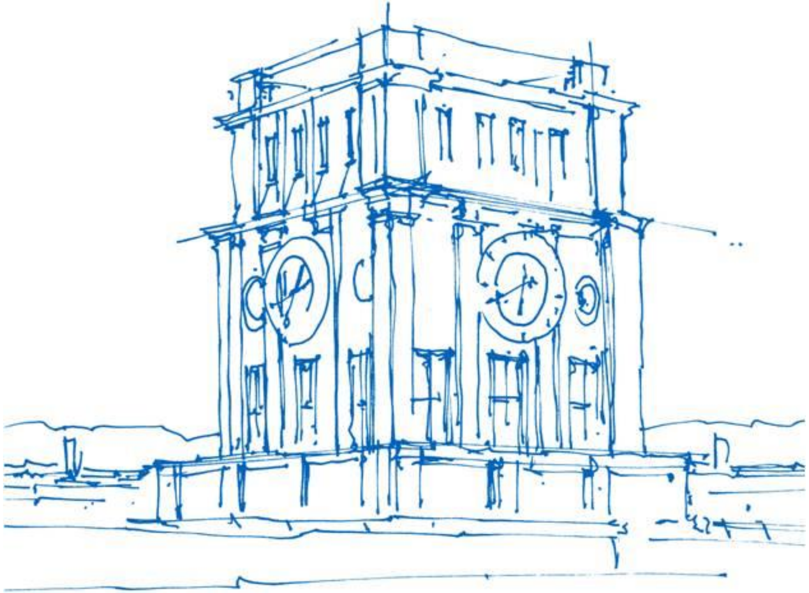
Calibration of Microscopic Traffic Simulation in an Urban Environment Using GPS-Data

SUMO User Conference 2024

Christopher Stang, Klaus Bogenberger

Agenda

1. Introduction
2. Network
3. Methodology
4. Results
5. Conclusion & Outlook



Uhrenturm der TUM

Introduction

Introduction

Accurate traffic models are important for well-founded traffic engineering: Usage of traffic count and speed measurements of road segments common approach for the calibration of traffic simulation models.

SUMO offers the tools **flowrouter** and **routesampler** for generating traffic demand models on the base of traffic count measurements.

Following approach applies a two-step optimization process by using collected GPS-data with information about vehicle count and speed measurements.

A priori optimization

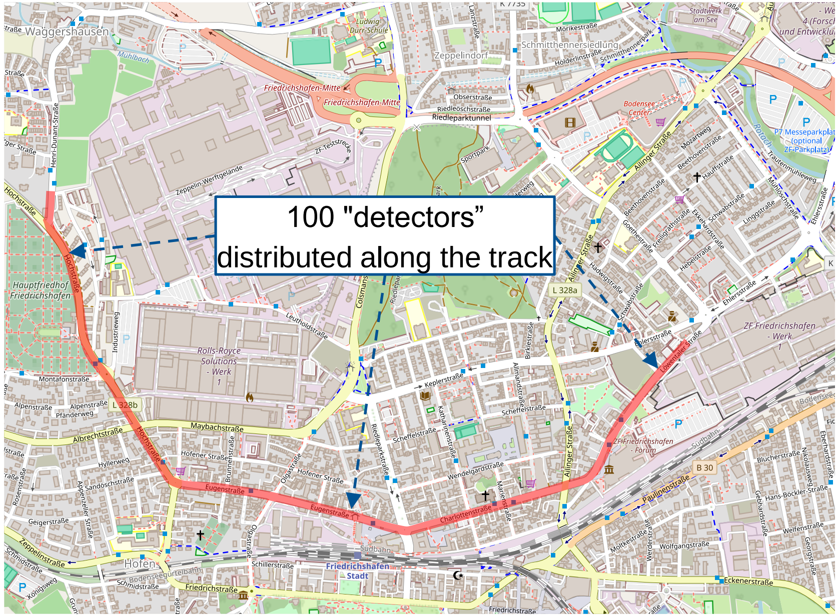
of count measurements by adopting
Integer Linear Programming

A posteriori optimization

of speed (and count) measurements by adopting
Integer Linear Programming+Evolutionary Algorithm

Network

Network



Source: OpenStreetMap. [Map section Friedrichshafen][Map]. Map data from OpenStreetMap. Open Database License ODbL (https://opendatacommons.org/licenses/odbl/). https://www.openstreetmap.org/#map=15/47.6602/9.4736 (accessed February 13, 2024)

Information about dataset provided by TomTom

Network

- City: Friedrichshafen
- Track: Henri-Dunant-Strasse to Löwentaler-Strasse (ca. 3 km)

Time period

2017-2019, 36 months

Kind of information

- Vehicle counts
- Vehicle speed distribution

for each detector

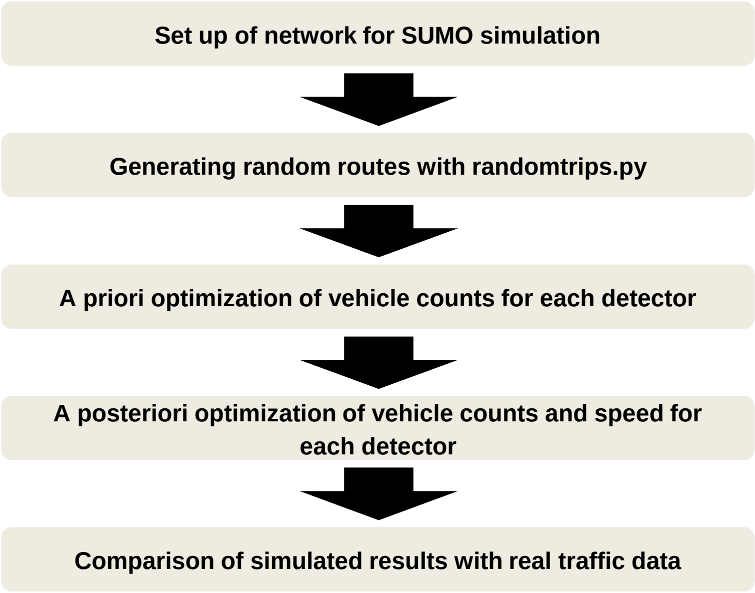
Aggregation of dataset for each detector

- One representative working and weekend day of each month
- Each day subdivided into time intervals of 2 hours

What is the best approach to set up a realistic traffic simulation with the existing dataset?

Methodology

Methodology: Workflow



Set up of network for SUMO simulation

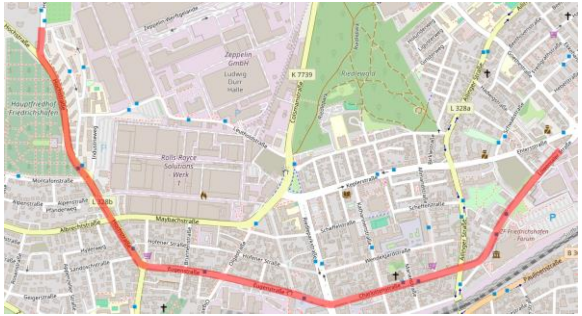
Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

A posteriori optimization of vehicle counts and speed for each detector

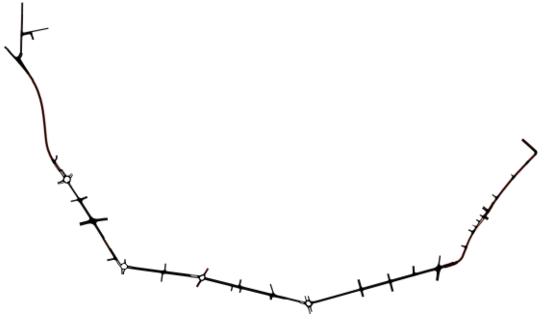
Comparison of simulated results with real traffic data

OSM Friedrichshafen

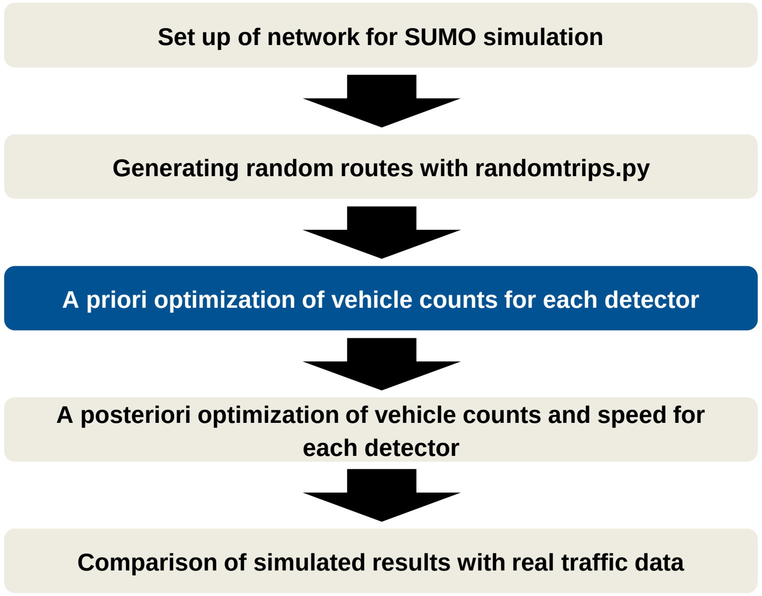


Source: OpenStreetMap. [Map section Friedrichshafen][Map]. Map data from OpenStreetMap. Open Database License ODbL (https://opendatacommons.org/licenses/odbl/). https://www.openstreetmap.org/#map=15/47.6602/9.4736 (accessed February 13, 2024)

SUMO Friedrichshafen



Methodology: A priori optimization


Set up of network for SUMO simulation
Generating random routes with randomtrips.py
A priori optimization of vehicle counts for each detector
A posteriori optimization of vehicle counts and speed for each detector
Comparison of simulated results with real traffic data

Problem Formulation: Integer Linear Programming

$$\min \boldsymbol{c}^T \boldsymbol{x}$$

subject to:

$$\boldsymbol{b}_l \le \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}_u$$

$$\boldsymbol{x} \ge 0, x_i \in \mathbb{Z}$$

$$\boldsymbol{b}_l = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Detector edge

$D_i \ (i = 1, \dots, m)$

$$\boldsymbol{b}_u = \begin{bmatrix} counts, D_1 \\ counts, D_2 \\ \vdots \\ counts, D_{m-1} \\ counts, D_m \end{bmatrix}$$

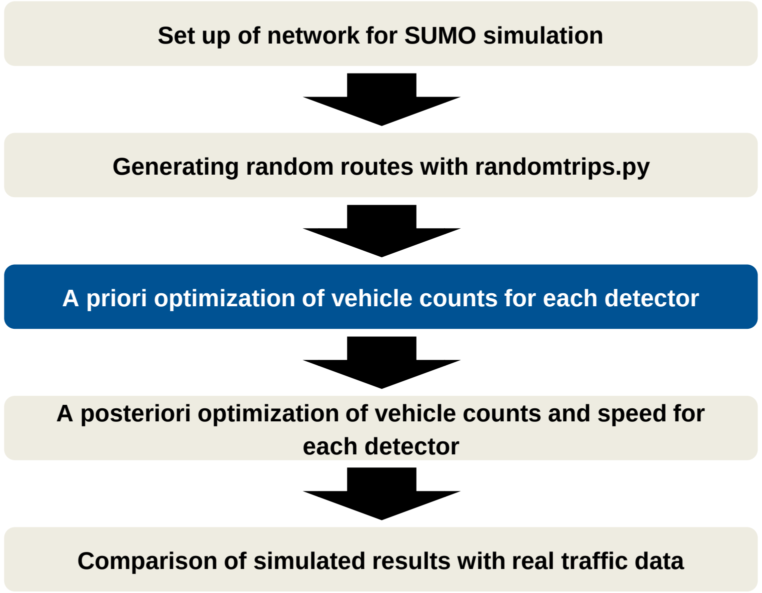
n: number of routes

m: number of detectors

$$\boldsymbol{A}' = \begin{bmatrix} D_1 & D_1 & \cdots & D_1 \\ D_2 & D_2 & \cdots & D_2 \\ \vdots & \vdots & \vdots & \vdots \\ D_{m-1} & D_{m-1} & \cdots & D_{m-1} \\ D_m & D_m & \cdots & D_m \end{bmatrix} \Rightarrow \boldsymbol{A} = [a_{ij}]_{m \times n} \; with \; a_{ij} = \begin{cases} 1 \; if \; a'_{ij} \in Route \; j, \\ 0 \; else. \end{cases}$$

1. Create Matrix $\boldsymbol{A}'$ $(\dim \boldsymbol{A} = m \times n)$ with **m** detector edges and **n** generated routes
2. Run trough each column (route) and check if route contains a detector edge
3. If yes, set the corresponding detector edge to 1, otherwise to 0

Methodology: A priori optimization



Set up of network for SUMO simulation

Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

A posteriori optimization of vehicle counts and speed for each detector

Comparison of simulated results with real traffic data

Problem Formulation: Integer Linear Programming

Route 1

Route n

$$\begin{array}{ccccccc} \boldsymbol{b_l} & & \boldsymbol{A} & & \boldsymbol{x} & & \boldsymbol{b_u} \\ \begin{bmatrix}0\\0\\\vdots\\0\\0\end{bmatrix} & \le & \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \vdots & \vdots\\ a_{(m-1)1} & a_{(m-1)2} & \cdots & a_{(m-1)n}\\ a_{m1} & a_{m2} & \cdots & a_{mn}\end{bmatrix} & & \begin{bmatrix}x_1\\x_2\\\vdots\\x_{n-1}\\x_n\end{bmatrix} & \le & \begin{bmatrix}counts, D_1\\ counts, D_2\\ \vdots\\ counts, D_{m-1}\\ counts, D_m\end{bmatrix}\end{array}$$

Multiplicator of each route

$\sum$

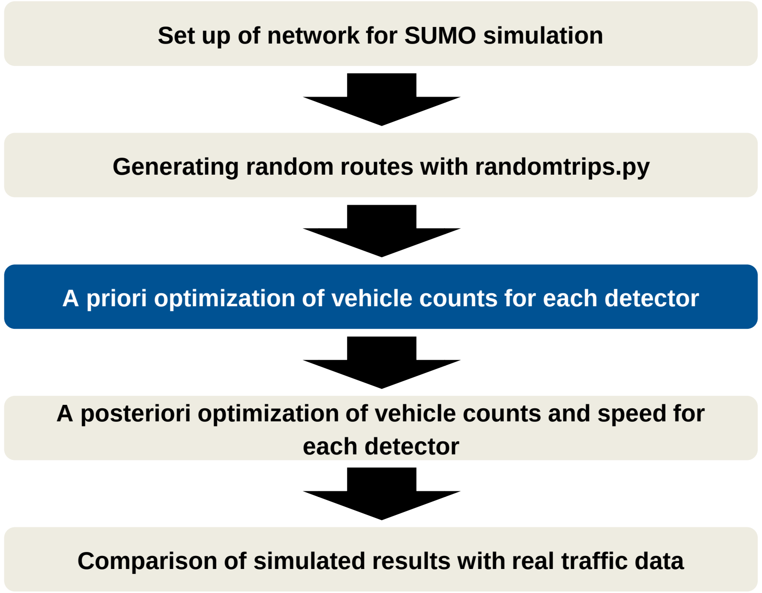
$$\boldsymbol{c}^T = \left[\sum_{i=1}^{m} a_{i1} \quad \sum_{i=1}^{m} a_{i2} \quad \cdots \quad \sum_{i=1}^{m} a_{in}\right]$$

$$\max F(x) = \sum_{i=1}^{m} a_{i1} \cdot x_1 + \cdots + \sum_{i=1}^{m} a_{in} \cdot x_n = \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} \cdot x_j$$

$$\min F(x) = -\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij} \cdot x_j$$

$\sum$ Counts along all routes

Methodology: A priori optimization



Set up of network for SUMO simulation

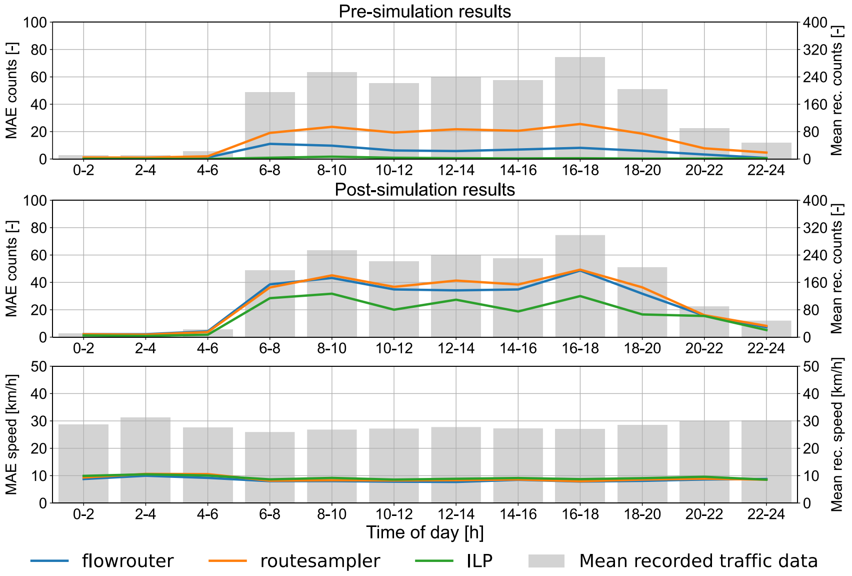
Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

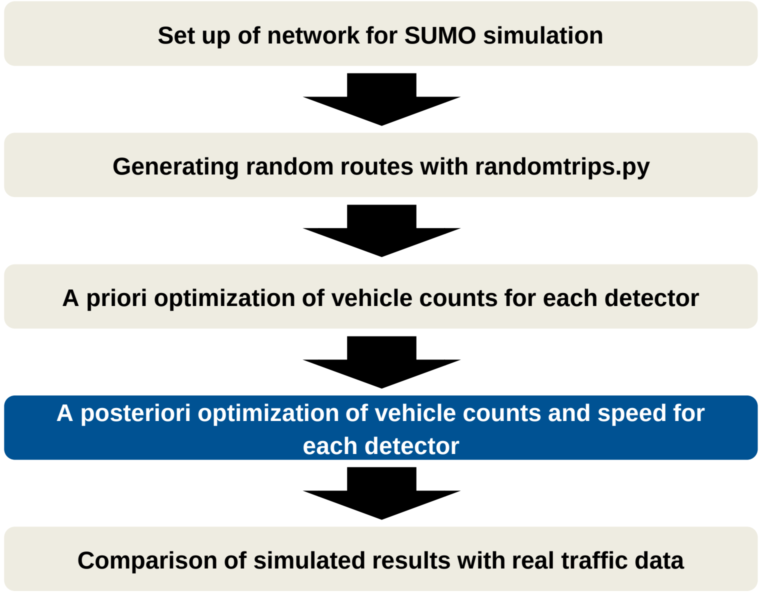
A posteriori optimization of vehicle counts and speed for each detector

Comparison of simulated results with real traffic data

Simulation results



Methodology: A posteriori optimization



Set up of network for SUMO simulation

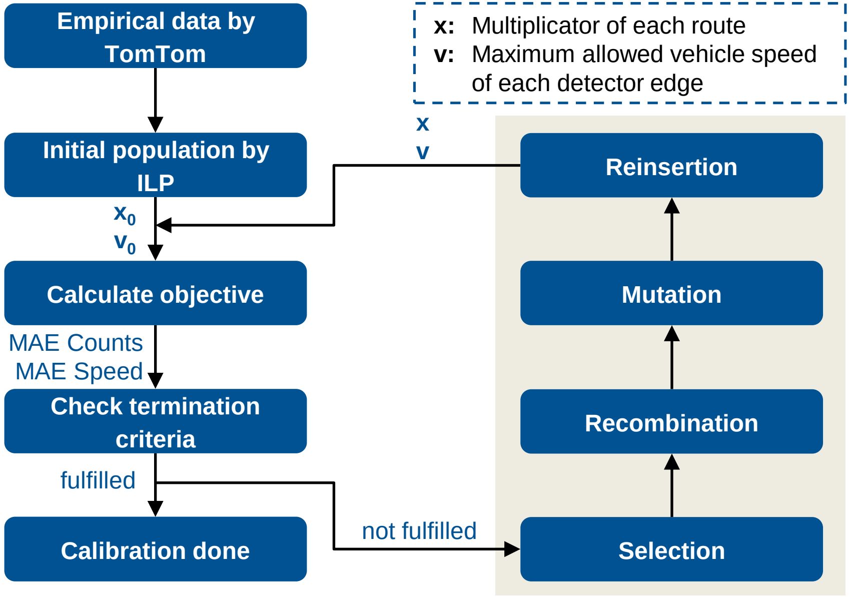
Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

A posteriori optimization of vehicle counts and speed for each detector

Comparison of simulated results with real traffic data

Problem Formulation: ILP+Evolutionary Algorithm



Empirical data by TomTom

x: Multiplicator of each route
v: Maximum allowed vehicle speed of each detector edge

Initial population by ILP

Calculate objective

Check termination criteria

Calibration done

Reinsertion

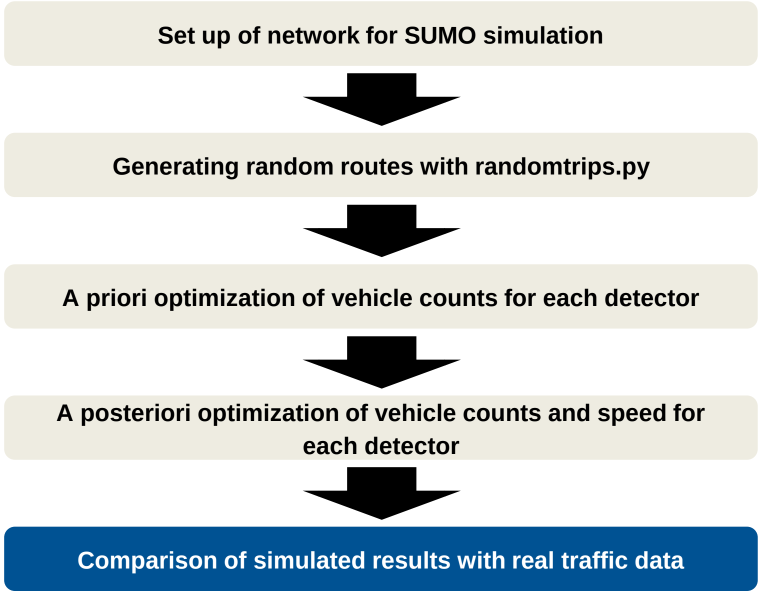
Mutation

Recombination

Selection

Results

Results



Set up of network for SUMO simulation

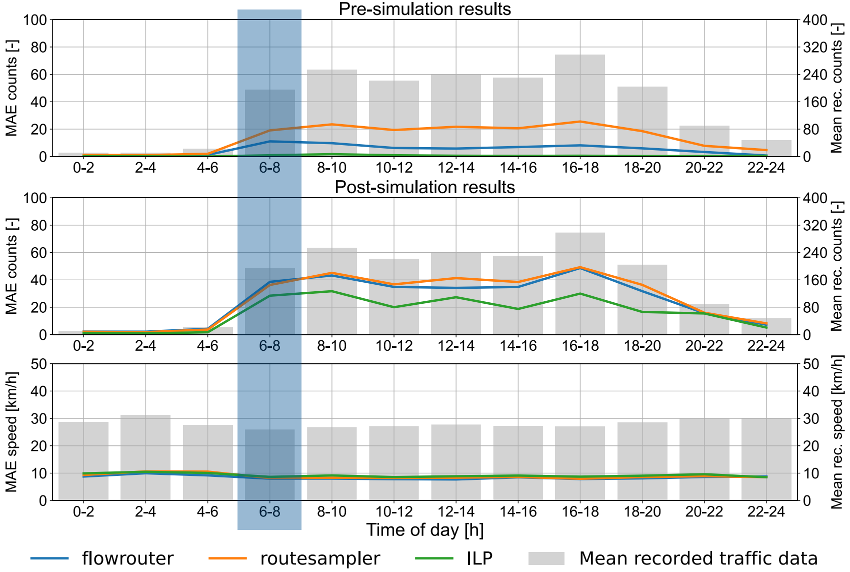
Generating random routes with randomtrips.py

A priori optimization of vehicle counts for each detector

A posteriori optimization of vehicle counts and speed for each detector

Comparison of simulated results with real traffic data

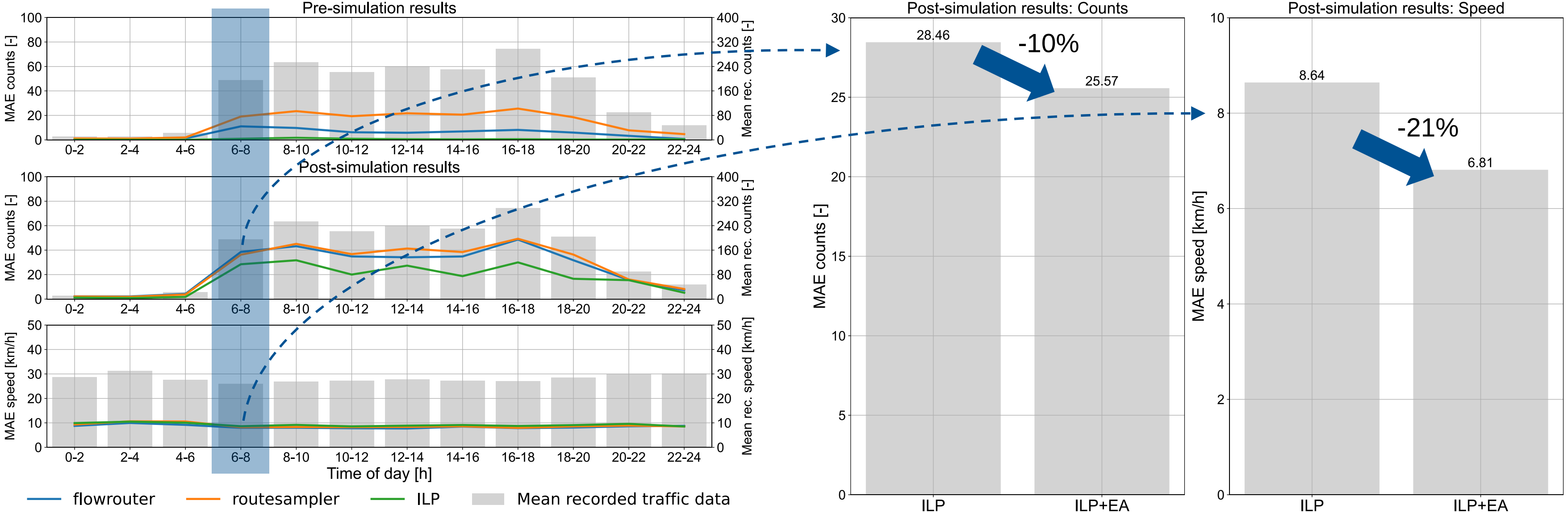
Simulation results: A priori optimization



Results

Simulation results: A priori optimization

Simulation results: A posteriori optimization



Conclusion & Outlook

Conclusion & Outlook

Conclusion

Method of two-step optimization for the calibration of traffic simulations by using vehicle count and speed measurements was presented.

Method was implemented and tested in a subnetwork of Friedrichshafen. Method was compared with the SUMO tools **flowrouter** and **routesampler** showing better results than these tools.

Outlook

Method of a posteriori optimization will be extended to a larger time frame.

Method offers the possibility for traffic-based testing for AD/ADAS-development.

Thank You.

Appendix

Methodology: Overview of methods

	Flowrouter	Routesampler	ILP approach	ILP+EA approach
Input	Network Edge based count data	Network Edge based count data Initial set of routes	Network Edge based count data Initial set of routes	Network Edge based count data Initial set of routes Edge based speed data
Optimization method	Maximum flow problem A priori optimization	Linear Programming A priori optimization	Integer Linear Programming A priori optimization	Integer Linear Programming Evolutionary Algorithm A posteriori optimization
Optimization objective	Count data	Count data	Count data	Count data Speed data
Output	Route-file Flow-file	Route-file	Route-file	Route-file Speed-file