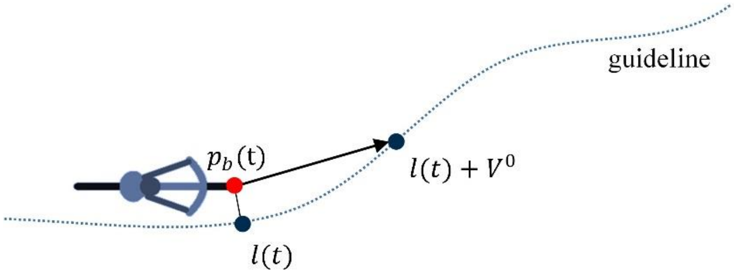


Figure 2. Locating the next interim destination using a guideline.

The model used to generate the movement and interactions of cyclists and other non-lane-based road users belongs to the family of social force models. Twaddle [14] formulated a model for cyclists' movement and interactions based on the NOMAD model for pedestrian dynamics [15], [16]. The basic formulation of the NOMAD model is:

$$a_p(t) = \frac{v_p^0 - v_p(t)}{T_p} - A_p \sum_{q \in Q_p} u_{pq}(t)\, e^{\frac{-d_{pq}(t)}{R_p}} \tag{1}$$

where:

$$d_{pq}(t) = \left\| r_q(t) - r_p(t) \right\| \tag{2}$$

$$u_{pq}(t) = \frac{r_q(t) - r_p(t)}{d_{pq}(t)} \tag{3}$$

The desired velocity v_p^0 is towards the desired destination of pedestrian p. The velocity $v_p(t)$ and the position $r_p(t)$ are two-dimensional vectors at instant t. The set of pedestrians in a defined vicinity of pedestrian p is given by Q_p. Four pedestrian-specific parameters, the desired speed V_p^0, the necessary time for acceleration T_p, the interaction factor A_p and radius of interaction R_p, are defined. The model was calibrated using trajectory data extracted from video data at four intersections in Munich. Guidelines were generated by clustering the observed trajectories and using the centroid trajectory of each cluster as the guideline for all the cyclists in the cluster. This is a shaky presumption, but it made it possible to calibrate the model parameters using Maximum Likelihood Estimation (MLE). More detailed information about the model specification, calibration and validation can be found in Twaddle [14].

Because the simulated cyclists and other non-lane-based road users are no longer constrained to a lane in the simulation, it is necessary to define boundaries that cannot be crossed as well as other obstacles that must be avoided. For example, if an on-road bicycle lane is shouldered by a steep curb, green strip, or roadside parking, the right side boundary of the bicycle lane must be simulated as an obstacle to prevent crossing.

Practically, the centerlines of (sub-)lanes can be extracted from a SUMO simulation using TraCI and used to act as guidelines in this approach. Alternatively, guidelines can be defined by the user based on observed motion patterns. The definition of unique guidelines is likely particularly relevant at intersections where the behavior of cyclists tends to stray most from the planned infrastructure use. A desire line analysis [17]–[19], which examines the forms of unique pathways used by cyclists to cross an intersection (or carry out any other type of maneuver) can be useful for creating guidelines.

3. Implementation and first results

The original model formulated based on the concept of guidelines and social force was demonstrated in SUMO using the TraCI Python interface by Twaddle [14]. Simulations of the research intersections were built in SUMO and the centroids of the trajectory clusters were extracted, smoothed, and used as guidelines as described above.

CyclistModel is an open-source Python package that uses TraCI to integrate the guideline/social force model approach in SUMO. *CyclistModel* is based on the original Python code developed by Twaddle [14] and can be accessed here https://github.com/HeatherAnne85/CyclistModel. A flowchart of the basic functionality of CyclistModel is shown in Figure 3.

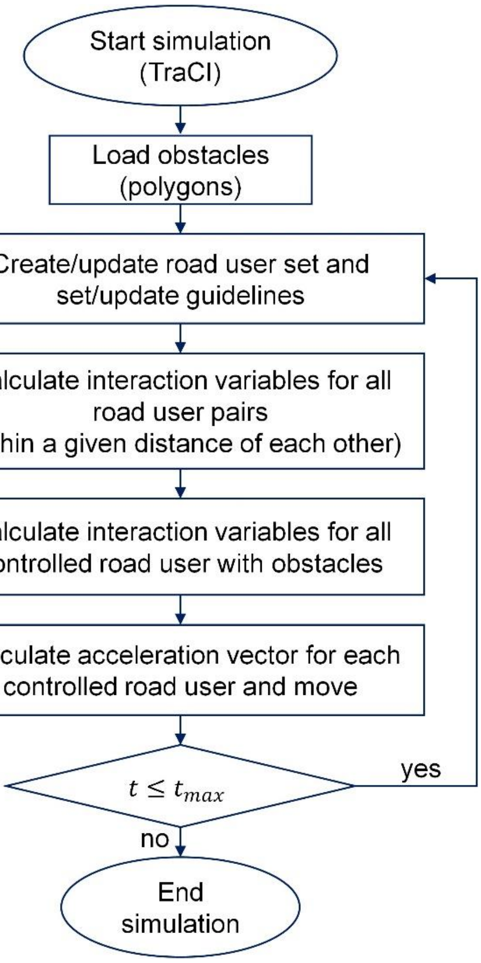

Start simulation (TraCI)
Load obstacles (polygons)
Create/update road user set and set/update guidelines
Calculate interaction variables for all road user pairs (within a given distance of each other)
Calculate interaction variables for all controlled road user with obstacles
Calculate acceleration vector for each controlled road user and move
$t \leq t_{max}$
yes
no
End simulation

Figure 3. Computational steps in *CyclistModel.*

Currently, *CyclistModel* extracts the centerline of each lane along the route of a cyclist to use as the guideline for the adapted NOMAD model. Polylines and polygons created in SUMO are imported through TraCI and are integrated as obstacles in the simulation. Figure 4 shows an example of the same simulation of a 1.5 m wide bicycle facility with two variations in the placement of the non-crossable boundary (red line). In the simulation on the left, the bicycle facility is on the roadway and a curb prevents cyclists from deviating onto the sidewalk. As a result, cyclists move onto the roadway to carry out passing maneuvers (if traffic permits). The image on the right shows the non-crossable boundary on the left shoulder of the bicycle lane and a resulting passing maneuver using the sidewalk.

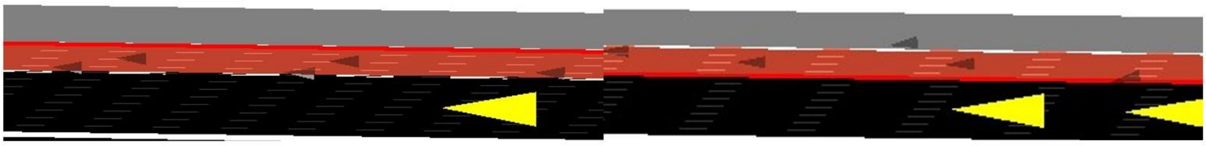


Figure 4. Example with a non-crossable boundary between the bicycle lane and sidewalk (left) and bicycle lane and roadway (left) and resulting passing behavior.

Qualitatively, the presented approach enables the simulation of the higher flexibility and more fluid interactions characteristic of bicycle traffic. However, it is difficult to use standard methods for assessing validity to determine if this method is more capable of realistically simulating bicycle traffic than the sub-lane approach. Macroscopic parameters that are typically used to evaluate microscopic traffic simulations, such as average delay time, total travel time, and speed distribution, do not allow for the assessment of flexible infrastructure use (such as moving against the given direction of travel, using unintended infrastructure (such as the sidewalk) and crossing using unexpected maneuvers. Depending on the level of realism necessitated by the simulation study research question, model calibration and validation using parameters that capture behaviors characteristic of cyclists and other non-lane-based road users is necessary.

Some work has been done in this area for mixed traffic in countries where lane discipline is less strong and traffic is more heterogeneous (e.g. India, see for example [3]) that is useful for defining parameters for the calibration of mixed traffic simulations. These tend to build on car traffic-based approaches and do not account for the spatial distribution of the parameters. Twaddle [14] proposed evaluating the simulation by dividing the intersection into a grid of small cells (1.5m by 1.5m was used in the original publication, Figure 5).

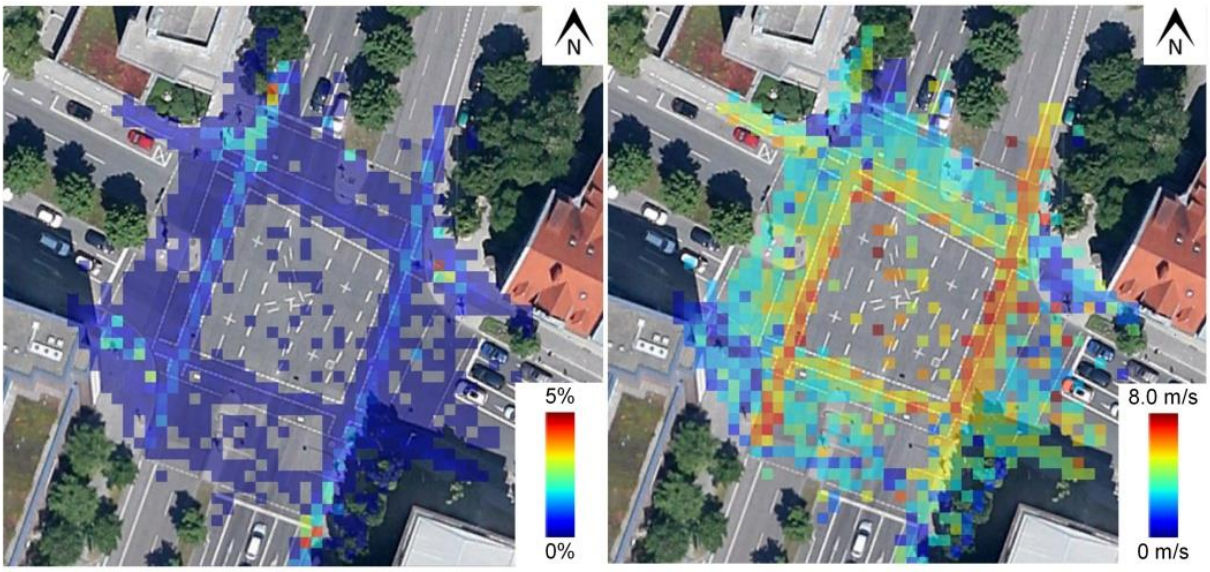


Figure 5. Examples of cell grids for the evaluation of microscopic traffic simulations (Twaddle [14]).

Parameters such as average, minimum and maximum speed, occupancy and density in each cell can be calculated from simulated and observed trajectories and compared. Examples of heat maps produced from trajectory data from an intersection in Munich, Germany are shown in Figure 5. The validity of the simulation model can be assessed based on the summed or averaged measure of error (e.g. RMSE) across the cells. As the aim of this paper is to introduce the concept of the guideline/social force approach, no validation is presented here.

Although *CyclistModel* functions as intended, it is still a very rudimentary prototype that requires a great deal of development work to become a functional tool for simulation studies.

The Python package is computationally expensive and very slow, partially because all road users in the simulation are loaded to extract interaction parameters (although interaction parameters are only calculated for road user pairs in close proximity to one another). Extensive further model developments are necessary to capture the dynamics, movements and interactions of cyclists and other non-lane-based road users in a realistic way.

4. Discussion and conclusion

The modeling approach presented in this paper creates a link between lane-based and social force-type models that could allow for the more realistic modeling of flexible road users in lane-based microscopic traffic simulation tools such as SUMO without altering the fundamental set-up of the simulation environment. The behavior of cyclists lies on a spectrum spanning between the rule-based, longitudinally-guided motion patterns of motorists and the continuous, two-dimensional behavior of pedestrians. In reflection of this transitionary nature, the *Cyclist-Model* approach combines attributes of both modeling paradigms. The major advantages of this approach over the lane-based are:

1. The flexible behavior characteristic of cyclists and other non-lane-based road users is easily simulated. For example, the tactical or operational maneuvering between bicycle lanes and adjacent sidewalks or roadways can occur where the infrastructure allows. The choice to change infrastructures does not have to be explicitly modeled. Similarly, at intersections, the variability in pathways used to cross the intersection is greatly increased compared to one-dimensional lane-based models.
2. Cyclists and other non-lane-based road users interact with other road users that are not in their given (sub-)lane of travel. This makes it more possible to capture the more fluid and less rule-stipulated interactions of cyclists and makes difficulties concerning the modeling of conflict areas at intersections obsolete.

Theoretically, this approach makes it possible to capture flexible behaviors. However, is the presented approach superior to the simpler and less computationally expensive sub-lane approach in terms of replicated observed traffic flow? Under what conditions is it beneficial or necessary to accurately and realistically simulate cyclists and other flexible road users? These questions can only be answered by expanding the fundamental knowledge of cyclists' movement patterns and interactions with all kinds of road users under various conditions. Currently, the overall lack of empirical data and fundamental research in this area hinders efforts to calibrate, validate and compare behavioral models for bicycle traffic.

Here, the NOMAD social force model was applied because of the straightforward formulation and the inclusion of anisotropic and velocity anisotropic behavior. However, if social force-type models are to be applied to bicycle traffic, a great deal of observational and experimental work is needed to advance specification, calibration and validation. It could also very well be that other types of models are better able to replicate the behavior of non-lane-based road users.

Further developments and improvements to the Python package *CyclistModel* are needed. So far, the simulated cyclists interact with one another based on internal social forces and do not take into account traffic rules or regulations or signal control. Fundamental knowledge concerning how cyclists take these factors into account in moving through the environment, behavior models based on these findings and additional code in *CyclistModel* are all necessary to make the simulated cyclists behave realistically at intersections.

It could be beneficial to use both the sub-lane approach and the guideline/social force approach for modeling bicycle traffic at different locations in the same simulation. For example, cyclists traveling on a separated bicycle lane that has some degree of physical separation from both pedestrian and motor vehicle traffic could be simulated realistically using the sub-lane approach. At intersections, however, it may be necessary to switch to the guideline/social force approach to reach an acceptable level of realism. This is easily doable with *CyclistModel* with some small changes in the code.

Data availability statement

The data used to calibrate and validate the original version of the Cyclist Model from Twaddle [14] is available upon request at the Chair of Traffic Engineering and Control at the Technical University of Munich.

Underlying and related material

CyclistModel is an open-source Python package that uses TraCI to integrate the guide-line/social force model approach in SUMO. CyclistModel is based on the original Python code developed by Twaddle [14] and can be accessed here https://github.com/HeatherAnne85/CyclistModel.

Author contributions

Heather Kaths contributed to the paper in the following ways: conceptualization, methodology, data curation, formal analysis, visualization, software, and writing the original draft. Aboozar Roosta contributed in the following ways: software and reviewing and editing the manuscript.

Competing interests

The authors declare that they have no competing interests.

References

[1] E. B. Lieberman, "Brief history of traffic simulation," *Traffic Transp. Simul.*, vol. 17, 2014.

[2] M. Sekeran, M. Rostami-Shahrbabaki, A. A. Syed, M. Margreiter, and K. Bogenberger, "Lane-Free Traffic: History and State of the Art," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 1037–1042.

[3] G. Asaithambi, V. Kanagaraj, and T. Toledo, "Driving Behaviors: Models and Challenges for Non-Lane Based Mixed Traffic," *Transp. Dev. Econ.*, vol. 2, no. 2, p. 19, 2016, doi: https://www.doi.org/10.1007/s40890-016-0025-6.

[4] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Phys. Rev. E*, vol. 51, no. 5, p. 4282, 1995.

[5] X. Chen, M. Treiber, V. Kanagaraj, and H. Li, "Social force models for pedestrian traffic–state of the art," *Transp. Rev.*, vol. 38, no. 5, pp. 625–653, 2018.

[6] A. Schadschneider, H. Klüpfel, T. Kretz, C. Rogsch, and A. Seyfried, "Fundamentals of pedestrian and evacuation dynamics," in *Multi-Agent Systems for Traffic and Transportation Engineering*, IGI Global, 2009, pp. 124–154.

[7] M. Li, F. Shi, and D. Chen, "Analyze bicycle-car mixed flow by social force model for collision risk evaluation," *3rd Int. Conf. Road Saf. Simul.*, pp. 1–22, 2011.

[8] X. Liang, B. Mao, and Q. Xu, "Psychological-Physical Force Model for Bicycle Dynamics," *J. Transp. Syst. Eng. Inf. Technol.*, vol. 12, no. 2, pp. 91–97, 2012, doi: https://www.doi.org/10.1016/S1570-6672(11)60197-9.

[9] R. Schönauer, M. Stubenschrott, W. Huang, C. Rudloff, and M. Fellendorf, "Modeling concepts for mixed traffic: Steps toward a microscopic simulation tool for shared space zones," *Transp. Res. Rec. J. Transp. Res. Board*, vol. 2316, no. 1, pp. 114–121, 2012.

[10] German Aerospace Center (DLR) and others, "SUMO Documentation, 'Bicycle Simulation,'" 2022. https://sumo.dlr.de/docs/Simulation/Bicycles.html.

[11] C. Nobis, "Analysen zum Radverkehr und Fußverkehr," *Mobilität Deutschl. - MiD*, p. 84,

2019, [Online]. Available: http://www.mobilitaet-in-deutschland.de/pdf/MiD2017_Analyse_zum_Rad_und_Fussverkehr.pdf.

[12] PTV Group, ifok GmbH, and Fraunhofer-Institut für System- und Innovationsforschung ISI, “Nationaler Radverkehrsplan 3.0,” Berlin, 2021. Accessed: Jul. 09, 2021. [Online]. Available: https://nationaler-radverkehrsplan.de/de/bund/nationaler-radverkehrsplan-nrvp-2020.

[13] J. A. Michon, “A critical view of driver behavior models: what do we know, what should we do?,” *Hum. Behav. traffic Saf.*, pp. 485–520, 1985, doi: https://www.doi.org/10.1007/978-1-4613-2173-6.

[14] H. A. Twaddle, “Development of tactical and operational behaviour models for bicyclists based on automated video data analysis.” Technische Universität München, 2017.

[15] S. P. Hoogendoorn and W. Daamen, “Microscopic calibration and validation of pedestrian models: Cross-comparison of models using experimental data,” in *Traffic and Granular Flow’05*, Springer, 2007, pp. 329–340.

[16] S. P. Hoogendoorn, “Normative pedestrian flow behavior theory and applications,” Delft University of Technology, Faculty Civil Engineering and Geosciences, 2001.

[17] A. Lind, J. Honey-Rosés, and E. Corbera, “Rule compliance and desire lines in Barcelona’s cycling network,” *Transp. Lett.*, vol. 13, no. 10, pp. 728–737, 2021.

[18] University of Amsterdam and Copenhagenize Design Co., “The Desire Lines of cyclists in Amsterdam,” Amsterdam, 2014.

[19] M. S. Wexler and A. El-Geneidy, “Keep’em separated: Desire lines analysis of bidirectional cycle tracks in montreal, canada,” *Transp. Res. Rec.*, vol. 2662, no. 1, pp. 102–115, 2017.

[20] G. Grigoropoulos *et al.*, “Evaluation of the Traffic Efficiency of Bicycle Highways: A Microscopic Traffic Simulation Study,” 2018.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.221

Published: 29 June 2023

SUMO Simulations for Federated Learning in Communicating Autonomous Vehicles

A Survey on Efficiency and Security

Levente Alekszejenkó[1] [https://orcid.org/0000-0002-3196-1950], and
Tadeusz Dobrowiecki[1] [https://orcid.org/0000-0002-8307-5096]

[1] Department of Measurement and Information Systems,
Budapest University of Technology and Economics,
Budapest, Hungary

Abstract: In transportation, a vehicle's route is one of the most private information. However, to mutually learn some phenomena in a city, for example, parking lot occupancies, we might have to reveal information about it. In this paper, we focus on assessing the privacy loss in a vehicular federated machine learning system. For the analysis, we used the Monaco SUMO Traffic Scenario (MoST). We also used the simulation inputs as statistical data to calculate privacy loss metrics. Results show that a vehicular federated machine learning system may pose a smaller privacy threat than individual learning, but its performance is lower compared to a centralized learning approach.

Due to the vast amount of data and processing time, we also describe a method to build a Docker image of SUMO together with a software client-server architecture for SUMO-based learning systems on multiple computers.

Keywords: federated learning, communicating vehicles, efficiency, security

1. Introduction

Modern vehicles carry an abundance of sensors. Additionally, recent developments in machine learning made it possible to process even more data about the transportation infrastructure than at any time in the past.

As it would be impossible to store all this information, we shall build compact, predictive models to utilize all the knowledge. These models are valuable inputs to optimize the traffic in a city. In the future, these models can also be parts of autonomous driving algorithms.

There are several ways to build machine learning models. In this paper, we consider that vehicles try to learn such phenomena *individually* and as *participants of a federated learning system*. Like in mobile edge networks, federated learning might have many advantages also in vehicular learning, including low latency, privacy, and efficient use of network bandwith [1]. Therefore, with the help of the Monaco SUMO Traffic

Scenario [2] (MoST) simulations in Eclipse SUMO [3], *we analyze the performance of these learning approaches and compare them to a centralized method in which the infrastructure itself collects the data and trains the model.*

Additionally, while sharing data, even in federated learning, we shall be cautious not to share too much personal information, especially route origins, and destinations. As the route is private information, keeping it hidden from unauthorized access is both a data security problem and a legal obligation. In this paper, we also *evaluate the mentioned learning schemes from a data privacy point of view*.

In the following, in Section 2, we review the relevant literature. Section 3 describes the parametrization and properties of the used simulation. The mentioned learning schemes are described in Section 4–6. Finally, Section 7 concludes the paper. Moreover, long-running computations required the dockerization of SUMO and a TraCI script, which we describe in Appendix A. Appendix B, in addition, proposes a method to distribute SUMO and machine learning workers between multiple computers.

For further development, we published our source codes at https://github.com/alelevente/sumo_sec.

2. Related works

Crowdsensing complex traffic phenomena, such as parking place availability [4], seems to be a promising new way of optimizing the traffic flow in cities. However, sharing data on Vehicular Ad-hoc Networks (VANETs) also poses some technical challenges. We have to solve the problem of secure data exchange while respecting the limits of the available network bandwidth [5].

Several papers focus on simulating cyberattacks on Connected Autonomous Vehicles (CAVs) and VANETs, including active [6], [7] or Distributed Denial of Service attacks [8]. These simulations usually use SUMO to evaluate vehicle movements.

Moreover, federated learning [9] also gained importance in machine learning on mobile devices in recent years. Federated learning ensures that its participants shall not exchange their training data. It guarantees a certain level of security and also reduces communication costs. Therefore, a vehicular network can take advantage of the federated scheme as well [10]. Unfortunately, the security level in a federated learning scenario highly depends on identically distributed datasets: with non-IID (not Identically and Independently Distributed) data, it is possible to carry out various attacks against the participants [11]. However, there are numerous countermeasures to such attacks; they might not be applied onboard a vehicle due to the limited amount of power and computational resources.

In this paper, we measure the parking lot occupancy by multiple measurement setups. We evaluate the idea that vehicular crowdsensing schemes can function as *cooperation-based* location privacy-ensuring methods to hide routes, which is a critically privacy-sensitive property [12], of the vehicles. To infer this information, it is enough to suppose a *passive* attack carried out by an honest-but-curious party.

3. Simulation

To obtain measurement results, we used the Eclipse SUMO traffic simulation tool with the MoST scenario. We have changed the simulation timestep from 0.25 s to 1.0 s to reduce the computation time.[1] We parametrized the simulation to have a maximum of 15 minutes of departure offset stochastically for each vehicle. It models that the population follows some daily routine; however, individuals might not depart at the same time each day. That yields an average traffic demand similar to what the MoST defines with a certain perturbance around it.

SUMO is supposed to provide training data on parking lot occupancies measured by moving cars. Hence, during the simulation, we recorded the position of each vehicle with 0.1 Hz frequency through the Traffic Control Interface (TraCI). We also recorded the occupancy of parking lots once each minute. As the MoST originally defined the scenario, the measurement times ranged from 4:00 a.m. to 2:00 p.m.

These measurements were repeated 60 times, corresponding to a measurement series of 3 months of working day data. As running a single simulation takes approximately 1 hour on a PC, we sped it up by running multiple simulations in parallel. To this end, as described in Appendix A, we created a Docker image that runs a TraCI client and a SUMO instance.

We assume that a vehicle *knows* a parking lot's occupancy if it passes through an edge with a center that is not more than 50 m away from the edge of the parking lot. This distance can be understood as the communication or visual range of the cars. After the simulations had terminated, we determined for each vehicle its measured parking lots and the actual occupancy values when the cars were nearby.

4. Centralized learning

We trained a neural network only with the simulated parking lot occupancy data to obtain a baseline model. Figure 1 illustrates the scheme of this learning setup. For large parking garages, it is possible to implement such an approach by collecting each garage's occupation data to train a neural network on a remote server. However, this centralized setup requires an infrastructure that detects whether a parking lot is free or occupied. Hence, real-world implementation would be impractical due to its installation and operational costs.

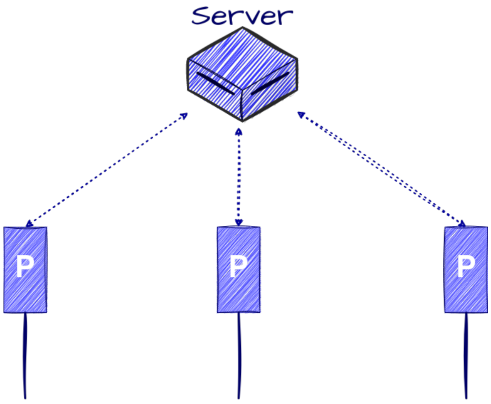


Figure 1. Scheme of the centralized learning approach. The server has connection to each parking lot; hence, it can collect data from them, and use this data for learning a predictive model.

[1] Because of this, accidents might occur in SUMO due to the small τ values. In our case, this is only considered to be a random event without any further investigation.

4.1. Neural network architecture and parameters

The training data consisted of $\{\texttt{ID of parking lot, timestamp, occupancy}\}$ records, from which the first two properties were the training features and the `occupancy` column was the label to predict. We normalized the timestamps (ranging from 4:00 to 14:00) to the $[0.0, 1.0]$ interval, and we standardized the occupancy data (ranging from 0.0 (empty) to 1.0 (full)) by the mean and standard deviation values measured on the first simulated day. As there are relatively few parking lots in the MoST, we were able to represent them by a one-hot-encoding.

The used neural network is quite simple: it consists of 3 hidden, fully connected layers (with $[200, 100, 20]$ neurons respectively) with rectified linear unit activation (ReLU) introduced as eq. (2) in [13]. As the parking lot occupancy prediction is a regression problem, a mean squared error (MSE) loss function suffices. Finally, we chose the RMSprop optimizer [14] because we empirically found that it results in a slightly faster convergence.

Out of the 60 days of measurement, the first 55 days served as training data, and the rest was the test data. During the training process, 30% of the samples were the validation set. To achieve maximum performance as well as to avoid overfitting, we applied an early stopping mechanism that terminated the training process if there was no significant improvement (0.0001 improvements in MSE loss on the training set within 3 consecutive epochs).

4.2. Performance of the centralized model

In this centralized setup, we can utilize the whole dataset provided by SUMO. Therefore, the expectation might be that this approach performs outstandingly well in the parking lot occupancy prediction.

The results confirm this anticipation: on the last 5 test days, the model produces an MSE loss value smaller than 0.002. As, e.g., Figure 2 shows, the estimation fits the measured value of the parking lot occupancy with minimal error.

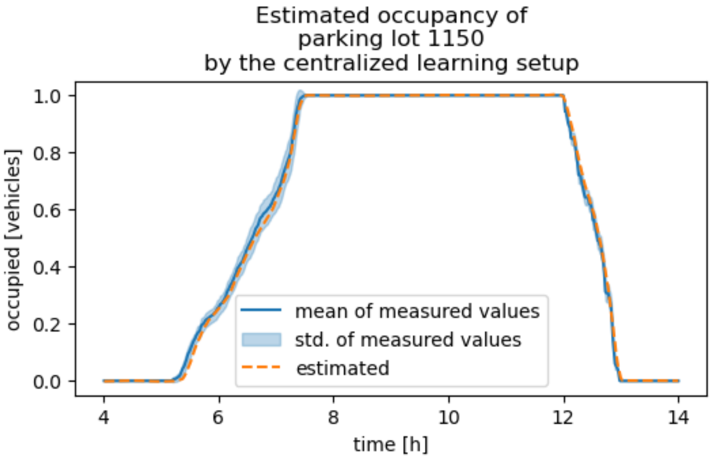


Figure 2. Parking lot occupancy estimation of the centralized approach

5. Individual learning

As we have noted at the beginning of Section 4, a centralized free parking lot prediction system would require many sensors and communication to track the occupancies. Consequently, this results in a high installation and operational cost, especially in curbside parkings.

Fortunately, as modern vehicles will carry more and more sensors, they might be able to measure the occupancy rate of the parking lots on the fly. However, such records can require a vast amount of storage space or have a high communication cost when sending them to a remote server. Therefore, we have to represent this information more compactly. For example, a trained machine-learning model can efficiently encode such data in a predetermined storage space (engineers can determine the number of model parameters at design time). Based on this idea, we tested how an individual vehicle, see Figure 3, can measure and learn the occupancy of the parking lots.

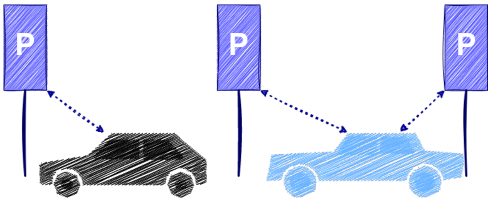


Figure 3. Scheme of the individual learning approach. Individual vehicles try to learn the occupancy of the parking lots.

In the following, for illustration, we will see how vehicle `commercial_3-1_98` from the MoST learns and what performance it achieves.

5.1. Performance of the individual training process

To ensure that only the access to the data influences the training process, we trained an identical neural network to the one proposed in Section 4.1. Hence, the only difference is, in this case, that the neural network utilized only the measurement data of the parking lots which lay along the route of the `commercial_3-1_98` vehicle. The edges from which these parking lots are observable according to Section 3 are shown in Figure 4.

Naturally, we shall not expect that the parking lot occupancies for the whole time range can be accurately estimated. We can only assume that for the measured parking lots at the observation time, the vehicle will be able to approximate the occupancy values. Figure 5 confirms this hypothesis: Figure 5a illustrates how the `commercial_3-1_98` vehicle can predict the occupancy of parking lot no. 1140, which lays along its path. Around the observation time (depicted as a red line), the vehicle can more or less accurately estimate that this specific parking is full. On the other hand, parking lot no. 1101 is not in the knowledge base of the `commercial_3-1_98` vehicle; therefore, it cannot accurately estimate its occupancy, see Figure 5b.

As a vehicle often follows identical routes at the same time of the day, corresponding to the daily routine of its owner, even this model might be helpful to recommend parking lots that are free with high probability at a given time.

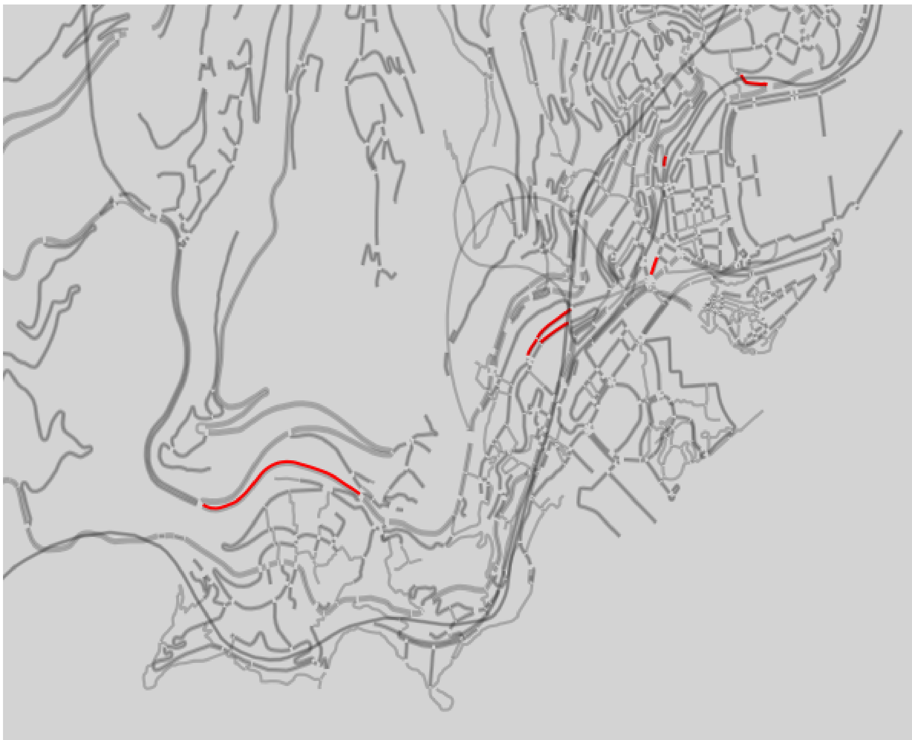


Figure 4. A section of the map being simulated by the MoST. Edges on which vehicle `commercial_3-1_98` measured the occupancy of the parking lots are depicted in red. By connecting these points, we might obtain information about the vehicle's route.

5.2. Vulnerabilities of the individual learning scheme

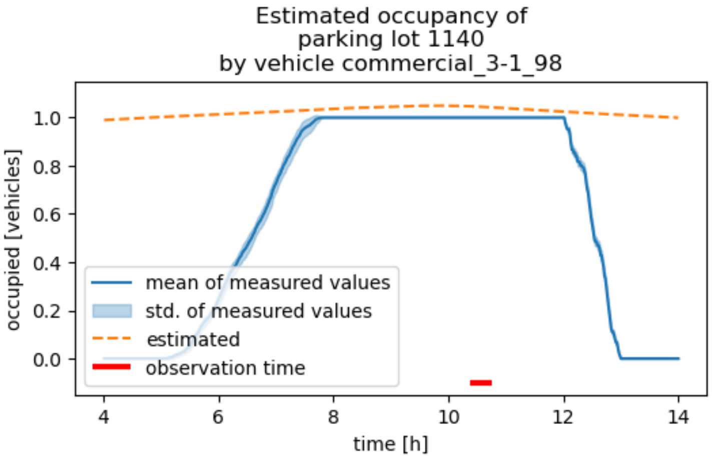
The trained neural network efficiently encodes the measured occupancy data; therefore, it can also be helpful to another vehicle unfamiliar with the environment (at a given time in a district). Therefore, cars might share their trained models.

Supposing that the receiver might be malicious, we shall evaluate the possible vulnerabilities of such a data-sharing technique. To this end, we assume that the receiver is honest-but-curious, trying to infer the route, i.e., the measured parking lots and the observation time of the sender vehicle. The receiver is also an oracle possessing all the occupancy data of the parking lots.

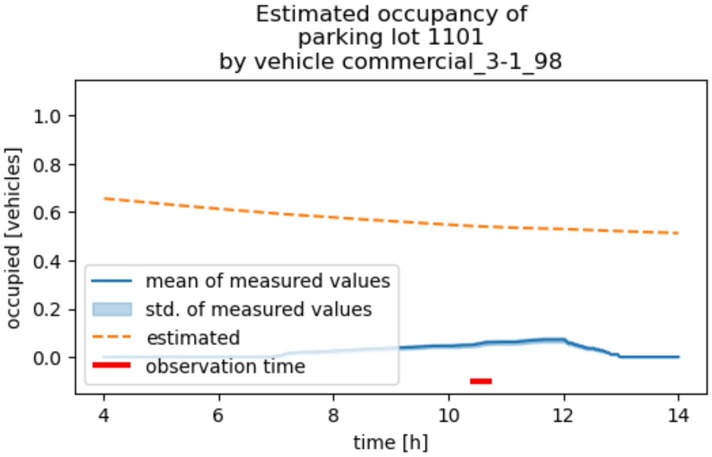
By calculating the prediction accuracy values per parking lot, the attacker can successfully identify some measured parking lots, see Figure 6. We estimated that a vehicle while following its route measures 5.22 parking lots on average in the MoST. Therefore, an attacker can select 5 parking lots having the lowest prediction loss values as the inferred route of the sender. That gives the malicious receiver a map similar to Figure 4, on which it might be able to approximate the path of the sender by connecting the edges with rational and legal routes.

To approximate the observation time, the attacker can apply the following heuristic: first, it selects 5 parking lots, of which predicted occupations are the most accurate. After that, it computes the prediction accuracy $l_p(t)$ of these parking lots per timestep. To smoothen the achieved curve, it can apply a moving average with a window size of e.g. 60 minutes. Let the smoothened curve be $\hat{l}_p(t)$. Then the $\hat{t}_m$ observation time estimate can be defined as:

$$\hat{t}_m = \arg\min_t \hat{l}_p(t). \tag{1}$$



(a) Estimation of the occupancy of a known parking lot



(b) Estimation of the occupancy of an unknown parking lot

Figure 5. Parking lot occupancy estimation of an individual vehicle

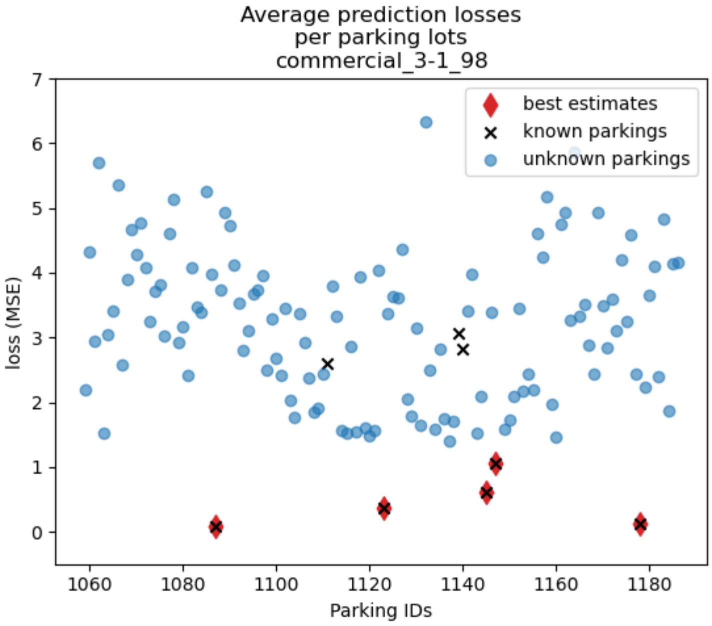


Figure 6. Average prediction accuracies for known and unknown parking lots.

As Figure 7 illustrates, the heuristic of (1) can approximate the observation time of `commercial_3-1_98` with 8 minutes difference. Considering that the observation time offset follows a uniform distribution in the $[0, 15]$ minutes range, this difference is smaller than the offset range.

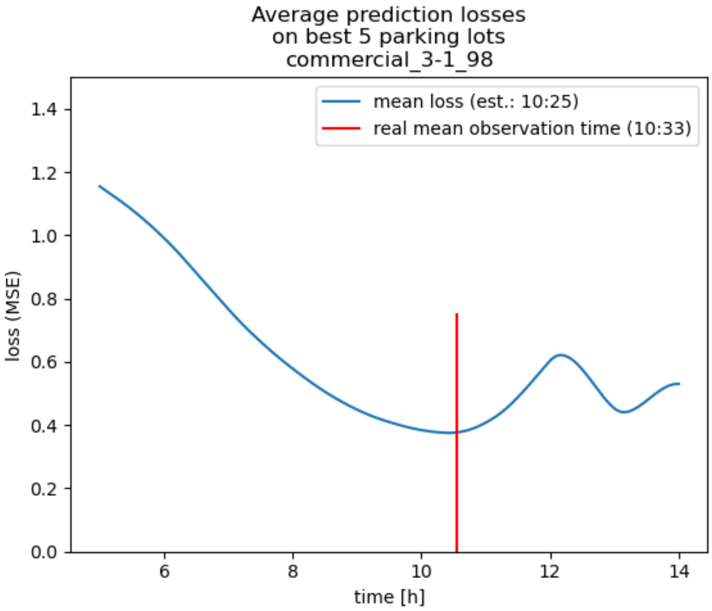


Figure 7. Average prediction accuracies for known parking lots during the simulation time. The minimum of the blue curve is at 10:25 which is the estimate of the observation time according to (1).

6. Federated learning

As both the centralized and individual learning schemes have their drawbacks, it might be fruitful to combine them. It would result in cheap measurements done by the vehicles and an accurate model on the server side. It is the idea of the federated learning scheme, see Figure 8.

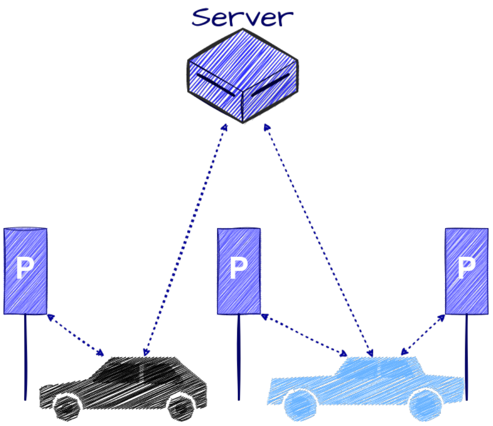


Figure 8. Scheme of the federated learning approach. Individual vehicles train their own models which they send to the server. The server aggregates the models and shares this federated model with the participating vehicles.

Considering that the data provided by the SUMO simulations require approximately 240 GiB of storage space, its processing even on a high-end PC is not feasible. Therefore, we sampled 10% of the vehicles. In this way, we had measurements of approximately 4000 vehicles. This sampling can also represent the situation when only a portion of the cars can measure complex phenomena such as the parking lot occupancy.

After each simulated day, the server randomly chose 50 participant vehicles. These participants received the actual federated model (in the beginning, it was initialized randomly) and trained their local update models, identical to the one described in Section 4.1, with all their data from previous measurements until the time of the training. The participants send the update to the remote server which aggregates them by the `FedAvg` algorithm [9] to obtain the next iteration of the federated model.

Unfortunately, operating this learning scheme requires either plenty of time or the parallelization of the learning tasks. To this end, we developed a software architecture that distributes the computation among multiple PCs. Appendix B highlights the main components and design concepts of the architecture.

6.1. Performance of the federated learning scheme

We expect that the accuracy of the federated learning scheme shall converge to the performance of the baseline, centralized approach. However, there might be some rarely visited parking lots. Therefore, the training data of the federated learning scheme might be sparser than in the centralized approach. That can reduce the numeric performance of the federated system, which achieves an average MSE loss of a little bit above of 1.0 on the test data.

Although this loss value is 3 magnitudes higher than the baseline, a significant part of the imprecision comes from the prediction error of such remote parking places. In real life, we may tolerate this kind of mistake because rarely visited parking lots are usually empty. Hence, it seems more important to predict accurately the occupancy of frequently used parking facilities. Figure 9 illustrates the performance of the federated system on an often-used parking lot. As we can see, the shape of the prediction curve is similar to the real one, but it does not fit as well as that in Figure 2. Due to the limited datasets of the participating vehicles, this process requires either more communication rounds or more participants to achieve the convergent state.

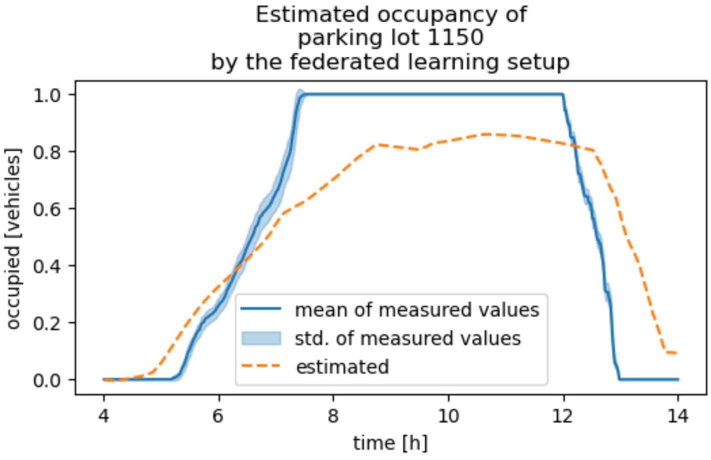


Figure 9. Parking lot occupancy estimation of the federated learning approach

6.2. Vulnerabilities of the federated learning scheme

Considering that the federated learning scheme also incorporates data sharing, we shall investigate the possible security vulnerabilities. Again, we are interested in the routes of the vehicles, i.e., the measured parking lots and the observation times.

The whole federated system slowly converges; therefore, we shall concentrate on the participants' updates. The updates reflect the gradients of the loss function at the participants; consequently, they contain implicit information from the participants' training datasets [9]. Hence, if we compare the performance of the received updates with the sent federated model, we might extract the route of the participant.

To evaluate the success rate of such a comparison-based honest-but-curious attacker, we defined its accuracy as follows. For each participant, the attacker carries out an inference fundamentally similar to the methods described in Section 5.2. The difference is that, in this case, the attacker evaluates the performance of both the federated and the participant models per parking lot. Let us denote the MSE prediction loss of the federated model on the i^{th} parking lot as $l_t^{(f)}(i)$, and the MSE prediction loss of a participant's model on this parking lot as $l_t^{(p)}(i)$. We suppose that the models perform better on the range of the training data ($l_t^{(p)}(i) < l_t^{(f)}(i)$ if the participant vehicle measured parking lot no. i). Therefore, (2) gives a heuristic that the participant is likely measured parking lot no. j:

$$j = \arg\min_i \left(l_t^{(p)}(i) - l_t^{(f)}(i) \right). \tag{2}$$

Let us collect the 5 best parking lots by the above heuristic. Then, we can check how many of these collected parking lots are in the measured parking lot list of the given participants. This ratio will be the accuracy of the attacker: e.g., the accuracy will be 1.0 if all 5 selected parking lots are in the list, up to 0.0 in case these two sets are disjunct. Figure 10 illustrates the curve of this accuracy value. The heuristic performs surprisingly well which proves that in the beginning of the training process, an honest-but-curious attacker can succesfully infer which parking lots were measured by a participating vehicle. Moreover, as the linear trend estimate indicates, the success

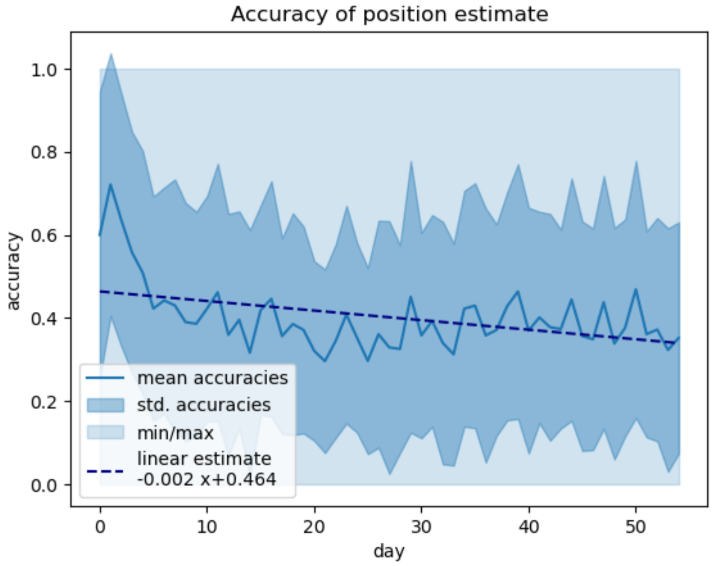


Figure 10. Position inference success rates

rate of the attacker decreases as more and more training rounds are performed. This is due to the convergence of the federated system: a participant in a well-trained scheme would not be distinguished from the other participants. However, we shall note that as a real-world traffic infrastructure constantly changes, we shall not assume that such a convergent state exists.

Moreover, we also tried to estimate the observation times of the participating vehicles. For this approximation, we calculated the $\hat{l}_p^{(f)}(t)$ and $\hat{l}_p^{(p)}(t)$ smoothened MSE loss values on the identified 5 best parking lots per timestep of the federated and the participant models respectively, similarly as in Section 5.2. Then, the observation time estimation $\hat{t}_m^{(p)}$ can be defined as:

$$\hat{t}_m^{(p)} = \arg\min_t \left(\hat{l}_p^{(p)}(t) - \hat{l}_p^{(f)}(t)\right). \tag{3}$$

Figure 11 illustrates the prediction power of the heuristic based on (3). As depicted, the average prediction accuracy oscillates around 2 hours. Consequently, in half of the measurement cases, the approximation error is even smaller than that. As the 1σ standard deviation range shows, the estimate often might be accurate, leaving only a marginal error. It concludes that an honest-but-curious attacker in a federated system may successfully infer both observation time and position.

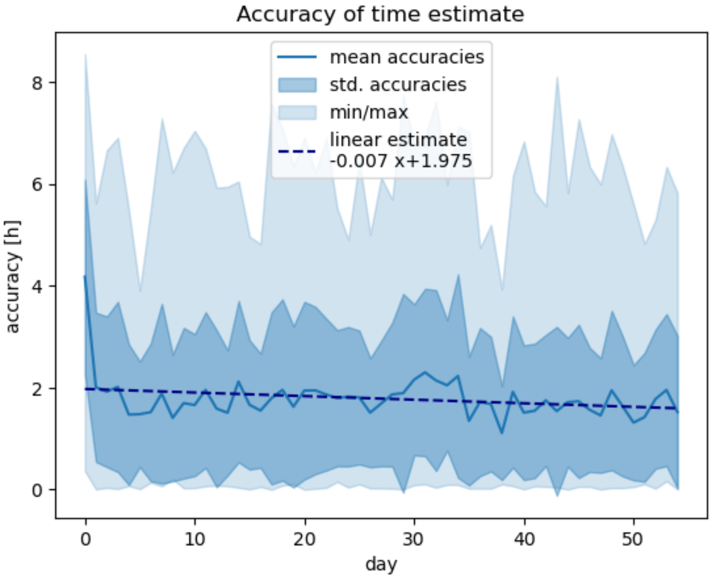


Figure 11. Observation time inference accuracies

As the linear trend indicates it in Figure 11, the observation time estimate gets more accurate as the system trains. That can have various explanations. The first one is that the federated system is yet to converge; therefore, if we perform more training rounds, the probability of a successful observation time inference will decrease. The second possible explanation relies on the execution order: first, a participant trains its local model, then sends back its updates. The malicious server evaluates this update and finally aggregates the received models into the federated one. In this order, a participant can more accurately predict a specific parking lot at a given time than the federated system, explaining why the trendline does not increase. Lastly, it is also possible that we do not have enough data points to achieve a stable, constant value. However, it is possible to proceed with the measurements; operating the federated learning system is computationally really demanding.

7. Conclusion

In this paper, we presented how communicating vehicles can measure and learn a complex phenomenon, i.e., the parking lot occupation of the city. To obtain the input data, we ran Eclipse SUMO multiple times with the MoST scenario. Unfortunately, running the simulations requires plenty of time. To mitigate the time demand, we created a Docker image containing Eclipse SUMO and a TraCI script that can run in parallel in several instances on a PC. As connecting this system with machine learning tools is also challenging, we present a possible software architecture that allows us to distribute the SUMO-based learning system among various computers.

We conclude that a *centralized learning scheme can outperform a federated one*, but its real-world implementation is not feasible due to the need for thousands of sensors in a city. On the other hand, vehicles might learn the parking lot occupancies in a specific district and time of the day; such individually trained models cannot perform well outside the range of their training datasets. Assuming that this neural network model is the compressed version of the collected data, it might be worth sharing it with other vehicles. But it should be emphasized that this data sharing poses a potential privacy risk as an honest-but-curious partner can infer the vehicle's route and observation time.

We also tested a federated learning scheme to combine the benefits of cheap data collection and acceptable model performance. However, such *a federated system is neither entirely secure, especially not at the beginning of the training process*. It is a challenging task to make this federated measurement and learning system protected and well-performing. To this end, our future research focuses on solving that problem.

A. Creating a Docker image with SUMO and TraCI

As running even one instance of the MoST scenario in Eclipse SUMO takes approximately one hour, running it in multiple instances in parallel is quite beneficial. Unfortunately, we ran into a problem[2] when we tried to execute our TraCI script either multithreaded or in separate processes.

As a workaround, we created a Docker image containing SUMO and our TraCI script to collect measurement data. In our GitHub repository, one can find a Dockerfile that creates an Ubuntu-based SUMO installation. Besides pulling the latest SUMO version, the resulting image will incorporate the measurement script too.

B. A multi-computer client-server architecture for SUMO-based learning systems

The operation of the described federated system requires training multiple neural network agents. Unfortunately, it also consumes a significant amount of time even on a high-end PC (with AMD Ryzen 7 3800X CPU, 32 GiB of RAM, and an Nvidia RTX3060 GPU). Moreover, we used Tensorflow to implement the neural network, which, for some unknown reason, does not support multithreaded or process-based parallel training. Therefore, we designed a framework to distribute the workload among various computers, see Figure 12. This approach is fundamentally similar to the parallel training described in [15]. However, a reinforcement learning process heavily depends on the

[2] As of Eclipse SUMO version 1.15.0, the problem may be somehow related to multiprocessing, as the circumstances of getting a TraCI error were not deterministic.

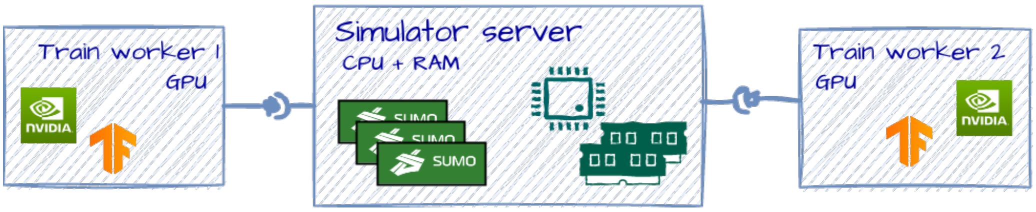


Figure 12. Schematics of a distributed learning system based on SUMO simulations

simulation speed; in our case, the performance bottleneck is the training time of a neural network.

One can deploy multiple SUMO containers, described in Appendix A, on a computer. As SUMO efficiently uses the system memory and runs most of the time on a single CPU core, the `simulator server` might not necessarily be a high-performance computer.

Moreover, training neural networks can be much faster on computers with GPU. To take advantage of it, we can create several `train workers`. These train workers shall operate a simple HTTP server, e.g., implemented by Flask, and provide services such as training a neural network. Or use the neural network to predict a value. These services can be accessed by calling the corresponding HTTP requests. As the interface is through HTTP protocol, we can deploy `train workers` to multiple computers. For more complex tasks and setups, one might also place the `train workers` into Docker containers and build up a Kubernetes-based system.

Data availability statement

As input data, we have used the Monaco SUMO Traffic Scenario [2] in our simulations which one can access at `https://github.com/lcodeca/MoSTScenario`.

Underlying and related material

The source codes producing the presented results are available at: `https://github.com/alelevente/sumo_sec`.

Author contributions

Levente Alekszejenkó is a Ph.D. Student under the supervision of Tadeusz Dobrowiecki. As a part of Mr. Alekszejenkó's Ph.D. research, he conducted the investigation and created the visualization presented in this paper. Both authors are responsible for the presented concepts and methodologies. Prof. Dobrowiecki also had many constructive commentaries on the manuscript written by Mr. Alekszejenkó in the pre-publication stage.

Competing interests

The authors declare that they have no competing interests.

Funding

This research was funded by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02.

This study was also supported by OTKA 139330 and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

References

[1] W. Y. B. Lim, N. C. Luong, D. T. Hoang, *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020. DOI: 10.1109/COMST.2020.2986024.

[2] L. Codeca and J. Härri, "Monaco SUMO Traffic (MoST) Scenario: A 3D Mobility Scenario for Cooperative ITS," in *SUMO 2018, SUMO User Conference, Simulating Autonomous and Intermodal Transport Systems, May 14-16, 2018, Berlin, Germany*, B, May 2018.

[3] 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: https://elib.dlr.de/124092/.

[4] F. Bock, S. Di Martino, and A. Origlia, "Smart parking: Using a crowd of taxis to sense on-street parking space availability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 2, pp. 496–508, 2020. DOI: 10.1109/TITS.2019.2899149.

[5] H. Khayyam, B. Javadi, M. Jalili, and R. N. Jazar, "Artificial intelligence and internet of things for autonomous vehicles," R. N. Jazar and L. Dai, Eds., pp. 39–68, 2020. DOI: 10.1007/978-3-030-18963-1_2.

[6] S. Iqbal, P. Ball, M. H. Kamarudin, and A. Bradley, "Simulating malicious attacks on VANETs for connected and autonomous vehicle cybersecurity: A machine learning dataset," in *2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, July 20-22. 2022.*, 2022, pp. 332–337. DOI: 10.1109/CSNDSP54353.2022.9908023.

[7] P. Sharma, D. Austin, and H. Liu, "Attacks on machine learning: Adversarial examples in connected and autonomous vehicles," in *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, 2019, pp. 1–7. DOI: 10.1109/HST47167.2019.9032989.

[8] M. Türkoğlu, H. Polat, C. Koçak, and O. Polat, "Recognition of DDoS attacks on SD-VANET based on combination of hyperparameter optimization and feature selection," *Expert Systems with Applications*, vol. 203, p. 117 500, 2022, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.117500.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, A. Singh and J. Zhu, Eds., ser. Proceedings of Machine Learning Research, vol. 54, PMLR, Apr. 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html.

[10] J. Posner, L. Tseng, M. Aloqaily, and Y. Jararweh, "Federated learning in vehicular networks: Opportunities and solutions," *IEEE Network*, vol. 35, no. 2, pp. 152–159, 2021. DOI: 10.1109/MNET.011.2000430.

[11] A. Blanco-Justicia, J. Domingo-Ferrer, S. Martínez, D. Sánchez, A. Flanagan, and K. E. Tan, "Achieving security and privacy in federated learning systems: Survey, research challenges and future directions," *Engineering Applications of Artificial Intelligence*, vol. 106, p. 104 468, 2021, ISSN: 0952-1976. DOI: 10.1016/j.engappai.2021.104468.

[12] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, “Location privacy-preserving mechanisms in location-based services: A comprehensive survey,” *ACM Comput. Surv.*, vol. 54, no. 1, Jan. 2021, ISSN: 0360-0300. DOI: 10.1145/3423165.

[13] K. Fukushima, “Cognitron: A self-organizing multilayered neural network,” *Biological Cybernetics*, vol. 20, pp. 121–136, 1975. DOI: 10.1007/BF00342633.

[14] T. Tieleman and G. Hinton, “Lecture 6e – rmsprop: Divide the gradient by a running average of its recent magnitude,” *Neural Networks for Machine Learning*, 2012. [Online]. Available: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[15] N. Kheterpal, K. Parvate, C. Wu, A. Kreidieh, E. Vinitsky, and A. Bayen, “Flow: Deep reinforcement learning for control in SUMO,” in *SUMO 2018 – Simulating Autonomous and Intermodal Transport Systems*, E. Wießner, L. Lücken, R. Hilbrich, *et al.*, Eds., ser. EPiC Series in Engineering, vol. 2, EasyChair, 2018, pp. 134–151. DOI: 10.29007/dkzb.

SUMO User Conference 2023
Conference paper
https://doi.org/10.52825/scp.v4i.222

Published: 29 June 2023

Challenges in Reward Design for Reinforcement Learning-based Traffic Signal Control: An Investigation using a CO_2 Emission Objective

Max Eric Henry Schumacher[1] [https://orcid.org/0000-0003-0486-3498],
Christian Medeiros Adriano[1] [1] [https://orcid.org/0000-0003-2588-9937], and
Holger Giese[1] [https://orcid.org/0000-0002-4723-730X]

[1] Hasso-Plattner Institute, University of Potsdam, Germany

Abstract: Deep Reinforcement Learning (DRL) is a promising data-driven approach for traffic signal control, especially because DRL can learn to adapt to varying traffic demands. For that, DRL agents maximize a scalar reward by interacting with an environment. However, one needs to formulate a suitable reward, aligning agent behavior and user objectives, which is an open research problem. We investigate this problem in the context of traffic signal control with the objective of minimizing CO_2 emissions at intersections. Because CO_2 emissions can be affected by multiple factors outside the agent's control, it is unclear if an emission-based metric works well as a reward, or if a proxy reward is needed. To obtain a suitable reward, we evaluate various rewards and combinations of rewards. For each reward, we train a Deep Q-Network (DQN) on homogeneous and heterogeneous traffic scenarios. We use the SUMO (Simulation of Urban MObility) simulator and its default emission model to monitor the agent's performance on the specified rewards and CO_2 emission. Our experiments show that a CO_2 emission-based reward is inefficient for training a DQN, the agent's performance is sensitive to variations in the parameters of combined rewards, and some reward formulations do not work equally well in different scenarios. Based on these results, we identify desirable reward properties that have implications to reward design for reinforcement learning-based traffic signal control.

Keywords: Traffic Signal Control, Reinforcement Learning, Reward Modeling, Pollutant Emissions

1 Introduction

Deep reinforcement learning (DRL) is a data-driven approach that holds promise for improving traffic signal control (TSC), because DRL can learn to adapt to changing traffic demands [1]–[3]. To achieve this, a DRL agent interacts with its environment and learns to take actions that maximize a cumulative scalar reward. By doing so, the agent can optimize the flow of traffic and improve the overall efficiency of the system.

In TSC, the actions correspond to changes in traffic lights and rewards correspond to traffic flow metrics (e.g., average vehicle speed, braking accelerations, and queuing lengths at intersections). However, in real-world applications of DRL, the agent's reward should also reflect the users' goals, which in TSC could be, to minimize traffic delays [4], [5] and CO_2 emissions [6], [7]. Nonetheless, it is not obvious how to select reward formulations that are also effective in satisfying users' goals. This is an open and challenging research problem known as the ”agent alignment problem”. The optimization goal of minimizing travel time in the context of TSC is challenging due to the influence of various external factors, such as free flow speed and current congestion level, which are beyond the agent's immediate control [8]. While this makes travel time an ineffective reward in practice [4], it is also not obvious which traffic flow metrics are guaranteed to be effective to this goal. There are several studies that combine traffic metrics as rewards for DRL agents [4], [5], [9]. Similarly, there is a growing body of research on training DRL agents to minimize pollutant emissions in TSC [6], [7]. However, these DLR-based approaches provide limited insight into how the convergence curves of traffic metrics behave relative to CO_2 emissions during the training of DRL agents. This information is important for designing reward functions that are effectively aligned with the users' goals. We investigate how to bridge this knowledge gap by performing a systematic study of the reward design space, which comprises single-metric rewards, combined-metrics and their corresponding parameterizations (weights in a linear function). For each candidate reward, we train a Deep Q-Network (DQN) [10] on two traffic scenarios, one with homogeneous traffic and one with heterogeneous traffic. To evaluate the various reward model formulations, we adopt the SUMO (Simulation of Urban MObility) simulator and its default emission model (Handbook Emission Factors For Road Transportation - HBEFA 3.1) [11]. Our evaluation consist of measurements of convergence curves of the agent's reward and the corresponding CO_2 emissions, producing the following results:

1. a CO_2 emission-based reward is inefficient for training a DQN agent,
2. only a few single-metric rewards were capable of minimizing CO_2 emissions,
3. metrics that individually did not produce effective reward formulations, were, when combined, successful in minimizing CO_2 emissions,
4. and, even when there exists an effective instance of a combined reward (e.g., a combination of queue and brake), there are still variations (i.e., from different parameterizations) of those same traffic flow metrics that produce ineffective rewards.

These results generalize both under homogeneous and heterogeneous traffic flow scenarios. Based on these results, we generated two contributions in the form of systematic analyses.

1. *Property-based analysis* of convergence curves. This analysis generates explanations for the cases of insufficient alignment between the agent's reward model and the CO_2 emission goal. The explanations consist of a paradigmatic classification of the reward models through orthogonal categories defined by two properties. *Informativeness* captures how well the agent approximates the given proxy reward, and *expressiveness* reflects how strong episode rewards correlate with episode CO_2 emission levels.
2. *Sensitivity analysis* of the challenges to align combined reward models with CO_2 emission goals. This analysis shows that alignment has two levels of sensitivity: the choice of traffic flow metrics, and the parameterization of these metrics in a linear reward formulation.

The remainder of the paper is organized as follows. In Section 2, we present the problem of agent alignment and its impact on TSC and emissions. We contextualize our work in relation to DRL for TSC, and for minimization of pollutant emissions (Section 3).

The approach and experimental setup is detailed in Section 4, while the corresponding results are presented in Section 5. The analyses of these results in terms of contributions, implications, and threats to validity are discussed in Section 6. Finally, we offer our conclusions and ideas for future work in Section 7.

2 Foundations

Deep reinforcement learning (DRL) is a popular approach that combines deep neural networks with reinforcement learning to enable agents to learn optimal behavior in complex environments. However, ensuring that the goals of the system align with the goals of the user is a critical challenge in DRL systems. Misaligned goals can result in unintended and potentially harmful outcomes that undermine the users' goals. This section examines the challenges that make alignment difficult in DRL systems and describes how reward models align with user goals.

2.1 The Agent Alignment Problem

The AI alignment problem [12] consists of finding ways to ensure that, quoting [13]: ”... these [machine learning] models capture our norms and values, understand what we mean or intend, and, above all, do what we want”. In other words, it involves matching agent rewards and users' goals regarding behavior [14], intent [15], incentive [16], inner and outer alignment [17], and instruction alignment [18]. Behavior alignment consists of producing predictions for given inputs, whereas intent looks at more general specification that cover different desired behaviors. Incentive alignment studies how rewards induce desired behaviors, whereas inner and outer alignment deals with partitioning the alignment in scopes that present specific dynamics. Instruction alignment consists of communicating human intent as a sequence of instructions that must be learned. These various definitions of alignment make specification, measurement, and evaluation challenging.

Therefore, a more pragmatic approach is to look at the failure of the agent to align with the user's goals (misalignment). Misalignment can have unintended consequences that are counterproductive (optimize against the users' goals), futile (no effect on users' goals), or simply could jeopardize users' goals (suboptimal behavior). Additionally, misalignment in DRL can increase the chances of reward hacking [19], [20]. For instance, in the case of a game boat race, an agent maximized a reward by indefinitely hitting a nearby target without ever concluding the race [21] – violating what the user intended.

One can argue for a proper definition for the user's goal and how it should be reflected on the reward model; however, this is still challenging, as evident in the many recent AI failure cases reported in the ”Artificial Intelligence Incident Database”[1]. In other words, there is no perfect alignment [15]. Instead, one needs to specify models that satisfy the conditions of being sufficiently *meaningful* and *precise* to steer the process of achieving user goals (e.g., reducing CO_2 emissions) by optimizing traffic flow metrics. For that, one needs a systematic way to evaluate how reward models align with user goals. Our approach presented in this paper is to partition the alignment specification problem into two metrics that allow to *express* a *meaningful* goal, and *inform* precisely enough how this goal can be achieved.

[1] https://incidentdatabase.ai/

2.2 Alignment Challenges

Partial observability in the form of hidden states (inherent to DRL environments) make alignment more difficult to achieve by preventing the agent from observing all the effects of its actions (in particular the delayed ones).The hidden states can result both from misspecified (wrong) [22] and underspecified (incomplete) [23] models. In the context of DRL, wrong or incomplete models can cause the agent to show good convergence curves at training time, but present unexpected behaviors after deployment. This can have consequences for the safety and cost of applications like autonomous vehicles and robotics.

Delayed and stochastic effects of actions are challenges when performing credit assignment, i.e., determining how each action contributed to achieve the users' goal. While delay and stochasticity cannot be eliminated, as they are properties of the environment, one can have reward models that are less sensitive to these factors. In the case of emissions, one can compare how different traffic flow metrics (e.g., average speed versus queue length) relate to changes in CO_2 emissions.

2.3 Deep Q-Network

Q-Learning is a popular reinforcement learning algorithm that helps agents make decisions based on rewards in their environment. It involves estimating the action-value function, which maps a state and action to the expected future rewards. In tabular Q-learning, the action-value function is represented as a table, but this becomes impractical for large or continuous state and action spaces [24]. Function approximation can solve this problem by representing the function using a neural network or another approximator.

Neural Fitted Q-Iteration (NFQ) [25] is an extension of tabular Q-learning with function approximation, improving scalability to large state-action spaces. However, NFQ uses a fixed dataset; thus, it is susceptible to overfitting on the training data. To mitigate this problem, Deep Q-Network (DQN) was introduced [10]. DQN builds on NFQ and introduces two key components: the experience replay buffer and the target network. The replay buffer stores the agent's experiences that can be retrieved for updating the Q-value estimates. The target network is used to set the TD targets, which are calculated based on the immediate reward and discounted future returns. Finally, our choice for DQN relied on its simplicity (off-policy and model-free), as it would allow to establish a comparison baseline for more sophisticated approaches like Proximal Policy Optimization algorithms (PPO) [26].

3 The State of the Art

This section introduces the topic of reward modeling in deep reinforcement learning (DRL) and its application to traffic signal control (TSC).

3.1 Reward Modeling

Reward modeling consists of learning to achieve specific user goals without requiring human feedback [14]. It has become a popular approach that precludes manually solving the credit assignment problem (e.g., via reward shaping [27]). However, because designed rewards can still be tampered by a learning agent [19], one still has to evaluate how alignment is done via reward modeling. This gives rise to the **Optimal**

Reward Problem - ORP [28], which aims to reduce the alignment problem to a reward modeling problem. This might involve defining intrinsic or extrinsic rewards [29]. The intrinsic reward constraints the agent on *how it can learn*, whereas the extrinsic reward instruments the user's goal by steering the agent on *what it can learn*[2]. We translate these intuitions respectively into two convergence properties named *informativeness* and *expressiveness* (formalized in Section 6.1).

3.2 Reward Models for Traffic Signal Control

Minimizing travel time is the main goal of a TSC policy. However, because travel time is affected by a multitude of factors and actions with delayed effects [8], traffic engineers rely on proxy reward metrics, like average waiting time, average intersection speed, or total braking acceleration. Accordingly, in DRL, various combinations for a reward models were investigated: queue length and delay in [5], queue length and pressure in [4], stop time and average speed and time lost [9], and many others (see Table-5 in [8]). We extend this family of work by combining more metrics (vehicle speed, brake acceleration) and evaluating their impact on CO_2 emissions.

3.3 Pollutant Emissions in Traffic

Traditionally, the first solutions comprised non-DRL control (both with SUMO [31], [32] and other simulators [33]–[35]). More recently, DRL-based TSC approaches to minimize pollutant emissions have been investigated [6], [7]. However, these DLR-based approaches provide limited understanding about the relationship between metrics for emissions and traffic flow, in particular, regarding how the convergence curves of metrics behave during the training of DRL agents. Without a proper understanding of this relationship, one is hindered in the task of reward modeling for aligning the agent's reward with CO_2 emission goals in TSC. Therefore, to bridge this gap, we investigated various formulations for a linear reward function based on traffic flow metrics, and computed the corresponding CO_2 emissions using SUMO's provided emission model from the Handbook Emission Factors For Road Transportation (HBEFA 3.1) [11].

3.4 Deep Reinforcement Learning for Traffic

The specification of the DRL approach goes beyond the choice of reward function: one needs to choose an algorithm and how to model the state-space. Among the many DRL algorithms to have been adopted [8], the DQN [10] algorithm has been one of the most popular choices (Table-1 in [3]). The adoption of DQN for TSC stems from its relative simplicity of having discrete actions, while still providing good convergence behavior [36].

Concerning the **state-space**, the traffic environment has been modeled at various levels of resolution, from coarse (flow) to fine (vehicle speed and position) [8], resulting in tabular discretized metrics [37], and image representations [36]. We opted for a lane segment level resolution and discretized metrics because studies could not show better results when adopting higher resolution [38] or more complex state representations [5].

[2] This could involve curiosity-driven exploration [30], which attributes credit based on the novelty of the state-action pair, usually measured by some information theoretic metric, e.g., entropy, mutual-information, or KL-divergence.

4 Methodology

In this section, we outline the methods used to study CO_2 emissions produced at signalized intersections. Our approach builds on the principles of reinforcement learning, where an agent learns to make decisions based on the interaction with its environment. We outline the traffic simulation scenario in Section 4.1 and formulate the reinforcement learning task in Section 4.2, defining the states, actions, and rewards used by the agent. Finally, in Section 4.3, we provide a detailed description of the experimental setup, including the neural network architecture, hyperparameters, and the setup of the traffic environment, used to train and evaluate the DQN algorithm.

4.1 Traffic Scenarios

We propose a scenario that comprises a controlled intersection (shown in Fig. 1), featuring two incoming and two outgoing lanes. The intersection allows two types of phases: either green or yellow in the north-south direction (NSG, NSY); or green or yellow in the east-west direction (WEG, WEY). In both cases, the orthogonal direction is set to red. Fig. 1 illustrates the intersection in NSG phase.

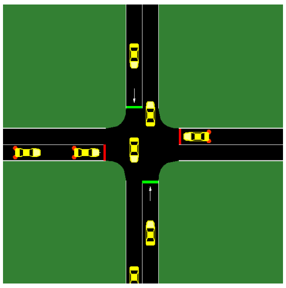


Figure 1. Screenshot (SUMO GUI) of a signalized intersection with four lanes.

We combine this infrastructure with traffic flows as shown in Fig. 2, consisting of two types: a time-varying Bernoulli distribution, and a traffic flow that remains constant throughout the simulation. At each second and on each road (north-south, west-east, etc.), a car is released into the simulation with a probability of p. Each traffic demand combined with the signalized intersection infrastructure gives rise to one scenario: a *heterogeneous traffic scenario*, using the time-varying demand, and a *homogeneous traffic scenario* (using the fixed demand).

For the heterogeneous traffic scenario, depicted in blue in Fig. 2, we deliberately chose a peak traffic volume of $p = 0.25$ – the maximum probability of releasing a car. This level of peak traffic makes the scenario challenging, as it exceeds the maximum intersection throughput and causes congestion temporarily. In contrast, the homogeneous traffic flow, depicted in red in Fig. 2, has a fixed probability of releasing a car with a value of $p = 0.2$. This value represents the maximum intersection throughput, ensuring that the flow remains steady throughout the simulation.

4.2 The Reinforcement Learning Task

Traffic signals play a critical role in ensuring safe and efficient traffic flow at intersections. Fixed pre-timed controllers are often insufficient in optimizing traffic flow, as traffic volume and driving behavior vary widely. Adaptive traffic signal control (ATSC) provides a solution, which uses electrical sensors and sets signals based on the data, adapting

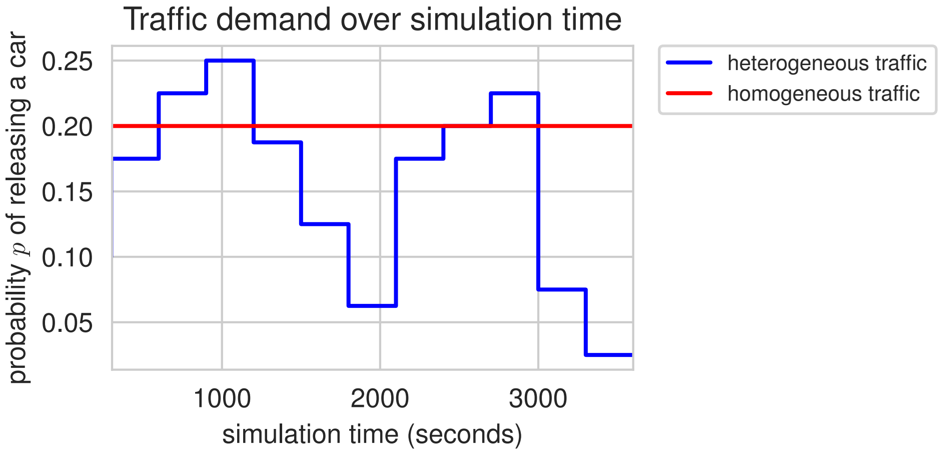
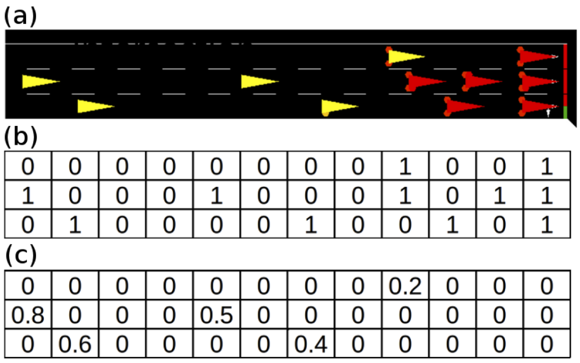


Figure 2. The probabilities of releasing cars into the system over simulation time.

to the current traffic situation. One of the simplest methods to achieve ATSC is actuated signal control, which triggers a specific signal based on sensory data gathered around the intersection. Reinforcement learning (RL) is a possible solution to obtain a program for ATSC. The output of the RL algorithm, the agent's policy, becomes the desired ATSC program, which works fully automated, and can be scaled. ATSC with DRL has achieved outstanding results, outperforming conventional methods in many situations. The agent repeatedly collects state information, acts, and updates its policy with a scalar reward, while being trained in safe or simulated environments. For the remainder of this Section, we will assume the environment described in Section 4.1 and specify the components of the reinforcement learning problem, the states, actions and rewards of the agent.

The agent's state or observation is a representation of the environment that the agent perceives at any given time, including relevant information that the agent can use to take actions that maximize its rewards. In the case of traffic signal control with reinforcement learning, the DTSE (Discrete Traffic State Encodings) state [39] is a commonly used representation that consists of two 2D matrices. The first matrix is a binary position matrix that encodes the presence or absence of a vehicle at each intersection, as depicted in Fig. 3 (b). The second matrix is a normalized velocity matrix that tracks the average speed of the vehicles on a given segment, as depicted in Fig. 3 (c).



(b)

0	0	0	0	0	0	0	0	1	0	0	1
1	0	0	0	1	0	0	0	1	0	1	1
0	1	0	0	0	0	1	0	0	1	0	1

(c)

0	0	0	0	0	0	0	0	0.2	0	0	0
0.8	0	0	0	0.5	0	0	0	0	0	0	0
0	0.6	0	0	0	0	0.4	0	0	0	0	0

Figure 3. Example of simulated traffic (a) with corresponding Boolean- (b) and Real-valued DTSE vectors (c). Image source: [39]

Our approach uses DTSE representations, which capture the position and speed of vehicles – key factors in determining CO_2 emissions. This allows the agent to make informed decisions about when to change traffic lights to achieve the goal of our RL task, which is to minimize CO_2 emissions.

The agent's actions are determined by the traffic scenario, as described previously. That is, the agent takes action every Δt (in seconds) and chooses from the set of allowed phases $\mathcal{A} = \{NSG, WEG\}$. Additionally, on each phase change, a yellow transition phase (NSY or WEY) is induced to ensure safety. In contrast to our approach, the agent could also cycle through a pre-defined sequence or operate in non-fixed intervals. However, using a fixed action interval with a set of allowed phases provides a balance between flexibility and difficulty, as non-fixed intervals make the problem harder, and a pre-defined sequence limits the agent's options.

The agent's reward is composed of one or multiple of the following average traffic metrics, aggregated over all lanes: queuing length (queue reward), vehicle speed (speed reward), braking acceleration (brake reward), and CO_2 emission rates (emission reward). Additionally, we provide linear combinations of average queuing length and braking acceleration (queue+brake reward) as well as queuing length and speed metrics (queue+speed reward).

4.3 Experimental Setup

Each experiment uses one of the intersection scenarios described in Section 4.1, with either heterogeneous or homogeneous traffic. Each training run uses simulations that last for 3600 seconds (simulation time), and the agent interacts in intervals of $\Delta t = 5s$, resulting in 720 steps $t = 1, \ldots, 720$ per episode. At episode termination, the simulation is reset, and the agent continues training. For a phase-switch, we selected a yellow time to of $t_{yellow} = 2s$. The agents observe DTSE features with speed and position information. To compute DTSE features, we split each road into 30 segments (segments of length $c \approx 8.33m$). Table 1 summarizes this general setup.

The DQN agent uses a Multi-Layer Perceptron (MLP) with two hidden layers as the neural network, each containing 64 neurons, and a linear output layer with four neurons (one for each action). We use the Adam optimizer [40] for mini-batch gradient descent, with a batch size of 64 and an initial learning rate of $\alpha = 1e-4$. To explore the environment, the agent begins with 100% exploration (ϵ = 1) and gradually decreases exploration linearly to 10% over the first third of training. The replay buffer holds up to 2000 samples, and learning begins after the first episode (720 steps of initial experience). The target network is updated every $C = 10000$ (steps), and the agent's discount factor is $\gamma = 0.99$, which captures long-term rewards. Hyperparameters and training setup are summarized in the second section of Table 1.

5 Results

This section is organized into three parts. In Section 5.1, we evaluate the suitability of CO_2 emission rates as a reward. In Section 5.2, we compare the performance of agents trained on proxy rewards to those trained on a CO_2 reward. Finally, in Section 5.3, we examine how different combinations of reward parameters impact agent's alignment.

Table 1. Environment, hyperparameter and training setup.

Parameter	Value	Description
episode length	$3600s$	episode length in seconds
Δt	$5s$	interval in which the agent interacts in seconds
T	720 steps	number of agent-environment interactions in an episode
t_{yellow}	$2s$	yellow transition time for phase switches
$\mathcal{A}$	$\{NSG, WEG\}$	action space of the agent
$\mathcal{S}$	DTSE	the agent's observable state space
c	$8.33m$	length of a DTSE segment
optimizer	Adam	optimizer
α	$1e^{-4}$	learning rate
batch-size	64	mini-batch size
buffer-size	2000	size of the replay buffer
learning starts	720 steps	number of steps of initial exploration without learning
C	10000 steps	update interval for the target-network of DQN
γ	0.99	discount factor

5.1 CO_2 Emissions as Reward

In Fig. 4 we show the performance of two DQN agents: one agent was trained on a speed reward, and the other agent was trained on the CO_2 emission reward. The solid line represents the median episode emission rate in g/h, and the shaded area shows the 95% confidence intervals. Our results demonstrate that while the agent trained on the CO_2 emission reward does improve in the first episode of training, it converges to a higher emission rate than the agent trained on the speed reward, and does not show any further improvement over time.

These findings suggest that training with the CO_2 reward leads to suboptimal behavior, as the agent is constrained in maximizing this reward and fails to learn an effective policy for minimizing CO_2 emissions. In contrast, the agent trained on the speed reward is able to converge to a better policy for emission minimization, ultimately achieving a lower emission rate.

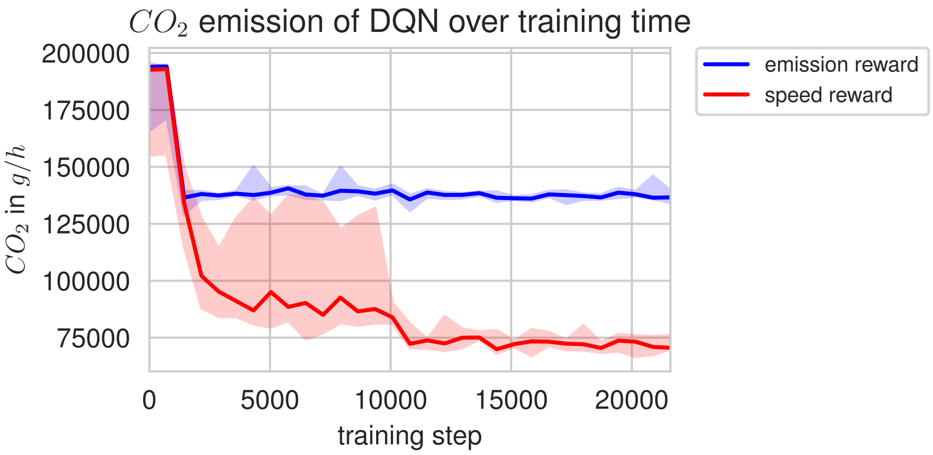


Figure 4. Two agents' performance on minimizing CO_2 emissions by following distinct formulations of cumulative reward. The blue agent has an emission-based reward formulation, whereas the red agent has a speed-based formulation.

5.2 Proxy Rewards for CO_2 Minimization

To explore alternative approaches to incentivizing emission reduction, we investigate the use of various proxy rewards in this section. Specifically, we analyze the performance of DQN agents trained on rewards based on queue lengths, braking accelerations, average speed, CO_2 emissions, a combination of queue length and braking acceleration, and a combination of queue length and average speed.

We present the results of our experiments in Fig. 5. This figure summarizes the performance of each agent on the different reward models, with each subplot showing the CO_2 emission rates (red line), proxy reward (green line), and maximum observed proxy rewards (dotted yellow line). In addition, the shaded areas in each subplot represent 95% confidence intervals for the emission rates and rewards.

Based on the results presented in Fig. 5, we observe that the DQN agent trained on the CO_2 emission reward converged to a suboptimal policy after one episode, resulting in comparatively high emission levels. Similar behavior was observed for the DQN agent trained on the queue reward, which achieved a reduction in CO_2 emissions, but at suboptimal levels. The agent trained on the brake reward had a positive correlation between CO_2 emissions and the episode reward, leading to no reduction in CO_2 emissions.

Good emission performance was achieved by agents using a speed reward and a combined queue and brake reward, denoted as queue-brake reward. The DQN agent trained on the speed reward achieved a relatively low CO_2 emission rate, while also achieving the highest speed reward among all agents. The DQN agent trained on the queue-brake reward achieved the lowest CO_2 emission levels so far, showing a negative correlation with CO_2 emissions (see Table 2).

Overall, the queue-brake reward was the most effective in reducing CO_2 emissions, while the speed reward was effective in achieving a relatively low CO_2 emission rate and high speed reward. Conversely, the emission and queue rewards resulted in suboptimal emission levels.

We calculated the degree of association between the episode CO_2 emissions and episode rewards as a measure of the behavior of the agent alignment (see Table 2). For that, we adopted the Kendall-*tau* rank correlation coefficient[3].

Table 2. Kendall-*tau* correlations (τ) between episode rewards and episode emissions. All values were statistically significant (*p-value* $\leq$ 0.05).

reward	τ	*p-value*
emission	-1.000	2.7e-91
speed	-0.832	8.0e-64
queue	-0.361	2.8e-13
brake	0.505	1.5e-24
queue-brake	-0.893	2.9e-73

5.3 Sensitivity to Reward Parameters

In this experiment, we explored the impact of reward parameter combinations on the performance of a DQN agent in managing traffic flow with the aim of minimizing CO_2

[3] The Kendall-*tau* coefficient is a non-parametric statistic that quantifies the strength and direction of association between two variables without assuming any specific distribution [41].

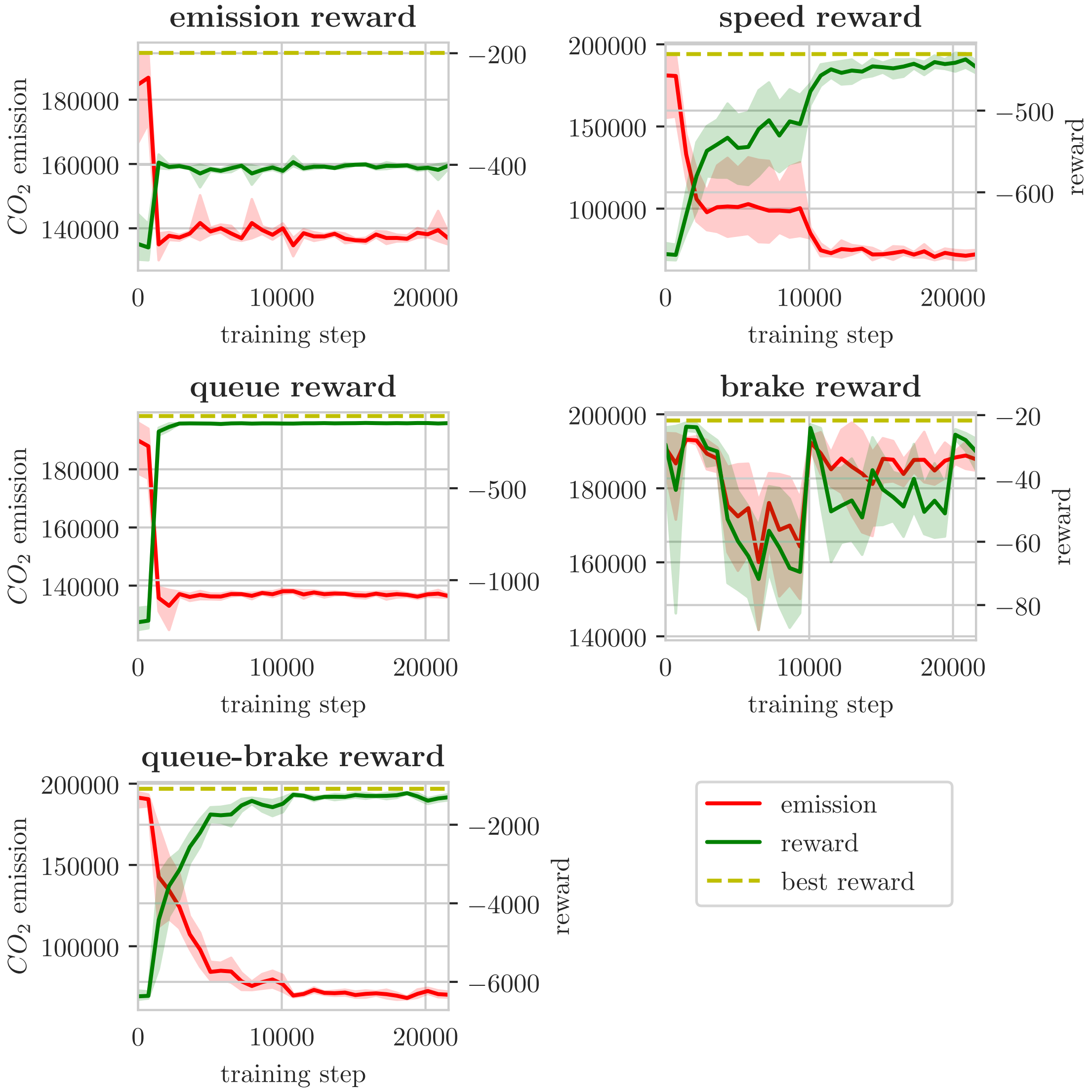


Figure 5. CO_2 emission rates in g/h (red) and absolute episode emission reward (green) of DQN agents over training time. The solid lines depict the median values, while the shades depict 95% confidence intervals. Each reward is combined to a ”best reward” (yellow) that corresponds to the highest value on this reward that was observed among all agents (trained with various rewards).

emissions. We varied the ratio of queue and brake in a combined reward, and queue and speed in a combined reward. For each combination of metrics, we trained six DQN agents for 21800 steps and evaluated their performance in both heterogeneous and homogeneous traffic scenarios.

In Fig. 6 we show the average episode CO_2 emissions in g/h (y-axis) and weightings of queue and brake reward (x-axis). We observed that a combination of both queue and brake reward was necessary to achieve the lowest CO_2 emissions.

Interestingly, we found that the combination ratio of $(queue, brake) = (0.5, 0.5)$ provided the best performance across both traffic scenarios. Additionally, we observed that combinations close to $(queue, brake) = (1.0, 0)$ or $(0, 1.0)$ demonstrated similar performance to those combinations. This suggests that the agent focuses on only one reward parameter, which does not lead to the best outcome.

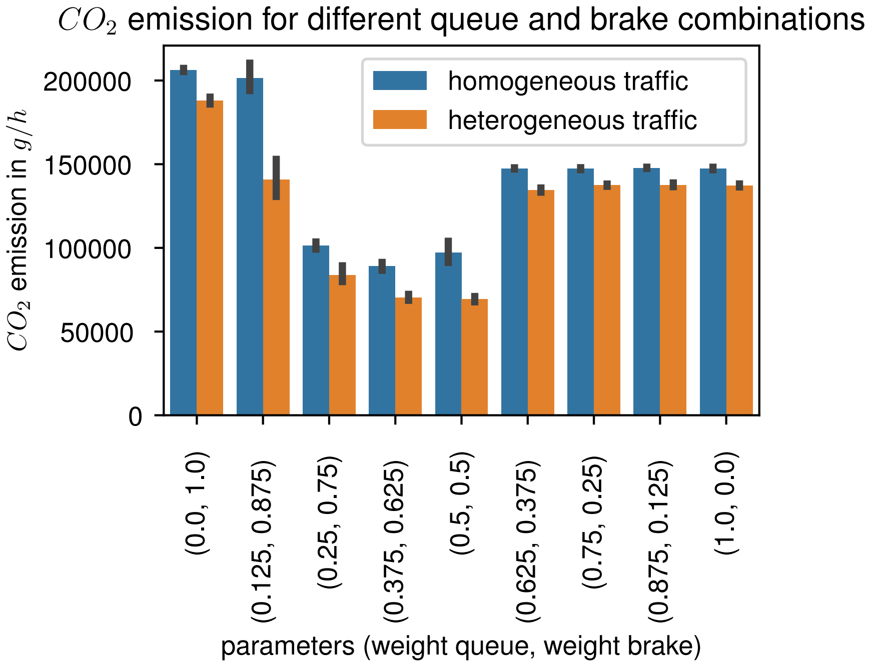


Figure 6. Episode CO_2 emission rates in g/h (y-axis) for DQN agents trained with various weighted combinations of queue and brake reward (x-axis).

For a combined queue speed reward, we would expect to see a similar trend in terms of the combination of rewards required to achieve good performance. However, as shown in Fig. 7, we observed that the level of queue metric must be zero (or close to zero) to achieve good performance.

Overall, our findings highlight the importance of reward parameter selection in training agents to optimize traffic flow and minimize CO_2 emissions.

6 Discussion

Next we discuss the results in terms of general properties for reward models, implications for modeling, and the threats to the validity of our results.

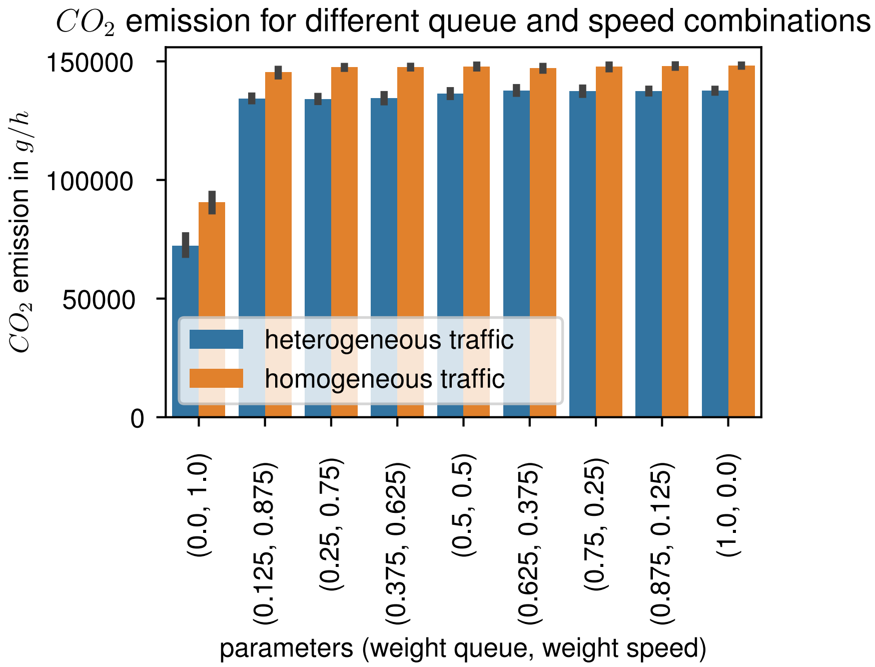


Figure 7. Episode CO_2 emission rates in g/h (y-axis) for DQN agents trained with various weighted combinations of queue and speed reward (x-axis).

6.1 Informativeness and Expressiveness

The two convergence curves shown in Section 5 correspond to: (1) how well the reward model *informs* the agent towards achieving the user's goal (CO_2) and (2) how well the reward model *expresses* the behavior of the emission. We call these two properties of the reward model *informativeness* and *expressiveness*. In other words, if the agent fails to converge to the optimal reward, we deem the reward model uninformative (see queue and emission reward in Fig. 5). Meanwhile, if the agent optimizes in the wrong direction, in our case positive correlation between reward and emissions (see brake reward in Table 2), then the reward model is not expressive.

These two properties are important because together they indicate if the agent is sufficiently aligned to the user's goals (minimizing CO_2 emissions). The judgment of sufficient alignment depends on how informative and expressive a reward model is. This is challenging because *informativeness* and *expressiveness* are continuous metrics based, respectively, on the measures of distance (from optima) and correlation (between reward and goal). Therefore, for the purpose of illustration and discussion, we assumed two arbitrary thresholds, which we introduce next.

Informativeness. A reward model (R_{mod}) is informative ($I(R_{mod}) = 1$) if the distance between the reward at convergence (R_{con}) and the optimal reward (R_{opt}) is smaller than δ. Formally, we have

$$I(R_{mod}) = \begin{cases} 1 & \text{if } (|R_{con} - R_{opt}| < \delta) \\ 0 & \text{otherwise} \end{cases}, \tag{1}$$

where R_{con} is the episode reward and R_{opt} is R_{con} of the best performing agent regarding that reward.

Expressiveness. A reward model (R_{mod}) is *expressive* ($E(R_{mod}) = 1$) if the correlation ($Corr$) between the sequence of the agent's episode rewards (R) and the corresponding episode CO_2 emissions (G) has a certain direction (positive or negative) and its magnitude is above a certain strength (ρ). The correlation should be negative ($\in [-1, \rho]$) if the user's goal G has to be minimized, otherwise positive ($\in [\rho, 1]$).

We formalized E for the case where G has to be minimized.

$$E(R_{mod}) = \begin{cases} 1 & \text{if } (Corr(R, O) \in [-1, \rho]) \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where the magnitude ρ depends on the use case. For the purpose of illustration and discussion, we set next $|\rho| \geq 0.30$, which corresponds to at least a medium strength correlation [42] and be negative (as it minimizes emissions), hence, the threshold becomes $Corr(\cdot) \in [-1, -0.30]$.

Applying these formulas (Eq. (1) and Eq. (2)) as threshold criteria for classification, we populated a Venn diagram (Fig. 8) with the results from Section 5. The intersection area shows the reward models that are both expressive and informative, hence, they are considered to be sufficiently aligned with users' goals (minimize CO_2 emissions). Only the brake reward is considered not expressive, whereas queue, emission, and queue-speed rewards are considered non-informative. Next, we discuss the implications of this classification.

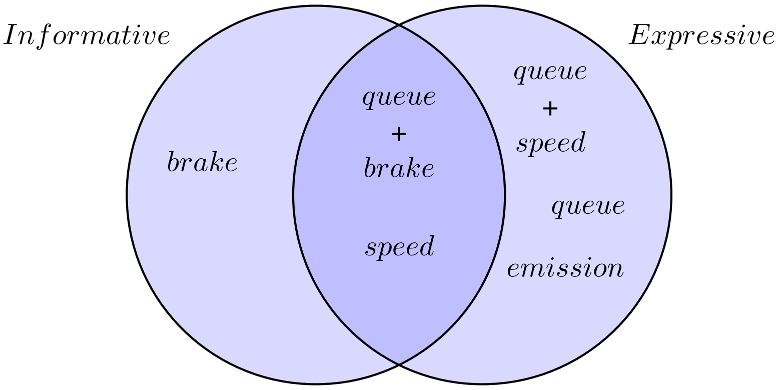


Figure 8. Classification of the rewards in terms of their *informativeness*, *expressiveness* and *alignment* (intersection).

6.2 Implications

Independent properties. The informativeness property did not necessarily imply expressiveness, and vice versa. Therefore, one has to monitor both properties while designing reward models. This is an additional requirement that involves a careful study of the thresholds that lead to agent alignment – satisfying users' goals. **Uninformativeness detection.** Many of the reward models that were deemed uninformative showed a very early convergence to a local minimum, e.g., queue reward and emission reward – their green curves follow a step-like function (Fig. 5). This suggests that the reward models provided a target that was too easy to learn; in other words, the agent is overfitting to the data collected in the first epoch.

Combining metrics. The design of the reward model should therefore incorporate metrics that make learning more challenging, for instance, with properties that are less correlated with emissions (lower *expressiveness*). This might explain why a combination of brake (low *expressiveness*) and queue (low *informativeness*) produced a sufficiently aligned agent, minimizing CO_2 emissions. Looking at the convergence curve of the proxy reward, green curve in Fig. 5, we can see diminishing returns over time, which suggests an increasing degree of difficulty for the agent to learn better policies as the training progresses. In other words, relative to the queue reward model, adding a brake metric made the learning more difficult. Conversely, adding the queue metric to the brake reward model provided the *expressiveness* that was missing.

Complementary properties. However, looking for complementary properties is not enough. Take, for instance, speed and queue metrics. Although the speed reward model is complementary to the queue reward model regarding *informativeness* and *expressiveness* – speed reward has higher correlation with emissions than the queue reward (see Table 2) – the combination of queue-speed did not produce an aligned agent. As we can see in Fig. 8, queue+speed convergence is categorized as *expressive*, but not *informative*. This is confirmed by the sensitivity analysis of the parameter weights for speed+queue combined reward (see Fig. 7).

Reward parameterization. Choosing the right traffic metrics to combine is not enough. One still has to decide on the weights that each metric should have in the reward model. While for the queue-brake reward we showed an optimum region (see Fig. 6), there is no guarantee that the combination of other metrics would present the same global optimum. This is important to design methods that systematically and efficiently look for the optimal parameterization. The shape of this parameterization space determines how informative and expressive a reward model should be to be considered aligned to the users' goal. Because a search in this space could be seen as a balance between exploitation (following an informative signal) and exploration (expressing desired behavior), one has to decide how to measure these properties. We note that assuming that these properties have uniform values during training is not realistic.

Property uncertainty. Defining how expressive or how informative a reward is might require new properties, for instance, properties that evaluate the uncertainty in the learning (convergence) process. The brake reward model illustrates this case, where there is larger than 10% variance in reward (green curve in Fig. 5) in the second half of training. This makes it challenging to decide how many training steps to execute or when training should stop, because slightly different stopping points could produce very different policies. Ideally, an engineer would like to know about the trade-off between reward model simplicity (only use the brake metric) and the risk of suboptimal rewards (high uncertainty at convergence).

6.3 Threats to Validity

Threats to validity [43], [44] act in ways that can hinder the reproducibility of the experimental results and corresponding interpretations.

Internal Validity evaluates if the causes of the measured effects can be attributed to our experimental design decisions [45]. In our case, we chose a benchmark (the best reward across agents - dotted lines in Fig. 5) and a set of proxy reward metrics (*speed*, *brake*, *queue*). We computed the effects on CO_2 emissions by varying the weights of metrics on combined reward models (e.g., X-axis in Fig. 6). When we claim that a given reward model is more or less informative or expressive, we are interpreting a measurement, i.e., the effect of a parameterization choice, that can still be confounded by what we did not control for, i.e, the other metrics not included in the given reward model, which might still indirectly affect the CO_2 emissions. To improve internal validity, we suggest more extensive simulations with more complex scenarios, for instance, by including real-world data.

External Validity discusses the situations in which the research outcomes might not generalize beyond the current experimental setup [45] comprised by both dataset and models parameters. Concerning the dataset, we showed similar results in two distinct scenarios of traffic demand. Although this might be a straightforward mitigation of the external validity threat (Fig. 6), a recent survey [2] reported that only seven out of 21 studies evaluated their models under distinct traffic scenarios. With respect to parameterization, we showed that certain pairs of weights for the queue-break reward produced suboptimal CO_2 emissions (see the extremes of the bar chart in Fig. 6). This highlights the challenge to generalize the combined reward results across a range of parameter values.

Construct Validity concerns the situations for which the performance indicators (thresholds) do not measure the actual concepts (constructs) [45]. This might happen because of bias in data generation, incorrect definitions, or inappropriate analysis methods (see Statistical Conclusion Validity). In our study, the mismatch between thresholds and the convergence properties (constructs) can happen through misspecification of the reward model and the properties themselves. One example is mistakenly deeming a reward model to be informative or expressive enough, when it is not. The reason for the mistake could be an inappropriate threshold or a reward model that is incomplete. To mitigate that, we specified convergence properties that are independent of the traffic signal control domain, but can be easily instantiated by choosing classification thresholds that are meaningful to what a user consider to be a sufficiently aligned agent reward model.

Statistical Conclusion Validity concerns the violations in the assumptions of the adopted statistical methods [45]. One example of possible violation is wrong assumption of normal data distributions. As we worked with small samples of reward outcomes, we adopted a non-parametric method (Kendall-tau) to compute the correlations, which we reported with their corresponding p-values (Table 2). Regarding conclusions about categorization within the two properties (*informativeness* and *expressiveness*), although we specified thresholds that were appropriate to discriminate among convergence curves, we did not take into account the inherent uncertainty in the convergence curves. A possible improvement could be to incorporate uncertainty measurements to the convergence analysis, for instance, the reward variance at the late training stages (so to ideally minimize it).

7 Conclusion

In the theory of bounded rationality [46], agents are bounded in their learning by the quality of the information they can access. We investigated this essential limitation in terms of the reward model, which we evaluated concerning the agent's alignment with the users' goals. Our main result is that, for the agent alignment with the goal of minimizing CO_2 emissions, it is *necessary* that the corresponding reward model formulation be both *expressive* and *informative*.

7.1 Results and Contributions.

Results. We showed that not all reward models are sufficiently aligned with users' goals (e.g., the models outside the intersection set depicted in Fig. 8). These results were reproduced in two distinct traffic scenarios. The sufficiently aligned reward models shared the characteristic of being both *informative* and *expressive*. However, the result from queue+speed indicated that to determine if an agent is aligned, it is not *sufficient* to look at the properties of the single-metric based rewards. Only after combining the individual traffic flow metrics into a properly parameterized reward formulation, one can ultimately assess the agent alignment (again by evaluating its convergence properties).

Contributions. We provided two systematic analyses: (1) a property-based paradigmatic classification for explaining the failure of an agent to align with users' goals and (2) a sensitivity analysis for explaining the challenges of aligning combined reward models with CO_2 emission goals.

7.2 Future work

Towards principles for reward model selection. We showed that combining complementary metrics worked to some extent. However, some outcomes are still counter-intuitive, i.e., we do not know how to predict good and bad combinations based on the properties of single-metrics rewards. This is critical, because one still has to rely on post hoc explanations (as we showed), instead of relying on principles to prioritize reward model combinations systematically.

Reproducibility in more challenging scenarios. A natural step is to reproduce our findings in more complex situations, for instance, incorporating real-data[4] to the simulations and a larger set of traffic flow metrics. Besides creating opportunities to falsify our current claims, we could explore more challenging questions like the effects of partial observability and confounding in reward modeling for agent alignment in TSC.

Alternative convergence property formulations. In order to evaluate non-linear relationships, we plan to study *expressiveness* in terms of mutual information or metrics like Wasserstein distance. Concerning *informativeness*, we plan to look at methods that incorporate variance as a criterion of quality of convergence.

Data availability statement

To promote the reproducibility of our results, we made our experimental setup, source code, and dataset publicly available[5], which further facilitates extension of our results and serves as a baseline for comparing different approaches w.r.t. reward modeling. Note that the convergence properties are independent of the choice of traffic flow metrics, therefore, they can be instantiated for TSC environments with different sensors and state-space configurations.

Author contributions

Max Eric Henry Schumacher contributed with the software implementation, performing the experiments for data/evidence collection, and writing the original draft. Christian Medeiros Adriano contributed with the conceptualization, methodology, and writing the original draft. Holger Giese contributed with the conceptualization and supervision.

Competing interests

The authors declare that they have no competing interests.

[4] This would also help bridge the gap in which real-world traffic data is still a minority among traffic simulation studies [2]

[5] https://github.com/EricSchuMa/reward-design-TSC. Our repository extends the publicly available OpenAI Gym interface for SUMO [47].

Funding

The authors are grateful for the funding and infrastructure provided by the Hasso-Plattner Institute.

References

[1] C. Louw, L. Labuschagne, and T. Woodley, "A comparison of reinforcement learning agents applied to traffic signal optimisation," in *SUMO Conference Proceedings*, vol. 3, 2022, pp. 15–43.

[2] H. Wei, G. Zheng, V. Gayah, and Z. Li, "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation," *ACM SIGKDD Explorations Newsletter*, vol. 22, no. 2, pp. 12–18, 2021, Publisher: ACM New York, NY, USA.

[3] A. Haydari and Y. Yilmaz, "Deep Reinforcement Learning for Intelligent Transportation Systems: A Survey," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–22, 2020, ISSN: 1558-0016. DOI: 10.1109/TITS.2020.3008612.

[4] H. Wei, C. Chen, G. Zheng, *et al.*, "Presslight: Learning max pressure control to coordinate traffic signals in arterial network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1290–1298.

[5] G. Zheng, X. Zang, N. Xu, *et al.*, "Diagnosing reinforcement learning for traffic signal control," *arXiv preprint arXiv:1905.04716*, 2019.

[6] J. Kim, S. Jung, K. Kim, and S. Lee, "The real-time traffic signal control system for the minimum emission using reinforcement learning in v2x environment," en, *Chemical Engineering Transactions*, vol. 72, pp. 91–96, Jan. 2019, ISSN: 2283-9216. DOI: 10.3303/CET1972016. [Online]. Available: https://www.cetjournal.it/index.php/cet/article/view/CET1972016 (visited on 09/21/2022).

[7] A. Haydari, M. Zhang, C.-N. Chuah, and D. Ghosal, "Impact of deep rl-based traffic signal control on air quality," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, ISSN: 2577-2465, Apr. 2021, pp. 1–6. DOI: 10.1109/VTC2021-Spring51267.2021.9448639.

[8] H. Wei, G. Zheng, V. Gayah, and Z. Li, "A Survey on Traffic Signal Control Methods," *arXiv:1904.08117 [cs, stat]*, Jan. 2020, arXiv: 1904.08117. [Online]. Available: http://arxiv.org/abs/1904.08117 (visited on 01/16/2022).

[9] A. C. Egea, S. Howell, M. Knutins, and C. Connaughton, "Assessment of Reward Functions for Reinforcement Learning Traffic Signal Control under Real-World Limitations," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, ISSN: 2577-1655, Oct. 2020, pp. 965–972. DOI: 10.1109/SMC42975.2020.9283498.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[11] D. Krajzewicz, M. Behrisch, P. Wagner, R. Luz, and M. Krumnow, "Second Generation of Pollutant Emission Models for SUMO," en, in *Modeling Mobility with Open Data*, M. Behrisch and M. Weber, Eds., Series Title: Lecture Notes in Mobility, Cham: Springer International Publishing, 2015, pp. 203–221, ISBN: 978-3-319-15023-9 978-3-319-15024-6. DOI: 10.1007/978-3-319-15024-6_12. [Online]. Available: http://link.springer.com/10.1007/978-3-319-15024-6_12 (visited on 11/21/2022).

[12] E. Yudkowsky, "The AI alignment problem: why it is hard, and where to start," *Symbolic Systems Distinguished Speaker*, 2016.

[13] B. Christian, *The alignment problem: Machine learning and human values*. WW Norton & Company, 2020.

[14] J. Leike, D. Krueger, T. Everitt, M. Martic, V. Maini, and S. Legg, “Scalable agent alignment via reward modeling: A research direction,” *arXiv preprint arXiv:1811.07871*, 2018.

[15] P. Christiano, *”clarifying ai alignment”*, 2018. [Online]. Available: https://ai-alignment.com/clarifying-ai-alignment-cec47cd69dd6.

[16] T. Everitt, R. Carey, E. D. Langlois, P. A. Ortega, and S. Legg, “Agent incentives: A causal perspective,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 11 487–11 495.

[17] E. Hubinger, C. van Merwijk, V. Mikulik, J. Skalse, and S. Garrabrant, “Risks from learned optimization in advanced machine learning systems,” *arXiv preprint arXiv:1906.01820*, 2019.

[18] L. Ouyang, J. Wu, X. Jiang, *et al.*, *Training language models to follow instructions with human feedback*, arXiv:2203.02155 [cs], Mar. 2022. [Online]. Available: http://arxiv.org/abs/2203.02155 (visited on 12/08/2022).

[19] T. Everitt, M. Hutter, R. Kumar, and V. Krakovna, “Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective,” *Synthese*, vol. 198, no. Suppl 27, pp. 6435–6467, 2021.

[20] M. Cohen, M. Hutter, and M. Osborne, “Advanced artificial agents intervene in the provision of reward,” *AI Magazine*, vol. 43, no. 3, pp. 282–293, 2022.

[21] J. Clark and D. Amodei, *Faulty reward functions in the wild*, Dec. 2016. [Online]. Available: https://openai.com/blog/faulty-reward-functions/.

[22] J. Skalse and A. Abate, “Misspecification in inverse reinforcement learning,” *arXiv preprint arXiv:2212.03201*, 2022.

[23] A. D’Amour, K. Heller, D. Moldovan, *et al.*, “Underspecification presents challenges for credibility in modern machine learning,” *Journal of Machine Learning Research*, 2020.

[24] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[25] M. Riedmiller, “Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method,” in *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, Springer, 2005, pp. 317–328.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.

[27] A. Y. Ng, D. Harada, and S. Russell, “Policy invariance under reward transformations: Theory and application to reward shaping,” in *Icml*, Citeseer, vol. 99, 1999, pp. 278–287.

[28] J. Sorg, R. L. Lewis, and S. Singh, “Reward design via online gradient ascent,” in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23, Curran Associates, Inc., 2010. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2010/file/168908dd3227b8358eababa07fcaf091-Paper.pdf.

[29] S. Singh, R. L. Lewis, and A. G. Barto, “Where do rewards come from,” in *Proceedings of the annual conference of the cognitive science society*, Cognitive Science Society, 2009, pp. 2601–2606.

[30] Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros, “Large-scale study of curiosity-driven learning,” in *Seventh International Conference on Learning Representations*, 2019, pp. 1–17.

[31] K. Belhassine, J. Renaud, L. Coelho, and V. Turgeon, “Signal priority for improving fluidity and decreasing fuel consumption,” in *SUMO Conference Proceedings*, vol. 3, 2022, pp. 159–169.

[32] J. E. L. Quichimbo, J.-A. Moreno-Perez, E. Lorenzo-Sáez, *et al.*, "Estimation of green house gas and contaminant emissions from traffic by microsimulation and refined origin-destination matrices: A methodological approach," in *SUMO Conference Proceedings*, vol. 1, 2020, pp. 27–37.

[33] B. De Coensel, A. Can, B. Degraeuwe, I. De Vlieger, and D. Botteldooren, "Effects of traffic signal coordination on noise and air pollutant emissions," en, *Environmental Modelling & Software*, vol. 35, pp. 74–83, Jul. 2012, ISSN: 1364-8152. DOI: 10.1016/j.envsoft.2012.02.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364815212000576 (visited on 09/06/2021).

[34] Y. Zhang, X. Chen, X. Zhang, G. Song, Y. Hao, and L. Yu, "Assessing effect of traffic signal control strategies on vehicle emissions," en, *Journal of Transportation Systems Engineering and Information Technology*, vol. 9, no. 1, pp. 150–155, Feb. 2009, ISSN: 1570-6672. DOI: 10.1016/S1570-6672(08)60050-1. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570667208600501 (visited on 09/06/2021).

[35] H. Rakha, M. Van Aerde, K. Ahn, and A. Trani, "Requirements for evaluating traffic signal control impacts on energy and emissions based on instantaneous speed and acceleration measurements," en, *Transportation Research Record*, vol. 1738, no. 1, pp. 56–67, Jan. 2000, Publisher: SAGE Publications Inc, ISSN: 0361-1981. DOI: 10.3141/1738-07. [Online]. Available: https://doi.org/10.3141/1738-07 (visited on 09/06/2021).

[36] X. Liang, X. Du, G. Wang, and Z. Han, "Deep Reinforcement Learning for Traffic Light Control in Vehicular Networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243–1253, Feb. 2019, arXiv:1803.11115 [cs, stat], ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2018.2890726. [Online]. Available: http://arxiv.org/abs/1803.11115 (visited on 02/15/2023).

[37] L. Prashanth and S. Bhatnagar, "Reinforcement learning with average cost for adaptive control of traffic lights at intersections," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2011, pp. 1640–1645.

[38] W. Genders and S. Razavi, "Evaluating reinforcement learning state representations for adaptive traffic signal control," *Procedia computer science*, vol. 130, pp. 26–33, 2018.

[39] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," *arXiv preprint arXiv:1611.01142*, 2016.

[40] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv, Tech. Rep. arXiv:1412.6980, Jan. 2017, arXiv:1412.6980 [cs] type: article. DOI: 10.48550/arXiv.1412.6980. [Online]. Available: http://arxiv.org/abs/1412.6980 (visited on 08/29/2022).

[41] M. G. Kendall, "Rank correlation methods.," 1948. [Online]. Available: https://archive.org/details/rankcorrelationm0000kend.

[42] H. Akoglu, "User's guide to correlation coefficients," *Turkish Journal of Emergency Medicine*, vol. 18, no. 3, pp. 91–93, 2018, ISSN: 2452-2473. DOI: https://doi.org/10.1016/j.tjem.2018.08.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2452247318302164.

[43] D. T. Campbell and T. D. Cook, "Quasi-experimentation," *Chicago, IL: Rand Mc-Nally*, 1979.

[44] W. R. Shadish, T. D. Cook, and D. T. Campbell, *Experimental and quasi-experimental designs for generalized causal inference.* Houghton, Mifflin and Company, 2002.

[45] R. J. Wieringa, *Design science methodology for information systems and software engineering*. Springer, 2014.

[46] H. A. Simon, “Bounded rationality,” *Utility and probability*, pp. 15–18, 1990.

[47] L. N. Alegre, *SUMO-RL*, https://github.com/LucasAlegre/sumo-rl, 2019.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.223

Published: 29 June 2023

Calibration of a Microscopic Traffic Simulation in an Urban Scenario Using Loop Detector Data

A Case Study within the Digital Twin Munich

Andreas Keler[1,2][https://orcid.org/0000-0002-2326-1612], Andreas Kunz[2], Sasan Amini[2][https://orcid.org/0000-0002-1675-4945], and Klaus Bogenberger[2][https://orcid.org/0000-0003-3868-9571]

[1] Applied Geoinformatics, University of Augsburg, Germany

[2] Chair of Traffic Engineering and Control, Technical University of Munich, Germany

Abstract. Travel demand is an essential input for the creation of traffic models. However, estimating travel demand to accurately represent traffic behaviour usually requires the collection of extensive sets of data on traffic behaviour. Traffic counts are a comparably cost effective and reproducible source of information on travel demand. The utilisation of traffic counts to estimate demand is commonly found in the literature as the static and dynamic O-D estimation problem. A variety of approaches have been developed over recent decades to tackle this problem. Usually initial estimates of the O-D matrix are calibrated by utilising traffic counts and considering different assignment models. Other approaches for the estimation of travel demand solely based on traffic measurements can be found in the simulation software SUMO. The present work demonstrates the systematic development of a network model in SUMO in the inner city of Munich. In a sample network the estimation of travel demand through the tools flowrouter and routeSampler is tested by utilising flow measurements from induction loop detectors. The tests delivered unsatisfactory results, which is proven through observations of traffic flows in the resulting simulations as well as comparisons to historic traffic counts. The lack of sufficient detector data and the complexity of the sample network are discussed as the main reasons for the results. It is concluded that the applied tools should be tested in future studies with a more extensive dataset to perform a more comprehensive review of both tools. Therefore, we deliver specific requirements based on the network example of Munich.

Keywords: Induction Loop Detectors, Calibration, Travel Demand Estimation, Urban Digital Twin

1. Introduction

The basis for the model was an automatically generated network of Munich’s inner city which was previously developed by the Chair of Traffic Engineering and Control (at TUM). The network is based on data from OSM and was transformed into a SUMO network [1,2]. This network was then manually edited and refined in this project work. Those improvements were made based on official site plans of the intersections which were provided by the City of Munich. These site plans also feature information on the location of loop detectors. A supplementary CSV-file containing information on the IDs and names of the detectors was also provided. In addition to the site plans, aerial images were used and site visits were made to edit and verify the model.

Aside from the data used for generating the network geometry, two sources of information on traffic signal programs were used to generate the control infrastructure in the area. On the one hand, original signal plans for a selection of signalised intersections were provided by the City of Munich. It should be mentioned that some of these signal programs are outdated due to redesigns of the junctions over recent years though they were in some cases the only source of information available. On the other hand, the City of Munich provided signal data for all signalised intersections in the city between 11:30 am and 11:00 pm from the 25th of July 2022. The dataset reports information on the status of the signal heads at all junctions over the course of the day. From this data, signal plans were reverse engineered. However, some traffic lights did not report data which may be related to erroneous communication from the traffic signals or ongoing construction works during which the regular signal heads are deactivated.

Moreover, the City of Munich provided data from induction loop detectors throughout the city of Munich between 11:30 am and 11:00 pm from the 25th of July 2022. The dataset reports the traffic volumes, speeds and occupancy measured at every detector in the city. However, speeds are only measured at few detectors. Aside from the data from induction loop detectors, traffic counts at different intersections within the study area were made available. The year of origin of the historic counts range from 2010 to 2018. The traffic counts were not directly used for modelling travel demand but were consulted to determine the analysis period as well as to perform plausibility checks of the resulting simulation.

2. Network Development

In the following, we describe the development of the network model. At first, it is described in detail how the network geometry was edited and examples for the process are given. Secondly, the implementation of the control infrastructure is discussed. This includes a description of the structure of the signal dataset which was provided by the City of Munich as well the analysis of the data. Additionally, information on the implementation of signal programs in the model is given.

2.1 Network Geometry

The first step in refining the initial raw network was to remove unneeded network elements from the model. These were namely all cycling paths and pedestrian paths which were remnants of the network conversion process. Furthermore, the network was manually cropped at the river Isar meaning that all edges on the eastern side of the river were deleted.

Following that, all intersections were checked and adjusted according to the information contained in the available site plans. These plans were always compared to other available information gained from aerial images and site visits to check whether the site plans show the most recent state of the junction. This was particularly necessary when the site plans indicated ongoing construction works since in some cases these construction works are already completed. When no site plans were available at all, the road geometry was edited solely based on the secondary sources, i.e. aerial images and site visits.

Once the intersection geometry was satisfactory, the induction loop detectors could be placed in accordance with the site plans and the supplementary CSV-file. It must be stated that the detector placement was done manually which means that their location in the model might not match the exact location in the real world. Since the detectors were only used to link the counting data to the edges, this simplification is valid. However, should the detectors be used in the future for other applications such as the actuation of traffic lights, the placement of the detectors should be reviewed. Furthermore, only those detectors that are relevant for road vehicles were placed.

Lastly, some additional refinements had to be made which included removing implausible turnarounds. These may lead to implausible routes and errors in the simulation. On the one hand, these turnarounds existed at the network boundaries which can lead to vehicles not being able to exit the network or driving into dead ends and turning around to continue their route which is implausible. On the other hand, especially at junctions along residential streets, turnarounds were possible which is not plausible for example due to space constraints in the real world. Aside from removing these turnarounds, node clusters were joined to remove unnecessary network elements.

To conclude this, the process of modelling the network geometry is illustrated at the example of two intersections. The first example, shown in Figure 1, depicts the situation at Sendlinger-Tor-Platz. The graphic on the left depicts the state of the junction in the initial network and the image on the right depicts the final intersection layout. Firstly, all remaining foot and cycling paths were removed (1). The next step was the adjustment of the edge geometry (2). For example, the right-turning lane at the western leg of the central junction branches off from the left-turning and through lane in the initial network. Due to ongoing construction works in the area, this lane currently runs in parallel to the other two lanes. Lastly, the connections were corrected (3). In the initial network the eastern leg of the intersection featured a mixed lane for right-turning and through traffic and an additional lane dedicated to through traffic. In reality, one lane for through traffic and one for right-turning traffic exists. It should be mentioned that the network geometry is not an exact depiction of the real world though it is a close approximation and represents the main characteristics of the junction.

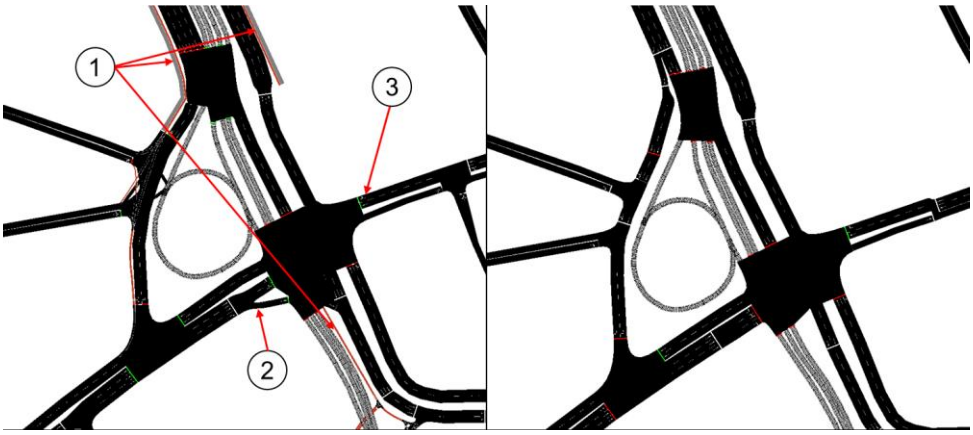


Figure 1. Sendlinger-Tor-Platz in the initial network (left) and the edited network (right).

A more complex example can be found at the intersection Lenbachplatz/ Elisenstraße. Figure 2 depicts the initial network on the left and the edited junction on the right. Initially the junction was heavily simplified and additional signal heads in the centre of the junction were missing. The intersection was thus edited as follows. The process started again by removing all residual foot- and cycling paths (1). Then the junction was split into two signalised intersections and the junction shape was manually adjusted so that it accurately represents the shape of its real-world counterpart (2). This also included the correction of all possible turning manoeuvres between all incoming and outgoing edges as well as an adjustment of the tram tracks. Lastly, all induction loop detectors were placed in accordance with the site plans and the supplementary CSV-file (3).

The same procedure was followed at all other junctions across the network though the focus was set on those junctions for which site plans were available. These were mostly the more complex major intersections of the road network. Junctions of minor importance in the secondary road network usually only required minor corrections, e.g. checking for the possible turning manoeuvres.

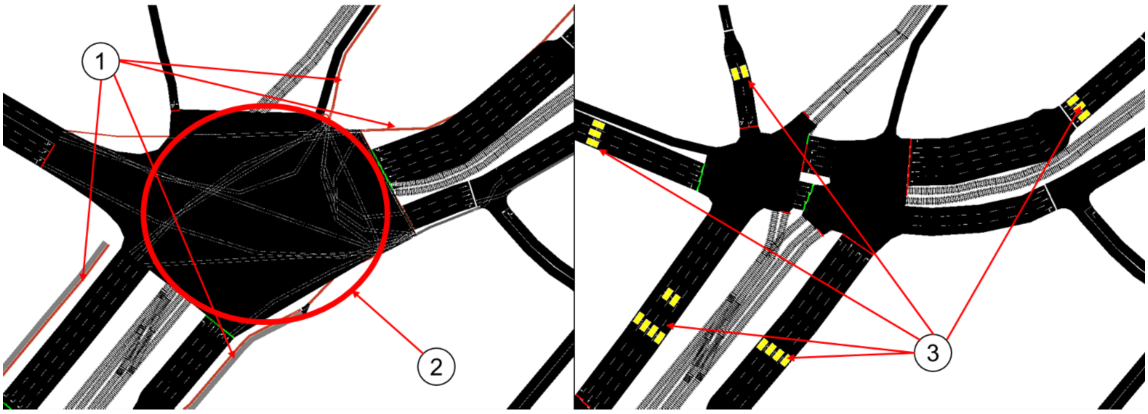
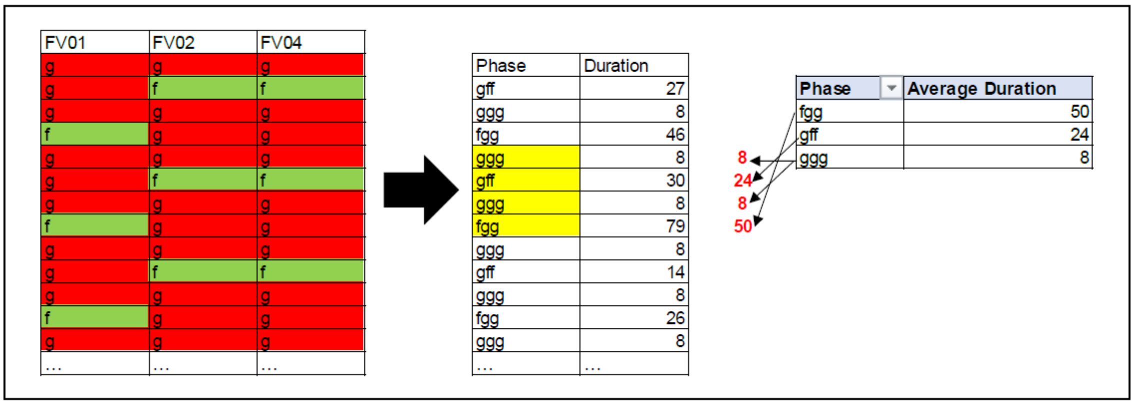


Figure 2. Intersection Lenbachplatz/ Elisenstraße in the initial network (left) and the edited network (right).

2.2 Traffic Signal Programs

Many traffic lights especially those located on main roads within the model's extent act traffic responsive or feature programs with public transport prioritisation or actuation. However, since public transport was not modelled and the control algorithms for actuated traffic signals were not known, it was decided to simplify the traffic signal programs and to design them as programs with a fixed cycle.

First and foremost, the signal programs were created by utilising a dataset containing the sequence of signal status of all signal heads at every intersection in Munich. From this data simplified signal programs with a fixed cycle were reverse engineered. An example of such a procedure is pictured in Figure 3 for the example of LSA 103. It bases on averaging the available values.



FV01	FV02	FV04
g	g	g
g	f	f
g	g	g
f	g	g
g	g	g
g	f	f
g	g	g
f	g	g
g	g	g
g	f	f
g	g	g
f	g	g
g	g	g
…	…	…

Phase	Duration
gff	27
ggg	8
fgg	46
ggg	8
gff	30
ggg	8
fgg	79
ggg	8
gff	14
ggg	8
fgg	26
ggg	8
…	…

Phase	Average Duration
fgg	50
gff	24
ggg	8

Figure 3. Determination of phase and cycle duration at the example of LSA 103.

A limitation of the dataset is that it only reports the status of active signal heads as locked ("g") or free ("f"). This means that amber times of the signal heads are unknown. Because of that the signal plans implemented in the model also only feature green and red phases which can lead to emergency braking by the simulated vehicles in case a signal head switches from red to green when a vehicle is close to the stop line.

In total, 109 traffic lights in the network feature signal programs which were reverse engineered from the signal dataset, 13 traffic lights were supplied with signal programs from original signal plans and 20 junctions use actuated signal programs that were automatically generated by SUMO.

3. Application of flowrouter and routeSampler

Subsequently, the two existing tools of the SUMO package for count-based travel demand generation are tested. On the one hand, demand was generated by using the tool flowrouter. As described previously the tool generates routes and flows, i.e. numbers of vehicles per route, based on detector data. On the other hand, the tool routeSampler was used. This tool requires a set of initial routes as well as counting data as input and then selects and multiplies routes from the initial set in such a way that the detector data is matched.

As pointed out previously, a small network was extracted from the larger one for the test of both tools. Figure 4 shows the large network on the left and the cropped network on the right. The cropped network consists mainly of Sonnenstraße and the important incoming and outgoing streets. Streets of minor significance in terms of their traffic volumes were excluded from the network in order to simplify the model. In total, the network model features eight signalised intersections.

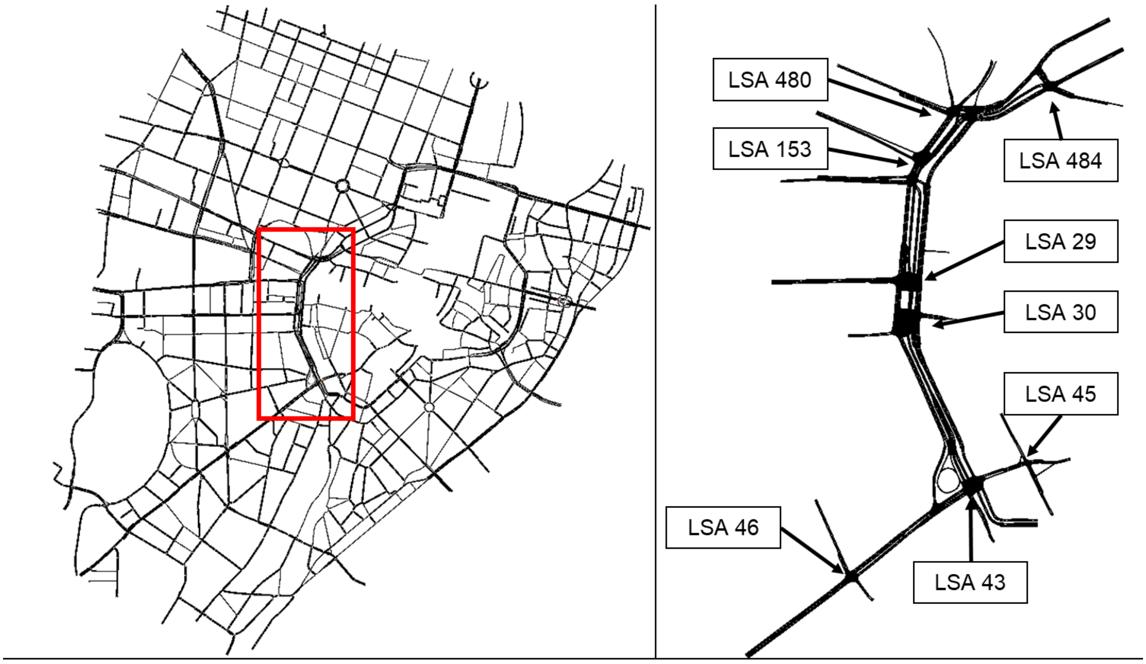


Figure 4. Full network model (left) and test network (right) in SUMO.

The evening peak hour was selected as the test period for the scenario. According to the historic traffic counts, the evening peak hour usually occurs between 4:00 pm and 6:45 pm. It was decided to use the period between 4:00 pm and 5:00 pm as the simulation period. In addition, 15 minutes were added before and after this period so that all vehicles detected within the analysis period can enter and exit the network.

Aside from LSA 43 at Sendlinger-Tor-Platz all intersections in the network are equipped with induction loop detectors. These detectors needed to be filtered at first. Two conditions had to be fulfilled by the detectors so that their data could be used for flowrouter and routeSampler.

Firstly, all lanes of an edge should be equipped with detectors since only then values over a cross-section are known. Secondly, the detectors should report plausible values. The first condition was checked through a manual inspection of the network. This led to the removal of three detectors at the intersection LSA 29. Then, the vehicles counted across all time intervals were summed up, to identify erroneous detectors which reported implausible values within the dataset. All detectors with a sum of zero across the dataset were excluded from the model. This included all detectors at LSA 153, LSA 480 and four detectors at LSA 29. Lastly, the three remaining detectors at the western leg of LSA 29 were removed since the sums of each individual detector were the same. Moreover, the values from all individual measurement intervals were identical. The remaining detectors are located at LSA 30, LSA 45, LSA 46 and LSA 484. At LSA 45 and LSA 46 not all incoming edges are equipped with detectors. The results of the analysis are illustrated in Figure 5.

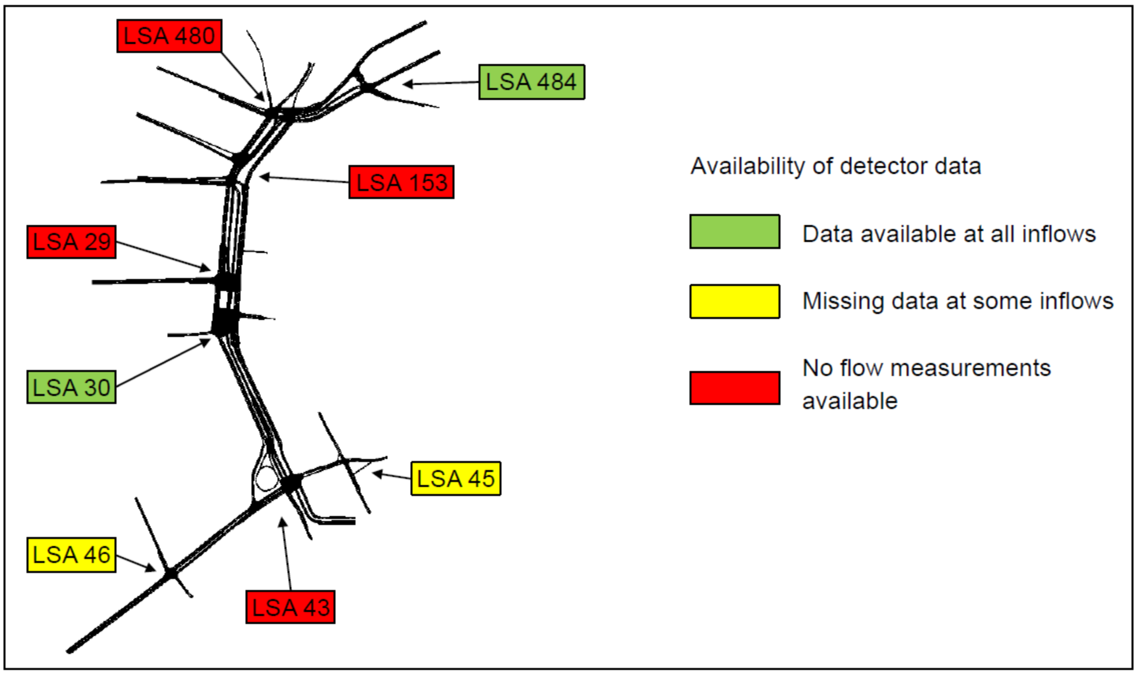

LSA 480
LSA 484
LSA 153
LSA 29
LSA 30
LSA 45
LSA 46
LSA 43
Availability of detector data
🟩 Data available at all inflows
🟨 Missing data at some inflows
🟥 No flow measurements available

Figure 5. Availability of detector data at intersections in the test network.

At first, the application of flowrouter was tested. The tool requires as input a network-file, a file that specifies the location and ID of the detectors and a file containing the flow measurements from the detectors. The flows file can either be provided as a CSV file or TXT file with a “;” as separator between columns. The file contains a column “Detector”, specifying the detector ID, a column “Time” containing the time interval in minutes and lastly a column “qPKW” which describes the vehicles counted at the respective detectors during an interval. Additionally, vehicles may be classified into passenger cars and heavy goods vehicles and measured speeds can be added. However, since the detectors in this case only reported vehicle counts without further distinction between vehicle types, all counting data was inserted into the “qPKW” column.

Following the specification of the flows file, flowrouter was run in the command prompt. The setting - -respect-zero was set so that detectors were also considered which may have counted no vehicles across one- or multiple time intervals. In addition, the options - -lane-based and - - interval 15 were set. The first means that the values of all detectors across one edge are not aggregated but the counts from each individual lane are used. The latter indicates the aggregation interval of the traffic counts which is 15 minutes. The output from flowrouter comes in form of a route file and a flows file. The route file consists of all individual possible routes and the flow file describes the number of vehicles for each route.

In theory, the user can specify flow-restrictions as input for flowrouter so that certain implausi-ble routes are not considered by the tool. This was attempted by utilising the script implausibleRoutes.py, which allows blacklisting of certain routes according to a specified heuristic. However, all attempts to manipulate the output from flowrouter by using this tool failed. Thus, the developers of SUMO were contacted via the official forum and according to them it is currently not possible to combine flow-restrictions with the option - - lane-based in flowrouter. Tests without the option - - lane-based did not produce satisfactory results.

The routeSampler tool needs an initial set of routes stored in a route file and an edgedata file which contains the flow measurements for all edges. In comparison to flowrouter it is not possible to utilise lane-based counting data of each individual detector. Through the aggregation of measurements across all lanes certain information from the detectors are lost. This includes for example the number of vehicles on dedicated turning lanes. It is also possible to define turn-counts as input for routeSampler. This information could not be obtained from the induction loop detectors since the detectors are only placed at the inflows of the intersections and no downstream measurements were available. The following paragraph details the specification of the route file and the edgedata file.

At first, all possible O-D pairs were written manually into a trip file. It was assumed that there are no trips that start and end in the same direction. Additionally, it was checked whether a trip is possible or not due to turn-restrictions at intersections. This resulted in a total of 174 O-D pairs in the network. Then, the route definitions were obtained by using duarouter. Since there are no parallel streets in the network which allow alternative routes, the tool returned one route definition for each O-D pair. The edgedata file was then obtained by transforming the flows file used for flowrouter into the required format for routeSampler. This can be done through the tool edegedataFromFlow which sums the data from all detectors across an edge and assigns the resulting value to the edge. Then routeSampler was run with the route file and the edgedata file. The additional commandline specification - -edgedata-attribute qPKW had to be set so that routeSampler was able to read the counting data from the edgedata file. routeSampler returns a route file which contains information on the selected routes from the initial set and the start time of every individual generated vehicle.

4. Results and Discussion

The observation of the resulting simulations revealed that neither flowrouter nor routeSampler were able to produce plausible estimations for travel demand in the network. First and foremost, this is related to the lack of available counting data. As discussed in the previous section, only few detectors reported values on the day the data was taken from. Detector measurements on Sonnenstraße itself were only available at the incoming edges of LSA 29. Other measurements were available at LSA 45, LSA 46 and LSA 484. However, of these intersections only LSA 484 is equipped with detectors on all incoming edges. Both tools were not able to deliver plausible estimates with this limited dataset. In addition to the lack of data, the characteristics of the study area are another reason for the unsatisfactory outcome. At the intersections LSA 29, LSA 480 and LSA 484, turnarounds are possible on certain edges. The overestimation of these turnarounds led to congestion in both simulations. Congestion was also experienced in both tools on minor streets on which traffic flows were overestimated which exceeded the capacity of the respective traffic signals.

The following paragraphs give examples for the shortcomings of both simulations by qualitatively describing a selection of errors. In case of flowrouter, the turning ratios at LSA 29 and LSA 480 are compared between the simulation and historic counts. For this purpose, test detectors were implemented in SUMO to measure the respective traffic flows. The comparison is done to prove the simulation's shortcomings. The assumption for this plausibility check is that turning ratios have not changed significantly at the selected intersections between the day of the traffic count and the day the detector data was taken from. Since this superficial qualitative analysis already indicates that the output from both tools is implausible, a more in-depth quantitative analysis seems not sensible.

A general finding of the observation of the output from flowrouter is the overestimation of traffic flows which originate and end in the same direction. This leads to unusual high shares of turnarounds at different junctions. For example, many vehicles which start their trip in the northeast of the network drive to LSA 29 where they turnaround and return to the northeast. Similar observations can be made at LSA 480 where vehicles coming from the south turn around and return in the same direction. Additionally, congestion effects occur at different intersections. One example can be found at the western leg of LSA 29. Here, the high amount of right-turning traffic cannot be handled by the corresponding traffic light. The following paragraphs illustrate these observations at some examples. These are merely a selection of observations from the simulation.

Figure 6 depicts the situation at LSA 29 and the described congestion on the western leg of the intersection. Additionally, the unusual high share of vehicles that turn around can be seen on the northern leg of the intersection.

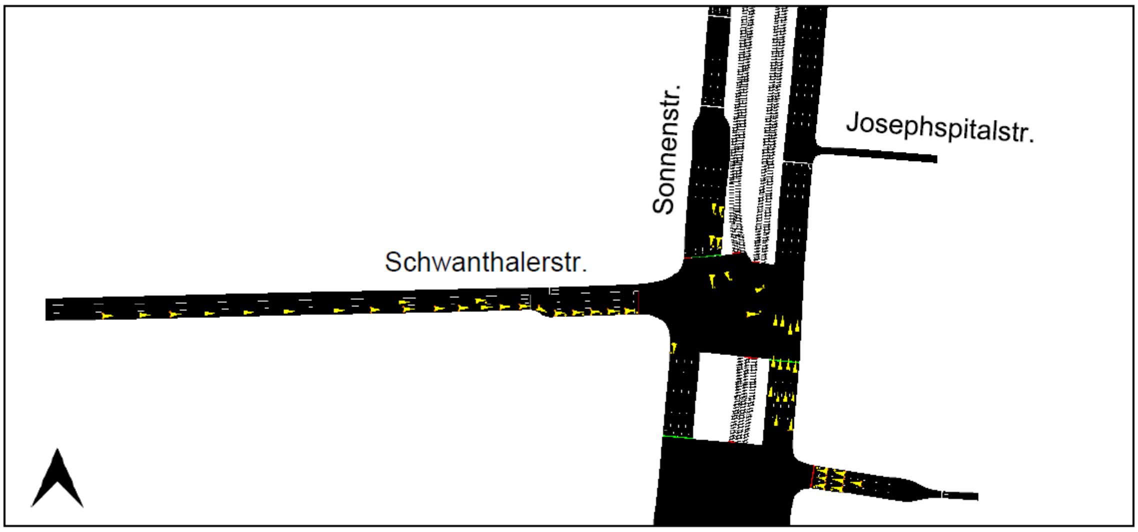


Figure 6. Availability of detector data at intersections in the test network.

Table 1 features the comparison of turning ratios between a historic traffic count and the simulation for the northern and western leg of the intersection. The historic count was made in the year 2014 by the city of Munich and contains information on daily traffic and both peak hours. For this comparison, the turning ratios from the evening peak hour are used. The table summarises the traffic flows to the northern and eastern leg of the junction as it was not possible to measure them individually in SUMO. However, observations of the simulation show that most of those vehicles drive into the northern direction.

Table 1. Comparison of turning ratios between a traffic count and the simulation at LSA 29.

From	Street Name	To	Historic Count	Simulation	Difference
North	Sonnenstr.	North or East	17%	56%	39%
		South	69%	44%	-26%
		West	13%	0%	-13%
West	Schwanthalerstr.	North or East	60%	40%	-20%
		South	40%	60%	20%

All in all, the comparison proves the discussed findings from observing the simulation. The share of vehicles which turn around at the northern leg of the intersection was overestimated. Additionally, it can be observed that no vehicles turn right. The turning ratios at Schwanthalerstraße were also not reproduced. While the counts indicate that 60 % turn left into Sonnenstraße or go straight into Josephspitalstraße, only 40 % of vehicles do so in the simulation.

Figure 7 depicts the situation at LSA 480. The figure particularly highlights the high number of vehicles on the leftmost lane at Sonnenstraße which perform a turnaround at the intersection. Additionally, the rightmost left-turning lane at the eastern leg of the junction is frequently congested over the simulation period. Observing these traffic flows in the simulation revealed that most of these vehicles turn into Prielmayerstaße at LSA 153. The historic traffic count at LSA 153 from the year 2018 shows that only 500 vehicles turn into Prielmayerstraße per day. Over the analysis period of the simulation (4:00 am to 5:00 pm) around 400 vehicles turned into Prielmayerstraße.

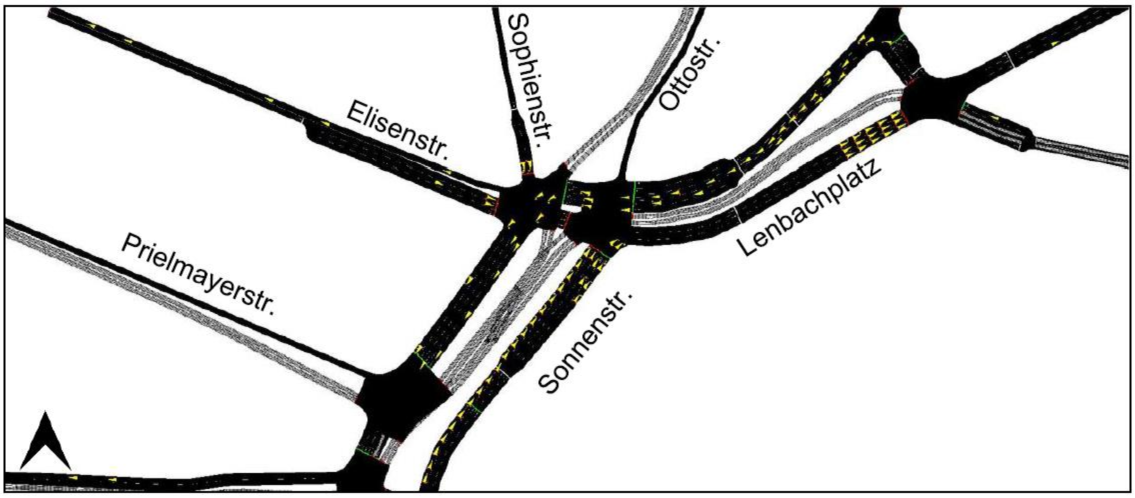


Figure 7. Simulation results at LSA 480 using flowrouter.

The comparison of turning ratios at LSA 480 is depicted in Table 2 for all legs of the intersection. The historic count was made in the year 2018 by the city of Munich and only contains information on daily traffic. Key findings of the comparison are that no vehicles turn right from Sophienstraße into Elisenstraße and no vehicles turn right from Elisenstraße into Sonnenstraße. The share of flows between the eastern leg and the western leg of the intersection are also comparably low. In the traffic counts, these flows account for 45 % of all incoming traffic flows from the east while in the simulation only 21 % of vehicles go straight into Elisenstraße. Interestingly, the share of vehicles turning around or left coming from the south is lower than in the traffic counts. However, most of these vehicles turn around while in the counts the share of turnarounds only makes up for around 1% of the inflow from Sonnenstraße.

Table 2. Comparison of turning ratios between a traffic count and the simulation at LSA 480.

From	Street Name	To	Historic Count	Simulation	Difference
North	Sophien-/ Ottostr.	South	91%	100%	9%
		West	9%	0%	-9%
East	Lenbach-platz	South	54%	70%	15%
		West	45%	21%	-24%
		North	1%	9%	8%
South	Sonnenstr.	West or South	21%	14%	-7%
		North	14%	7%	-7%
		East	65%	79%	14%
West	Elisenstr.	East	75%	100%	25%
		South	25%	0%	-25%

All in all, this qualitative description of shortcomings of the resulting simulation using the output from flowrouter proves that the tool was not able to produce plausible traffic flows from the limited amount of available detector data.

The application of routeSampler did also not produce plausible results. Running the simulation with the set of sampled routes from the tool led to severe congestion and a breakdown of the simulation after around 15 minutes of simulation time. Congestion can be found at several intersections and results from the overestimation of routes which include turnarounds. Additionally, traffic on minor streets such as Sophienstraße at LSA 480 was severely overestimated which led to congestion in the simulation. Due to the breakdown of traffic in the simulation the comparison of turning ratios between historic counts and simulation is omitted. The reason for this is that representative vehicle counts could not be performed in the simulation since the congestion prevented vehicles from passing the test detectors.

In conclusion, it can be said that given the limited amount of data it was not possible to create a plausible simulation with flowrouter or routeSampler. However, the results of this work should not be seen as a definitive evaluation of the capabilities of the two tools since the unsatisfactory results are mainly related to the poor data basis. In principle, the study area would be a suitable test network to evaluate the tools since most intersections are equipped with induction loop detectors. Because of that it is recommended to monitor the data platform of the city of Munich to check when the currently erroneous detectors report counting data again. Then a more in-depth analysis of flowrouter and routeSampler could be performed in a future study with a more extensive data basis. Alternatively, both tools could be tested in a different study area with more available counting data. A main research question for this study would be how both tools deal with the fact that measurements are mostly only available at the inflows of intersections. This could show whether it is possible to create plausible traffic flows at intersections as this is a main shortcoming of the results from this simulation given the limited amount of counting data.

Data availability statement

The underlying SUMO networks originate from freely accessible and usable OpenStreetMap data extracts. The induction loop record extracts and signal plans used in this study are coming from the City of Munich and are currently (April 2023) not freely-accessible. Historical induction loop records (together with their exact locations) are available as part of the UTD19 data set (https://www.research-collection.ethz.ch/handle/20.500.11850/437802) for Munich and other European cities [3].

Author contributions

All authors have contributed equally.

Competing interests

The authors declare that they have no competing interests.

Acknowledgement

The presented work is part of the accompanying research for the city of Munich, namely the “GeodatenService”, within the project Digital Twin Munich, DZ-M (“Digitaler Zwilling München”). For more information please visit: https://muenchen.digital/twin/ .

References

1. P. A. Lopez et al., "Microscopic traffic simulation using SUMO", Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC), pp. 2575-2582, 2018, doi: https://www.doi.org/10.1109/ITSC.2018.8569938.
2. D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO-Simulation of Urban MObility," International journal on advances in systems and measurements, vol. 5, no. 3&4, 2012.
3. A. Loder, L. Ambühl, M. Menéndez and K. W. Axhausen, "Understanding traffic capacity of urban networks", Scientific Reports, vol. 9, no. 1, pp. 1-10, 2019, doi: https://www.doi.org/10.1038/s41598-019-51539-5.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.222

Published: 29 June 2023

Analysis and modelling of road traffic using SUMO to optimize the arrival time of emergency vehicles

Soni and Weronek

Frankfurt University of Applied Sciences, Frankfurt am Main, Germany

Abstract: Traffic simulation tools are used by city planners and traffic professionals over the years for modelling and analysis of existing and future infrastructural or policy implementations. There are numerous studies on emergency vehicle (EV) prioritization in cities all over the world, but every area is unique and requires the data collection and simulation to be done separately. In this case, the focus area is the Mörfelder Landstraße in Frankfurt am Main, Germany, one of the busiest streets in this city. The study illustrates demand modelling, simulation and evaluation of a traffic improvement strategy for EVs. Vehicular traffic such as passenger cars and trams are simulated microscopically. To perform accurate traffic simulation, input data quality assurance and cleansing of Master Data is required. Therefore, the data is adapted to reproduce the real-world scenario and transformed into the readable format for the simulation model. Vehicular demand is calibrated by traffic count data provided by the Frankfurt Traffic Department. To model road traffic and road network, origin destination matrices using the Gravity Mathematical Model and Open Street Maps are generated, respectively. This process is time-consuming and requires effort. However, this process is critical to get realistic results. In the next step, the road traffic is simulated using SUMO (Simulation of Urban mobility). Finally, EV relevant key performance indicators (KPIs): total trip time and total delay time are derived from simulations. The real-world scenario is compared with five alternative scenarios. The comparison of the KPIs revealed that the real-world scenario results in longer travel times compared to the EV-prioritization scenario. In the least case, the overall travel times for EV has decreased significantly and, as we know, in the case of EVs, even a few seconds saved could prove crucial for a person in need.

Keywords: Demand Modelling, Origin Destination Matrices, Simulation, Emergency Vehicles, Traffic Improvement Strategy

1 Introduction

In the 21st century, high rate of urbanisation and the advancement in the transport sector has led to an increase in urban vehicular mobility. This resulted in people opting for a comfortable and luxurious life. But on the other hand, it has also negatively impacted the quality of life by increasing the potential for traffic problems such as traffic congestion, accidents, environmental issues for example, increase in greenhouse gases, carbon emission, particulate matter etc. To combat these problems traffic improvement strategies such as car pool lanes, public transport bus lanes, dedicated space for cyclists and pedestrians, to name some, are adopted. Testing and implementation of such strategies require prior investigation and analysis. Without these studies, the implemented strategies or policies could be unreliable and might end up costing even more in terms of infrastructure, time and in some cases even human life. To have a theoretical evaluation and predict the outcome of these strategies, traffic simulation plays a vital role.

For traffic simulation to be implemented properly numerous elements are needed but the following are the most important ones [1]:

- Network data such as roads, footpaths, tram routes
- Additional traffic infrastructure such as traffic lights, induction loops
- Traffic demand
- Traffic constraints e.g. speed limits, construction sites, bus lanes.

It is time consuming and requires effort to prepare a traffic simulation model using these elements. Therefore, many simulation tools provide ready to use simulation models so that the user can directly test their traffic improvement strategies and saves time and effort required for simulation [2].

One of the main motive of traffic simulation is to evaluate different traffic improvement strategies. This study shows another traffic improvement strategy based on emergency vehicles. *“An emergency vehicle is a vehicle that is used by emergency services to respond to an incident”* [3]. Even a small reduction in the arrival time of EVs (fire brigade, ambulance or police) can save lives of the people who need immediate assistance. To tackle such situations EVs have special rights such as violating red lights when approaching a traffic light junction (TLJ) or traveling in the opposite direction to reduce the arrival time. But this approach is not a full proof approach to optimize the arrival time. As, there are times when EVs are stuck in a long queue of vehicles in front of the TLJ or are stuck in a traffic congestion where there is no way to overtake.

The main objective of the study is to simulate the road traffic of the Mörfelder Landstraße in the Sachsenhausen area, Frankfurt am Main, Germany, followed by studying and evaluating different scenarios to optimise the arrival time of emergency vehicle which could help in combating the aforementioned situations.

This paper is structured as follows: Section 2 discusses in details about the master data, demand modelling and simulation process by elaborating on data pre-processing, network modification and traffic generation. Section 3 explains solution methodology, different case scenarios for EVs. Section 4 shows the result obtained from the case scenarios. Section 5 presents the conclusion and future work.

2 Master Data, Demand Modelling and Simulation

The data flow diagram based on Gane-Sarson methodology is shown in Figure 1. Master Data consists of the road network (supplemented with additional infrastructure and traffic constraints) and the aggregated vehicle count for 24 hours. The vehicular counts are provided in the form of shape file for the geographical location of the Sachsenhausen area in Frankfurt am Main and the road network is imported from Open Street Map [4].

A methodology named as Gravity Model [5] is used for calculating Origin Destination Matrices (ODMs). It is based on the principle of gravitation theory of Newtonian physics. With reference to the traffic planning, the Gravity Model theory states in [5] that: ”the number of trips between two Traffic Assignment Zones (TAZ) will be directly

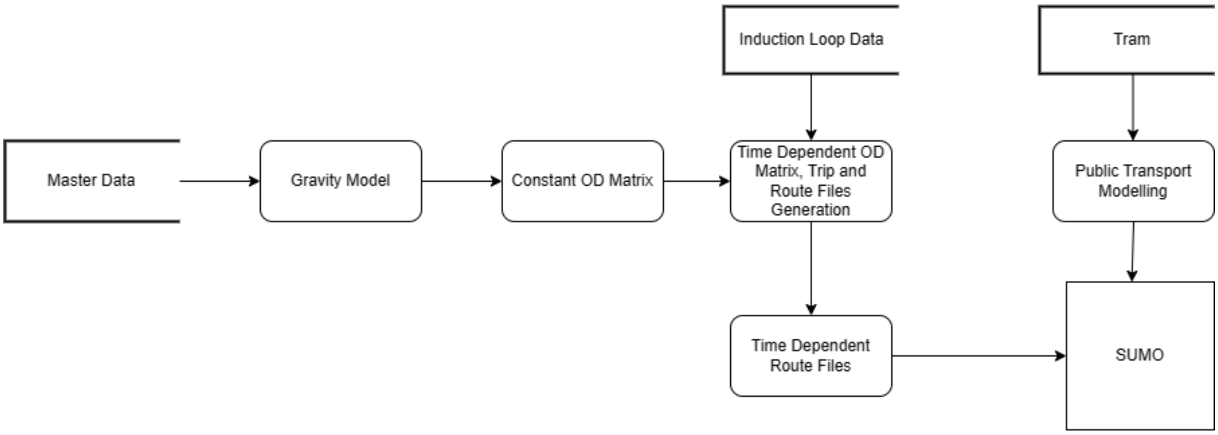

Induction Loop Data
Tram
Master Data
Gravity Model
Constant OD Matrix
Time Dependent OD Matrix, Trip and Route Files Generation
Public Transport Modelling
Time Dependent Route Files
SUMO

Figure 1. Data Flow Diagram, whereas Master Data comprises of additional infrastructure and traffic constraints

proportional to the number of productions in the production zone and attractions in the attraction zone. In addition, the number of interchanges will be inversely proportional to the spatial separation between the zones."

Mathematically, the Gravity Model is defined as [5]:

$$T_{ij} = P_i \left[\frac{A_j F_{ij} K_{ij}}{\sum_{k=1}^{n} A_k F_{ik} K_{ik}} \right], \tag{1}$$

with T_{ij}: number of trips from zone i to zone j, P_i: number of trips produced by zone i, A_j: number of trips attracted by zone j, F_{ij}: friction factor relating the spatial separation between zone i and zone j, K_{ij}: optional trip-distribution adjustment factor for interchanges between zone i and zone j, n: the number of zones.

The initial values of P_i and A_j are considered from the vehicular counts provided in the form of a shape file. The friction factor and trip distribution adjustment factor are not considered in this study as the only available data is traffic counts. Therefore, equation mentioned below is used for calculating the trip distribution:

$$T_{ij} = P_i \left[\frac{A_j}{\sum_{k=1}^{n} A_k} \right]. \tag{2}$$

Before applying this methodology, there are two assumptions made regarding the road network: First, the number of cars occupying the parking space and freeing the parking space are equal as in reality the difference is negligible compared to the normal traffic. Therefore it is not taken into consideration. The second assumption is that there is no generation or elimination of cars within the TAZ (conservative network). Additionally, the total number of cars generated at the entry points of the TAZ should be equal to the total number of cars eliminated at the destination points of the TAZ. This is known as "the closing condition at the edge" [6], also shown in equation 3:

$$\sum_{i=1}^{n} P_i = \sum_{j=1}^{n} A_j, \tag{3}$$

with P_i: number of trips produced by zone i, A_j: number of trips attracted by zone j, n: the number of zones [6].

If this closing condition is not met, which is shown in equation 3 then the balancing process is performed using equations 4 and 5. This process is adopted from [5] and is divided into two steps. Firstly, the balancing factor is calculated using equation 4. Secondly, the number of trips attracted by each zone is multiplied by this balancing factor calculated in step 1 to attain balanced number of trips attracted by each zones, shown in equation 5 and this leads to the fulfillment of equation 3:

$$Factor = \frac{\sum_{i=1}^{n} P_i}{\sum_{j=1}^{n} A_j}, \tag{4}$$

with $Factor$: balancing factor, P_i: number of trips produced by zone i, A_j: number of trips attracted by zone j and

$$A_j^{'} = Factor * A_j, \tag{5}$$

with $A_j^{'}$: balanced number of trips attracted by zone j.

Once the closing condition is met, the trip distribution matrix is generated using equation 2. The matrix balancing approach [6],[5] is carried out to ensure that the expected number of trips produced is equal to the calculated number of trips produced for all the zones. Similarly, the expected number of trips attracted is equal to the calculated number of trips attracted for all the zones. This is shown in equation 6 and 7. This is an iterative process, and it iterates until the calculated production and attraction is equal to the expected production and attraction i.e. $Factor_{Aj}$ and $Factor_{Pi}$ converges to 1. This process is implemented using a python script:

$$Factor_{Aj} = \frac{Given_{Aj}}{Total_{Aj}}, Factor_{Pi} = \frac{Given_{Pi}}{Total_{Pi}}, \tag{6}$$

with $Given_{Aj}$: expected number of trips attracted by zone j, $Total_{Aj}$: calculated number of trips attracted by zone j, $Given_{Pi}$: expected number of trips produced by zone i, $Total_{Pi}$: calculated number of trips produced by zone i and

$$D_{ij}^{'} = Factor_{Aj} * Factor_{Pi} * D_{ij}, \tag{7}$$

with D_{ij}: trip interchange calculated for each entry/exit zone.

Due to the numerical reasons, equation 6 and 7 do not converges to 1. To solve this issue, a heuristic approach is used where the study area is divided into 3 parts. This leads to the creation of 3 constant ODM. Hence section based demand modelling is performed. The study area for demand modelling is Mörfelder Landstraße. This stretch is around 3.3 km long, also highlighted in the Figure 2. A total of 21 entry/exit zones are present in the study area marked in red in Figure 2.

The calculated constant ODMs consist of aggregated count for 24 hours. Then the distribution of the counts over the period of 24 hours is done with the help of induction loop data. This data contains counts from June 2020 till March 2021 and each of this count is split with the time interval of 15 minutes starting from 00:00 until 23:57. With the combination of induction loop data and SUMO functionalities such as od2trips and duarouter, time dependent ODMs based route files are created. This acts as the input to SUMO to simulate the road traffic. In addition to the simulation of passenger cars, trams are also modelled with safety traffic lights at the tram stops. They are simulated using public transport model provided by SUMO. The frequency for the trams are set to every 10 minutes.

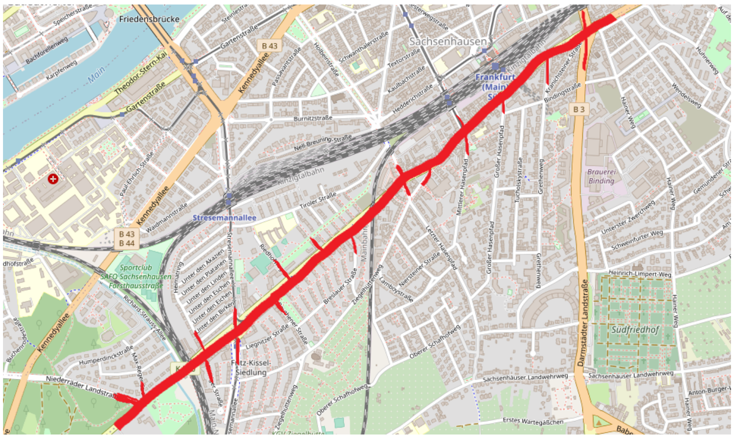


Figure 2. Study Area - Demand Modelling

3 Solution Methodology, Case Scenarios and Study Area

3.1 Solution Methodology

There are many studies carried out to optimize the arrival time of EVs such as optimization in routing and dispatching of EV which can led to faster routes for EV [7], ranking of alternatives for emergency routing [8]. However, behaviour of pedestrians, especially children is unpredictable, and even though SUMO can be used to model such patterns, but in the real world it does not function exactly in the simulation. In the case of re-routing an EV, the algorithm prioritizes the shortest route which is free of traffic. But the shortest route could include residential areas that consist of more foot traffic as compared to main streets. Thus, the preferred approach in this study is EV prioritization approach using V2X (Vehicle to Infrastructure) communication with TLJ. This approach is adopted from [9],[10],[11]. The basic approach is that as soon as the EV arrives at TLJ, traffic light is switched to green for the direction of EV trip and prioritizes the EV[9],[10],[11].

The following steps are performed for the EV prioritization application which is also known as the WALABI approach[9]:

- EV sends CAMs (Cooperative Awareness Messages) and route information
- Road side unit informs Traffic Management Center (TMC)
- TMC sets traffic lights on the route of the EV: green for the EV and red for all other traffic participants
- After the EV has passed the intersection normal operation continues.

For the aforementioned EV prioritization approach, the question arises what should be the optimal distance between an EV and the traffic light so that the traffic light should turn green. The study [10] shows that the EV is usually within the range of 300 meters from the TLJ and when EV enters this range, the traffic light is turned to green and when EV passes the TLJ the traffic light switches back to normal. Therefore 300 meters is considered as a threshold distance value for scenario 2 which is discussed in section 3.3.

There is a negative consequence of having this predefined value that is for the other vehicles who are waiting in front of the red signal. If the red phase on the traffic light increases then traffic congestion on the other side may also increase leading to more chaos and more time to diffuse the traffic congestion. Therefore to solve this issue, instead of taking a predefined value, it is calculated dynamically (dynamically calculating threshold distance). This threshold distance is calculated using the speed of the EV and the number of vehicles waiting in front of TLJ shown in equation (8) and (9). This approach is adopted from the study in [9]:

$$T_{free} = (N_{waiting} + 1) * t_B + t_{safety}, \tag{8}$$

with T_{free}: time which is needed to let the EV pass the traffic light, $N_{waiting}$: number of vehicles waiting in front of TLJ, t_{safety}: safety time which is 3 seconds, t_B: time required for one vehicle to pass the intersection which is 1.8 sec and

$$d = T_{free} * V_{EV}, \tag{9}$$

with d: distance of the EV to the intersection, V_{EV}: speed of the EV.

3.2 Emergency Vehicle Prioritization Study Area

The highlighted path shown in the Figure 3 is the route of EVs whose behaviour is evaluated in the simulations. The route length is approximately 1.5 km consisting of 3 major and 2 minor junctions of the Mörfelder Landstraße which are mentioned in Table 1.

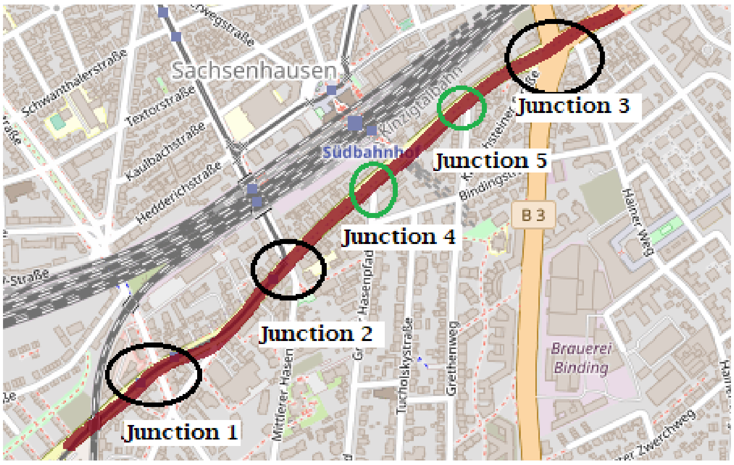


Figure 3. EV Study Area [4]

Table 1. Junction Details

Junction ID	Junction Name
Junction 1	Oppenheimer Landstraße and Mörfelder Landstraße junction
Junction 2	Schweizer Straße and Mörfelder Landstraße junction
Junction 3	Darmstädter Landstraße and Mörfelder Landstraße junction
Junction 4	Großer Hasenpfad and Mörfelder Landstraße junction
Junction 5	Grethenweg and Mörfelder Landstraße junction

3.3 Case Sceanrio

For each of the three scenarios which are considered for studying the behaviour for EVs, there are two cases considered. One is the usual traffic condition and other is the closed lane based on the assumption that only one lane stays available and all others are closed due to construction/incident reasons or by prioritizing these lanes for non car traffic. Hence making up a total of six scenarios.

1. Scenario 1: No-Priority for EVs i.e EV runs with their special rights such as violating red lights - Usual Traffic Condition (all available lanes are open)
2. Scenario 2: EV prioritization where prioritization starts at a pre defined distance i.e. 300 meters - Usual Traffic Condition (all available lanes are open)
3. Scenario 3: EV prioritization where prioritization starts at a dynamically calculating distance at run-time - Usual Traffic Condition (all available lanes are open)
4. Scenario 4: No-Priority for EVs i.e EV runs with their special rights such as violating red lights - Closed Lane (one or more closed lane present in the route of EV)
5. Scenario 5: EV prioritization where prioritization starts at a pre defined distance i.e. 300 meters - Closed Lane (one or more closed lane present in the route of EV)
6. Scenario 6: EV prioritization where prioritization starts at a dynamically calculating distance at run-time Closed Lane (one or more closed lane present in the route of EV).

In this study area, around 60% of the street has more than 1 lane. Figure 4 shows the setup of closed lanes where edges highlighted in red colour signifies that lanes are closed.

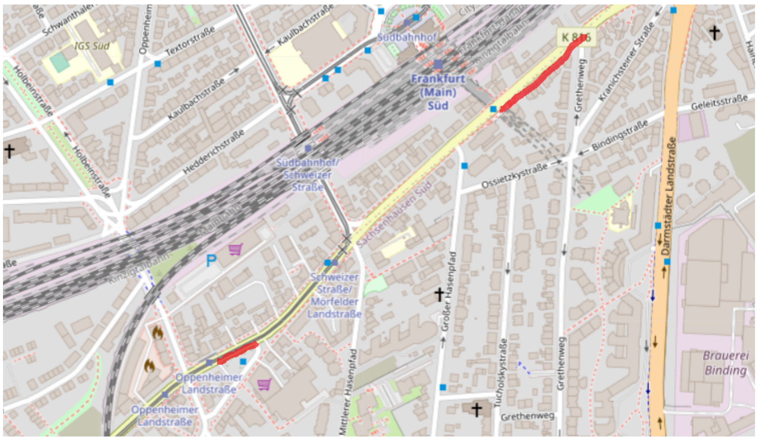


Figure 4. Closed Lane Setup [4]

To generate traffic in a realistic manner, induction loop data is used. This data of induction loops is cleaned, averaged out and normalised over the total number of cars which resulted in creation of traffic flow distribution over the course of the day. It is shown in Figure 5. The X axis represents the time of timeslice [hh:mm] and the Y axis represents the average rate normalised for the overall traffic per day. The maximum averaged, measured count per 3 minutes is observed around 8 am, which is 30 cars. It can be seen in the Figure 5 that the congestion in the morning from 7:00 am until 10:30 is the most on the street of the Mörfelder Landstraße and therefore that is the time range selected for testing the EV. A total of 10 EVs are run between this time range and their trip time and delay time are compared.

4 Results

This section explains the simulation results obtained for case scenarios discussed above. A total of 10 EVs (ambulances) are run. The departure time for each of these EV are 8:21, 8:36, 8:53, 9:06, 9:21, 9:35, 9:51, 10:06, 10:21 and 10:36 a.m. respec-

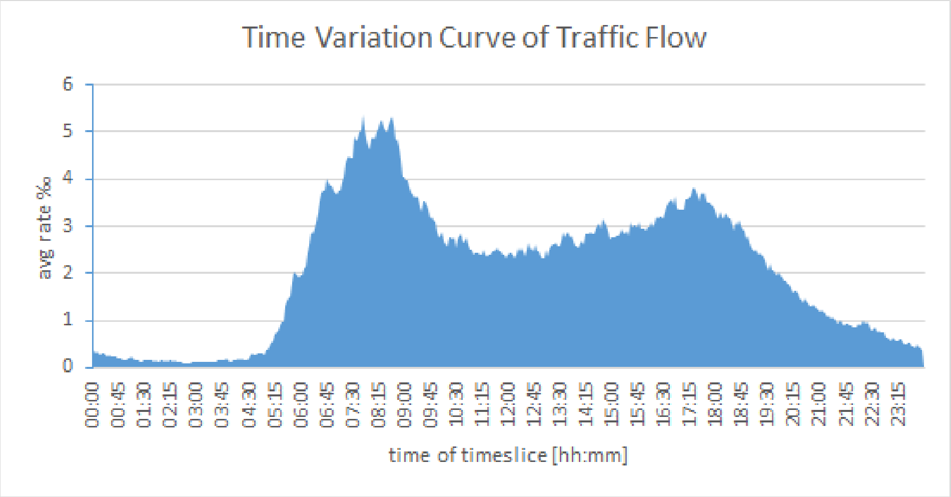


Figure 5. Traffic flow split of the 24 hour count for the Mörfelder Landstraße with timeslices of 3 minutes intervals

tively. The KPIs that have been considered are the total trip time (time required for the vehicle to finish the trip) and total delay time (time for which the vehicle travels below the ideal speed). For EVs, the speed is set 50% above the speed limit of the edge specified by the attribute ”speed factor” which is defined as 1.5 while configuring the EV in SUMO. This is adopted from the study [12].

4.1 Emergency Vehicle Behaviour - Normal Traffic Condition

Table 2 and 3 show the comparison of total trip time and total delay time for each of the EVs, where ”EV with No-Priority (Normal Traffic Condition)” scenario acts as the baseline reference for calculating the impact. For scenario 1, the trip time varies between 238 and 439 seconds. The average for scenario 1 is 315 seconds and empirical variance is 60.5 which is 19% of the average. This variance is almost the same for all other scenarios (20±3%). The first reason for this variance are the different traffic conditions such as traffic density. However, there are some specific events that have major impact on the trip time. In some simulations when a tram stops, the subsequent red traffic light led to a delay since the EV is not able to overtake the tram. This is also reflected in total delay time in Table 3 for e.g. ambulance with ID 6_Ambulance. For scenarios 2 and 3 the average trip time is 153 and 162 seconds respectively and average delay time is 82 and 91 seconds respectively.

Table 2. Normal Traffic Condition - Total Trip Time

Total Trip Time	No	With	With Dynamic
(seconds)	Priority	Priority	Priority
	Scenario 1	Scenario 2	Scenario 3
1_Ambulance	326	202(-38%)	210(-36%)
2_Ambulance	374	153(-59%)	122(-67%)
3_Ambulance	297	193(-35%)	216(-27%)
4_Ambulance	293	153(-48%)	146(-50%)
5_Ambulance	260	149(-43%)	163(-37%)
6_Ambulance	439	118(-73%)	141(-68%)
7_Ambulance	342	157(-54%)	181(-47%)
8_Ambulance	253	135(-47%)	131(-48%)
9_Ambulance	328	146(-55%)	155(-53%)
10_Ambulance	238	127(-47%)	152(-36%)

Table 3. Normal Traffic Condition - Total Delay Time

Total Delay Time	No	With	With Dynamic
(seconds)	Priority	Priority	Priority
	Scenario 1	Scenario 2	Scenario 3
1_Ambulance	255	131(-49%)	139(-46%)
2_Ambulance	303	82(-73%)	52(-83%)
3_Ambulance	226	123(-46%)	145(-36%)
4_Ambulance	222	82(-63%)	75(-66%)
5_Ambulance	195	78(-60%)	93(-53%)
6_Ambulance	368	48(-87%)	71(-81%)
7_Ambulance	272	86(-68%)	110(-59%)
8_Ambulance	182	64(-65%)	60(-67%)
9_Ambulance	258	75(-71%)	84(-67%)
10_Ambulance	167	56(-66%)	81(-52%)

4.2 Emergency Vehicle Behaviour - Closed Lane Scenario

Table 4 and 5 show the comparison of total trip time and total delay time for each of the EVs (Closed Lane), where ”EV with No-Priority (Closed Lane)” scenario acts as the baseline reference for calculating the impact. For scenario 4, the trip time varies between 242 and 469 seconds. The average for scenario 4 is 344 seconds and empirical variance is 70.2 which is 20% of the average. The variances of the scenarios 4, 5 and 6 are almost the same as scenarios 1, 2 and 3 which is (20±3%). The reasons for the variances are the same like in section 4.1 but the occurrences of these special events happened in different time intervals. This is also reflected in total delay time in Table 5 for e.g. ambulance with ID 2_Ambulance. For scenarios 5 and 6 the average trip time is 183 and 191 seconds respectively and the average delay time is 112 and 117 seconds respectively.

Table 4. Closed Lane - Total Trip Time

Total Trip Time (seconds)	No Priority Scenario 4	With Priority Scenario 5	With Dynamic Priority Scenario 6
1_Ambulance	291	268(-8%)	225(-23%)
2_Ambulance	467	159(-66%)	182(-61%)
3_Ambulance	397	214(-46%)	230(-42%)
4_Ambulance	302	164(-46%)	178(-41%)
5_Ambulance	375	209(-44%)	238(-37%)
6_Ambulance	258	169(-34%)	190(-26%)
7_Ambulance	349	197(-44%)	206(-41%)
8_Ambulance	356	133(-63%)	153(-57%)
9_Ambulance	399	178(-55%)	146(-63%)
10_Ambulance	242	140(-42%)	157(-35%)

Table 5. Closed Lane - Total Delay Time

Total Delay Time (seconds)	No Priority Scenario 4	With Priority Scenario 5	With Dynamic Priority Scenario 6
1_Ambulance	220	197(-11%)	154(-30%)
2_Ambulance	396	88(-78%)	111(-72%)
3_Ambulance	326	144(-56%)	160(-51%)
4_Ambulance	231	93(-60%)	107(-54%)
5_Ambulance	305	138(-55%)	167(-45%)
6_Ambulance	187	98(-48%)	120(-36%)
7_Ambulance	278	126(-55%)	108(-61%)
8_Ambulance	285	63(-78%)	82(-71%)
9_Ambulance	328	107(-67%)	75(-77%)
10_Ambulance	171	69(-59%)	86(-50%)

4.3 Threshold Distance

In Scenario 2 and 5, the threshold distance is constant i.e. 300 meters. In contrast for scenario 3 and 6, the threshold distance is calculated using equation 8 and 9. Table 6 and 7 show this distance for all major junctions. The variance of these distances is due to the change in the number of vehicles waiting in front of the TLJs and the speed of the ambulance when entering the study area. The velocity used in these equations are derived from initial calculated speed of the ambulances after entering the study area. It ranges between 36 and 55 km/h.

Table 6. Normal Traffic - Dynamic Distance

Dynamic Distance (Normal Traffic)	Junction 1	Junction 2	Junction 3	Junction 4	Junction 5
1_Ambulance	263	201	97	222	118
2_Ambulance	87	253	182	111	134
3_Ambulance	139	230	139	121	67
4_Ambulance	211	156	432	101	239
5_Ambulance	334	236	358	138	65
6_Ambulance	394	179	340	179	72
7_Ambulance	397	261	451	126	72
8_Ambulance	196	60	60	105	60
9_Ambulance	276	250	457	146	69
10_Ambulance	83	218	240	105	60

Table 7. Closed Lane - Dynamic Distance

Dynamic Distance (Closed Lane)	Junction 1	Junction 2	Junction 3	Junction 4	Junction 5
1_Ambulance	261	119	47	47	47
2_Ambulance	312	232	526	152	178
3_Ambulance	220	91	188	91	91
4_Ambulance	375	101	348	101	320
5_Ambulance	231	183	207	111	64
6_Ambulance	369	142	142	67	92
7_Ambulance	397	126	208	181	72
8_Ambulance	118	65	100	47	136
9_Ambulance	258	71	205	71	98
10_Ambulance	117	179	283	76	76

4.4 Aggregated Results

Table 8 shows the average impact for EVs under ”Normal Traffic” condition where the number in parenthesis gives the average of the absolute impact and the percentage gives the average of the relative impact compared to the baseline reference. The scenario ”EV with No-Priority” is the baseline reference. Table 9 shows the average impact for EVs with ”Closed Lane” condition. Here, the scenario ”EV with No-Priority (Closed Lane)/Scenario 4” is the baseline instead of ”EV with No-Priority/Scenario 1”. Moreover, Table 10 ”Baseline Comparison” shows the average increment in the travel time and delay time when the lanes are closed.

Table 8. Normal Traffic Average Impact

Normal Traffic	Baseline Reference Scenario 1	With Priority Scenario 2	With Dynamic Priority Scenario 3
trip time	315s	-51%(-162s)	-49%(-153s)
delay time	245s	-66%(-162s)	-63%(-154s)

Table 9. Closed Lane Average Impact

Closed Lane	Baseline Reference Scenario 4	With Priority Scenario 5	With Dynamic Priority Scenario 6
trip time	344s	-47%(-161s)	-45%(-153s)
delay time	273s	-59%(-160s)	-57%(-156s)

Table 10. Baseline Comparison

Base Line Reference	Normal Traffic Scenario 1	Closed Lane Scenario 4	Normal Traffic vs Closed Lane
trip time	315s	344s	+9%
delay time	245s	273s	+11%

5 Conclusion and Future Work

5.1 Conclusion

The optimization process used in this study involved data pre-processing. This includes improvement of master data quality which required network modelling and the creation of ODMs to make the models as realistic as possible. During the process of importing networks from OSM, the imported network contained a lot of errors due to the misalignment with reality such as errors in simple road links (lanes wrongly connected), classification of lanes etc. Therefore, network corrections were done using SUMO (SUMO's editing tool NETEDIT). ODMs were created by leveraging tools such as Python and Excel. These processes were time consuming but at the same time it was important for the execution of the models.

The simulation results in Table 8 and 9 show that the implementation of EV prioritization techniques results in a significant improvement of the KPI values. For ”Normal Traffic” condition, the average trip time and delay time is dropped by 51% and 49%, 66% and 63% respectively. For the ”Closed Lane” condition, increases in travel time and delay time was anticipated but the impact is lower than expected. The reason maybe that only 33% of the overall multi lanes were reduced to one lane. However, the average trip time and delay time is also dropped by 47% and 45%, 59% and 57% respectively. The maximum impact were seen on the scenarios where the tram stops ahead of the ambulance and the subsequent traffic light is switched to green. The model where threshold distance is calculated dynamically is not as good as expected. The reason is that the calculated distance is mostly lower than 300 meters for all major junctions which reduces the optimization of the travel time of the ambulances. Nevertheless, in all cases the travel time was reduced with the intervention into the traffic infrastructure. Therefore, it can be concluded that through the EV prioritization approaches using V2X communication, EVs can save precious seconds which could be the difference between life and death for a person in need.

5.2 Future Work

In future work, the impact of the length of the closed lanes on the arrival times of the EV should be investigated. Another interesting addition to the simulation would be to include foot traffic (pedestrians), buses and cyclists. The current model is used to study only one EV at a given instant during the simulation. Therefore, further studies could be implemented to handle multiple EVs at the same time. As SUMO is a continuously improving software and thus, for this model, there is still scope of improvement for lane changing functionalities e.g. overtake using the opposite lane. The traffic light control plans used in the study are edited as per demand model. Further work can be carried out to incorporate real world traffic control plans that could lead to even more accurate depiction of the real-world scenario. Since ”Dynamic Priority” scenario calculates the threshold distance often less than 300m, delivering the results in the section 4, the parameters in the ”Dynamic Priority” strategy needs to be optimized. Finally, this simulation needs to be redone with higher, post pandemic traffic rates.

Acknowledgement

All the relevant traffic data have been supplied by Straßenverkehrsamt of the city Frankfurt/Main whose support with any data related issues in compiling this research is gratefully acknowledged.

References

[1] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, “Microscopic traffic simulation using sumo,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 2575–2582.

[2] L. Bieker, D. Krajzewicz, A. P. Morra, C. Michelacci, and F. Cartolano, “Traffic simulation for all: A real world traffic scenario from the city of bologna,” in *SUMO 2014*, May 2014. [Online]. Available: https://elib.dlr.de/89354/.

[3] “Emergency vehicle.” (2023), [Online]. Available: https://en.wikipedia.org/wiki/Emergency_vehicle (visited on 09/05/2021).

[4] “Using openstreetmap.” (2023), [Online]. Available: https://wiki.openstreetmap.org/wiki/Using_OpenStreetMap (visited on 01/05/2023).

[5] W. A. Martin and N. A. Mcguckin, “Nchrp report 365: Travel estimation techniques for urban planning,” *TRB, National Research Council, Washington, DC*, vol. 18, p. 21, 1998.

[6] V. Dragu and E. A. Roman, “The origin–destination matrix development,” in *MATEC Web of Conferences*, EDP Sciences, vol. 290, 2019, p. 06 010.

[7] A. Haghani, H. Hu, and Q. Tian, “An optimization model for real-time emergency vehicle dispatching and routing,” in *82nd annual meeting of the Transportation Research Board, Washington, DC*, Citeseer, 2003.

[8] M. Woelki, T. Lu, and S. Ruppe, “Ranking of alternatives for emergency routing on urban road networks,” *WIT Transactions on the Built Environment*, vol. 146, pp. 591–598, 2015.

[9] L. Bieker-Walz and M. Behrisch, “Modelling green waves for emergency vehicles using connected traffic data,” in *SUMO Conference 2019*, M. Weber, L. Bieker-Walz, and M. Behrische, Eds., ser. EPiC Series in Computing, EasyChair, May 2019, pp. 1–11. [Online]. Available: https://elib.dlr.de/128822/.

[10] L. Bieker-Walz, “Cooperative traffic management for emergency vehicles in the city of bologna,” in *SUMO 2017 – Towards Simulation for Autonomous Mobility*, ser. Berichte aus dem DLR-Institut für Verkehrssystemtechnik, vol. 31, May 2017, pp. 135–141. [Online]. Available: https://elib.dlr.de/118034/.

[11] L. Bieker, “Emergency vehicle prioritization using vehicle-to-infrastructure communication,” in *Young Researchers Seminar*, vol. 2011, 2011.

[12] L. Bieker-Walz, M. Behrisch, and M. Junghans, “Analysis of the traffic behavior of emergency vehicles in a microscopic traffic simulation,” in *SUMO Conference 2018*, ser. EPiC Series in Engineering, vol. 2, May 2018, pp. 1–13. [Online]. Available: https://elib.dlr.de/120851/.

SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.226

Published: 29 June 2023

Generating and Calibrating a Microscopic Traffic Flow Simulation Network of Kyoto

First Insights from Simulating Private and Public Transport

Andreas Keler[1,2,3][https://orcid.org/0000-1111-2222-3333], Wenzhe Sun[4][https://orcid.org/ 0000-0002-7305-8671], and Jan-Dirk Schmöcker[4][https://orcid.org/0000-0003-2219-9447]

[1] JSPS International Research Fellow (Department of Urban Management, Kyoto University)

[2] Applied Geoinformatics, University of Augsburg, Germany

[3] Chair of Traffic Engineering and Control, Technical University of Munich, Germany

[4] Department of Urban Management, Kyoto University, Japan

Abstract. Microscopic traffic flow simulations as tools for enabling detailed insights on traffic efficiency and safety gained numerous popularity among transportation researchers, planners and engineers in the first to decades of the 21st century. By implementing a test bed for simulation scenarios of complex urban transportation infrastructure it is possible to inspect specific effects of introducing small infrastructural changes related to the built environment and to the introduction of advanced traffic control strategies. The possibility of reproducing present problems or the transportation services, such as the ones of public bus services is a key motivation of this work. In this research, we reproduce the road network of the city of Kyoto for observing specific travel patterns of public buses such as the bus bunching phenomena. Therefore, a selection of currently available data sets is used for calibrating a cutout of the Kyoto road network of a relatively large extent. After introducing a method for geodata extraction and conversion, we approach the calibration by introducing virtual detectors representing present inductive loops and make use of historical traffic count records. Additionally, we introduce bus routes partially contributed by volunteer mappers (OSM project). First simulation outcomes show numerous familiar (local knowledge) flow patterns.

Keywords: Simulation and Modeling, Network Modeling, Public Transportation Management, Calibration

1. Introduction

In this research, we aim to design data conversion procedures to create a digital twin (of at least the transportation-related aspects) of the city of Kyoto, Japan. Starting with gathering various data sources – static and dynamic information on infrastructural design elements, movement representations of tracked road users and sensors for providing traffic counts – we define suitable options for modeling and simulating private and public transport. The basic idea is to make use of the available traffic count information gathered in 5-minute-intervals from more than 1000 sensor location across the whole city for calibration purposes. By extracting OpenData on public transport services, we are able to model and simulate most of the present bus services and routes (and partially rail-based public transport).

We aim to first validate our microscopic traffic flow simulation with real bus trajectories and understand specific patterns of delays and congestions at selected time windows. Furthermore, our idea is to introduce method improvement procedures to not only design a data-driven conversion pipeline, but also to define a prototypical testbed for testing various traffic control strategies for different travel modes.

2. Methodological Approach – An Overview

2.1 Open Geodata Extraction and Conversion into Simulation Networks

One first complex task is the generation of a directed and routable road network, which should allow route generation by means of estimating OD-Matrices based on traffic count information coming from static sensors such as induction loops. The conversion of this network can be started with Volunteered Geographical Information (VGI) coming for example from the OpenStreetMap project [1]. Other, more detailed information might come from video data acquisitions and field observations. Our approach makes use of the netconvert tool [2] of the microscopic traffic flow simulation environment SUMO [3]. Similar to the approach of Keler, Grigoropoulos, and Mussack (2019) [4], we start with an extraction of raw OSM data by selecting a rectangle investigation area. After the conversion step into a PlainXML format, we are able to improve manually the quality of the road network representation by adjusting elements and specific details of the transportation infrastructure via the tool netedit (of SUMO). We refer to this by the blue box on the lower left side of Figure 1 by “Improving Conversion Quality”.

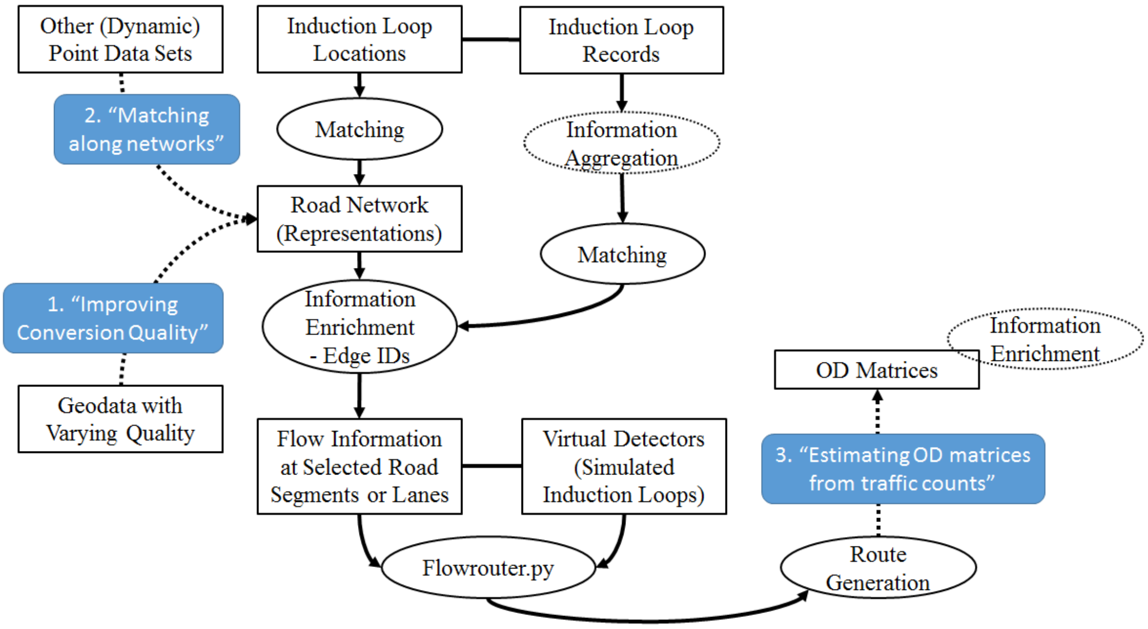

Other (Dynamic) Point Data Sets
Induction Loop Locations
Induction Loop Records
2. “Matching along networks”
Matching
Information Aggregation
Road Network (Representations)
Matching
1. “Improving Conversion Quality”
Information Enrichment - Edge IDs
Information Enrichment
OD Matrices
Geodata with Varying Quality
Flow Information at Selected Road Segments or Lanes
Virtual Detectors (Simulated Induction Loops)
3. “Estimating OD matrices from traffic counts”
Flowrouter.py
Route Generation

Figure 1. Workflow of the approach and components of improvement in blue boxes numbered from 1 to 3.

With the aim of creating a case study covering the most urban area of Kyoto and focusing on the most complex intersections in the City – we extracted a study area around the public bus route 205, as it is pictured in Figure 2.

2.2 Matching Sensor or Event Locations along Simulation Networks

After this road network representation generation, we are facing the problem of matching exact sensor locations onto lanes of the road network with respect to the allowed driving direction. Matching in the present approach for the Kyoto investigation area is conducted as Nearest-Neighbor-Search with a matching relation of 1:1, which means one point record (respective induction loop location) is matching exactly one road edge or one road lane (in this case with a specific driving direction).

Related research, which might improve the current data matching outcomes for the case of induction loop locations, includes the idea of an exact statistical method for analyzing co-location on a street network with a Japanese case study by Morioka, Okabe, Kwan, and McLafferty (2021) [5]. This fact is visualized in Figure 1 by the blue box on the upper-right side, indicating that this may highly influence the quality of the subsequent calibration and simulation results.

This problem of matching point data along a network is also partially intensively investigated by Okabe and Sugihara (2012) [6] in a practitioner book and an accompanying GIS tool named SANET (Spatial Analysis along NETworks).

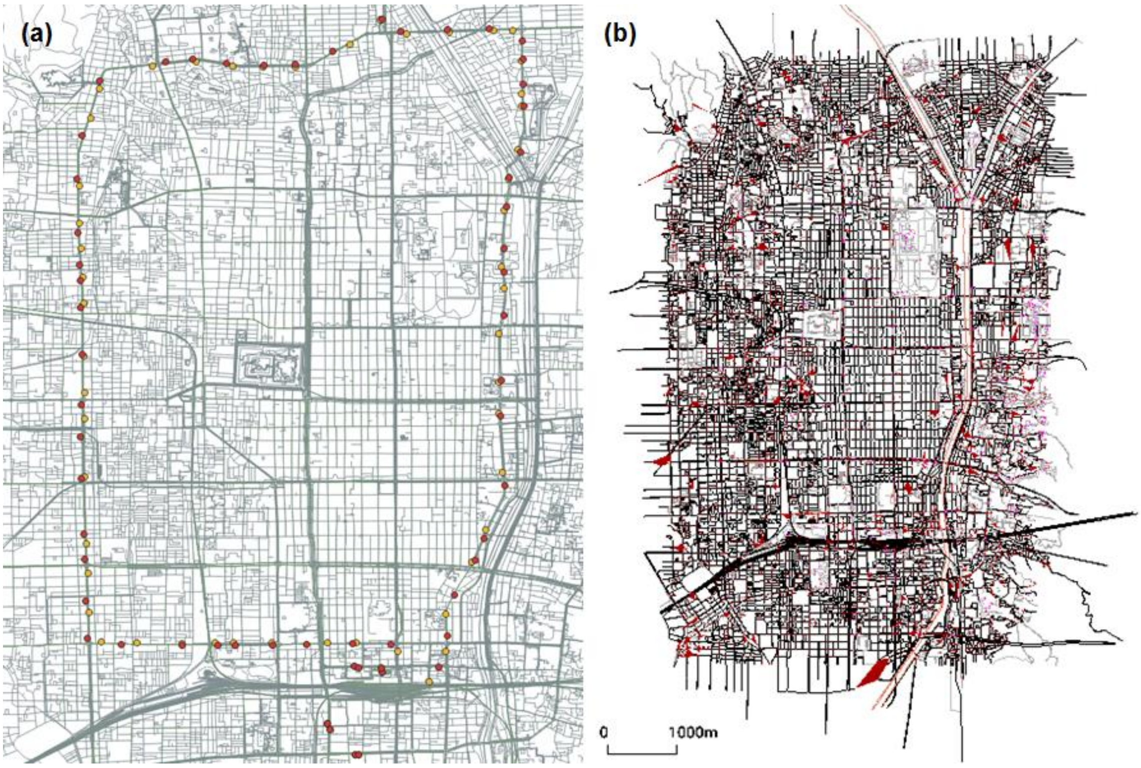


Figure 2. Locations of the bus stops of bus route 205 in Kyoto (a) and extracted and converted sumo traffic flow simulation network (b).

2.3 Calibration of the Simulation Network based on Aggregated Induction Loop Information - Estimating OD Matrices from Traffic Counts

The problem of estimating O-D demand flows using traffic counts is already very present in the literature and might be solved via numerous different approaches [7].

In our approach, we use the tool “flowrouter.py” [8] with partially reasonable results for our case study in Kyoto.

The simulation outcomes are currently being validated. Out of the 1100 installed permanent inductive loops in Kyoto, there are 451 located at the road network selection of the investigation area pictured in Figure 2. Based on selected or aggregated time windows we generate routes that comply with the observed or temporally-aggregated number for every virtual detector in our simulation runs. This means that every direction and lane that matches every of the

451 detectors is being taken into account while generating the routes of the private transport vehicles.

On the other hand, the buses are being simulated by introducing time tables and known headways for every fully- or partially-available bus route of our investigation area.

3. First Results and Novel Insights

Our first traffic flow simulation outcomes deliver novel insights of how to model road users and their compositions. As there different driving behavior with differing thresholds for car following and overtaking (compared to European conditions), deadlock situations as in Figure 3a appear at selected simulated intersections of the Kyoto simulation network. These situations might be avoided by adjusting the behavior of all simulated road users.

Other appearances, when simulating all public bus services, are similar to the bus bunching phenomena as pictured in Figure 3c. This might be a pattern of the real world worth to be validated with real bus trajectories. As Sun, Schmöcker, and Nakamura (2020) [9] state in their research this is a typical pattern for the bus service in Kyoto, implying a problem for passengers and other road users due to its relation with operational delays and with bus dwell times. In Kyoto, there is a schedule-adhere mechanism (holding the bus at the stop until the timetable time) but only when the bus is ahead of schedule. Bunching can happen when both buses are behind schedule and the operator will not hold the following bus to retrieve the headway [9,10].

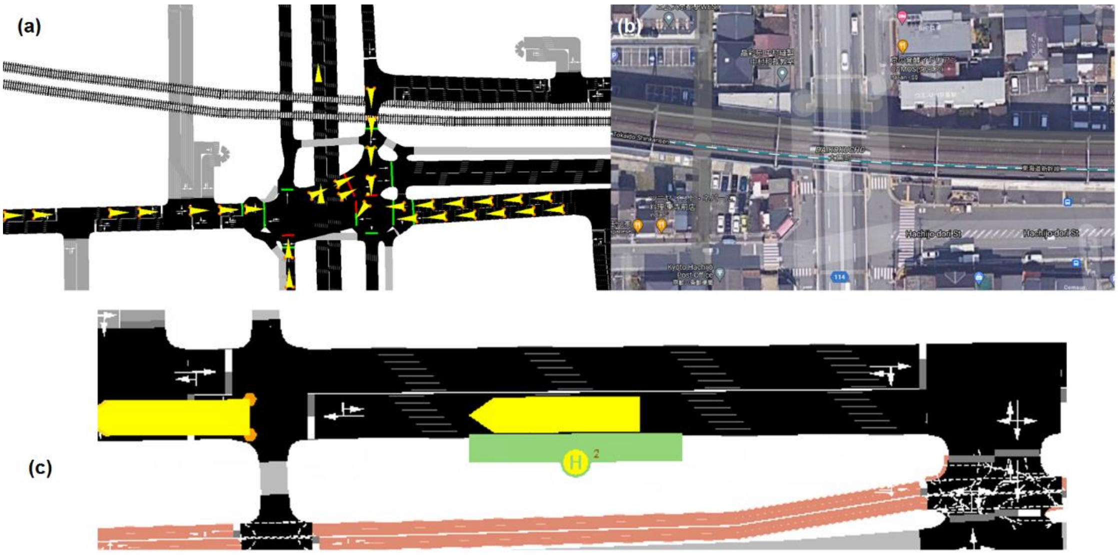


Figure 3. Selected results of the sumo simulation of Kyoto with (a) showing a deadlock situation at (b) a signalized intersection, and (c) bus bunching example.

4. Conclusion and Outlook

This work presents a framework for generating and calibrating a microscopic traffic flow simulation network of the larger scale investigation area in Kyoto as pictured in Figure 2. Several simulation outcomes comply with real world observations, mainly due to the availability of the detailed (records from 5-minute-intervals) historical traffic count data sets.

Nevertheless, several evaluation steps are required for inferring a valid simulation testbed, which can be seen as parts of the calibration procedure.

The inclusion of realistic behavior in the simulated traffic of Kyoto requires the adjustment of various model parameters. In case of the sumo applications this is related especially to the intersection model, but as well the car-following and lane-changing model. Introducing cyclists besides pedestrians would as well rely on for example adjusting the sub-lane model parameters in sumo.

Another bigger adjustment is related to the requirement to depict the signal programs at the signalized intersections in our investigation area. This would need an evaluation step for adjusting the previously estimated signal phases via applying the netconvert tool [2]. The availability of this information is currently being proven. Optionally, on-site observations would be required for a more detailed evaluation of the currently implemented programs at the respective intersections.

In an outlook to further linking additional information we can estimate selected possibilities due to the availability of additional Open Data, which might be important for estimating the demand of passenger flows. One example is pictured in Figure 4 by conducting a spatial analysis of GeoNames locations at the Kyoto investigation area.

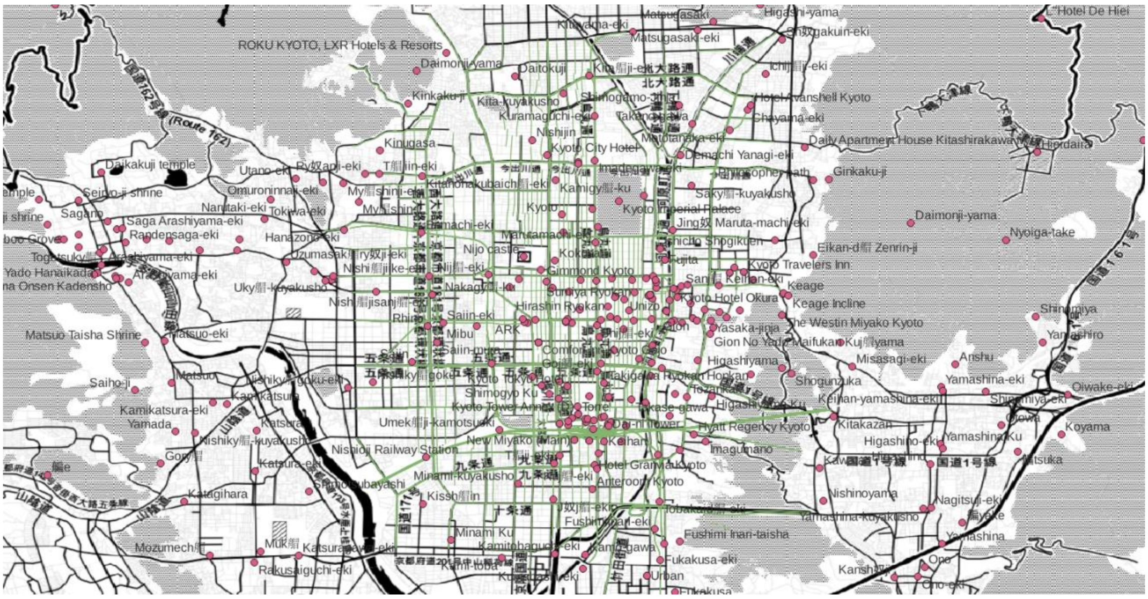


Figure 4. Spatial analysis example for estimating the amount of semantic and linked information in points of interest (POIs).

This takes also into account that intermodal trips should be defined for representing the most important times of the day of the inspected bus services – rush hours during week days with the presence of numerous tourists making use of selected bus routes for sightseeing purposes. This passenger demand is as well related to the distribution of populations in the city, which is itself related to distribution of selected building types and respective specifications. By including these insights into the present traffic flow simulation, we might estimate the numbers and spatial distributions of daily commuters and their daily travel patterns on a macroscopic level.

All in all, we can say that additional input data might highly benefit the already realistic simulation outcome of our present simulation testbed for the Kyoto case study. Additionally, the idea of a network-scale calibration for the entire City of Kyoto is an additional work in progress, which might benefit evaluating the findings discovered from our present study [10].

Data availability statement

The underlying SUMO networks originate from freely accessible and usable OpenStreetMap data extracts. The induction loop record extracts and signal plans used in this study are currently (April 2023) not freely-accessible.

The authors additionally refer to the recent (April 2023) main SUMO reference, which includes applications with the recent version of this tool: Lopez et al. (2018) [11].

Author contributions

All authors have contributed equally.

Competing interests

The authors declare that they have no competing interests.

Acknowledgement

This joint research was conducted within a research stay of the JSPS International Research Fellow program funded by Japan Society for the Promotion of Science.

The authors want to thank the 3 anonymous reviewers for their substantial reviews including specific remarks on the overall understanding of the conducted study and sharing their views on how to improve the value of this submission. These reviews have greatly benefitted the overall understanding when rewriting the manuscript.

References

1. K. Polous, J. M. Krisp, L. Meng, B. Shrestha, and J. Xiao, "OpenEventMap: A Volunteered Location-Based Service," Cartographica: The International Journal for Geographic Information and Geovisualization, vol. 50, no. 4, pp. 248-258, 2015, doi: https://www.doi.org/10.3138/cart.50.4.3130.
2. D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO-Simulation of Urban MObility," International journal on advances in systems and measurements, vol. 5, no. 3&4, 2012.
3. M. G. Armellini, O. A. Banse Bueno, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, and J. Rummel, "Brunswick simulation scenario for virtual-stops based DRT services with SUMO," in Proceedings of the 10th International Congress on Transportation Research, 2021.
4. A. Keler, G. Grigoropoulos, and D. Mussack, "Enriching complex road intersections from OSM with traffic-related behavioral information," in Proceedings of the ICA, 2019, vol. 2: Copernicus GmbH, pp. NA-NA.
5. W. Morioka, A. Okabe, M.-P. Kwan, and S. L. McLafferty, "An exact statistical method for analyzing co-location on a street network and its computational implementation," International Journal of Geographical Information Science, pp. 1-26, 2021, doi: https://www.doi.org/10.1080/13658816.2021.1976409.
6. A. Okabe and K. Sugihara, Spatial analysis along networks: statistical and computational methods. John Wiley & Sons, 2012.
7. E. Cascetta and A. A. Improta, "Estimation of Travel Demand Using Traffic Counts and Other Data Sources," in Transportation and Network Analysis: Current Trends: Miscellanea in honor of Michael Florian, M. Gendreau and P. Marcotte Eds. Boston, MA: Springer US, 2002, pp. 71-94.

8. M. Behrisch and J. Erdmann, "Route estimation based on network flow maximization," EPiC Series in Engineering, vol. 2, 2018.
9. W. Sun, J.-D. Schmöcker, and T. Nakamura, "On the tradeoff between sensitivity and specificity in bus bunching prediction," Journal of Intelligent Transportation Systems, vol. 25, pp. 1-17, 02/13 2020, doi: https://www.doi.org/10.1080/15472450.2020.1725887.
10. Q.-L. Lu, W. Sun, J. Dai, J.-D. Schmöcker and C. Antonio, “An MFD-Based Optimization Approach to Improve Transportation System Resilience under Infrastructure Disruption,” 25th Euro Working Group on Transportation Meeting (EWGT (accepted; to be published in: Transportation Research Proceedings).
11. P. A. Lopez et al., "Microscopic traffic simulation using SUMO", Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC), pp. 2575-2582, 2018, doi: https://www.doi.org/10.1109/ITSC.2018.8569938.