Creating, Calibrating, and Validating Large-Scale Microscopic Traffic Simulation

Theophile Cabannes[1]
Google Research
cabannes@google.com

Alben Rome Bagabaldo
Department of Civil and Environmental Engineering
University of California, Berkeley, CA 94720, USA
bagabaldo@berkeley.edu

Junghwan Lee[2]
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720, USA
johnjhlee@berkeley.edu

Qianxin Gan
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720, USA
ganqx2001@berkeley.edu

Ayush Jain[1]
Google
ayush.jain@berkeley.edu

Alice Blondel
École Nationale Supérieure de Techniques Avancées Paris
aliceblondel99@gmail.com

Alexandre Bayen[3]
Department of Electrical Engineering and Computer Sciences
Department of Civil and Environmental Engineering
University of California, Berkeley, CA 94720, USA
bayen@berkeley.edu

Published: January, 2023

[1]Work done at UC Berkeley.
[2]Worked for Google at the time of the publication.
[3]Worked for Google at the time of the publication.

ABSTRACT
The challenges of creating, calibrating, and validating a traffic microsimulation are not apparent until one tries to create their own. Through the development of a traffic microsimulation of the San Jose Mission district in Fremont, CA, this article shares a blueprint for creating, calibrating, and validating a large-scale microsimulation of any city. Codes and data are made openly available for anyone to reproduce the simulation or its creation inside the Aimsun microsimulator. The calibration process enables simulating the movement of 130,000 vehicles through a Fremont subnetwork with more than 4,000 links using a representative 2019 afternoon six-hour demand. Executing the simulation on calibrated data gives a linear regression between the simulated and real data with slope of 0.976 and R^2 of 0.845 across 83 sensors at 15-minute time intervals.

Keywords: Traffic Simulation, Digital Twin, Microsimulation, Large-scale Networks

INTRODUCTION

Motivation

Every year, each person living in a city in the United States loses 84 hours and 33 gallons (124.92 liters) of fuel in traffic congestion (*1*). Traffic congestion represents a loss in productivity valued at $15B annually (*1*), and road transportation is responsible for around 15% of the world's global energy consumption (*2*). These statistics represent substantial and significant optimality gaps yet to be addressed, the mandate for which lies with cities through the implementation of transportation policies that can mitigate traffic congestion and improve transportation-system efficiency.

To understand the impact of a city's candidate-transportation policy, transportation planners have three main options. First, they can perform case studies. For example, if the city of Fremont, CA would like to understand the impact of modifying the traffic signal timing plans on Mission Boulevard (a major street in the city) has on cut-through traffic (*3*), the traffic engineers can attempt to extrapolate from what the city of Leonia did for Ford Lee Road (*4*) or what the city of Pleasanton did for Dublin Canyon Road (*5*). If there are enough case studies, standardized machine learning techniques can be used to make statistically significant predictions as to how a chosen policy or set of policies impacts a given traffic state. Second, cities can use trial and error (often referred to as evidence-based practice or A/B experiments). Eventually, if many trials can be done, reinforcement learning can be used to explore and find optimal policies. The city of Fremont might try to change traffic signal plans, implement turn and access restrictions, activate ramp metering, and beyond (*3*), keeping only those policies that have the most desirable impact on the public. Third, a city can develop a digital twin of its road traffic network with which to try out and learn policies in a virtual setting (*6*). The city of Fremont might take this approach to avoid disturbing its citizens by frequently changing experimental traffic signal plans. Using digital twins is especially useful when tackling complex transportation challenges in which there is a lack of case studies, or it is infeasible to rapidly and affordably test potential solutions in the real world. Such cases include planning sustainable transportation systems where coordination between connected vehicles, transportation planners, and traffic managers is critical.

Depending on a city's goals and the type of candidate policies the city's traffic engineers seek to evaluate and implement, several types of digital twins/simulations can be constructed. This includes land use models (*7*) which help defining zoning or activity-based models (*8*) which display commutes or accessibility and attractiveness of a district, as shown in figure 1. Cities need to develop microsimulations in case they want to reproduce individual driving behaviors and fine-grain traffic resolution. Among the motivations of the authors' work is a desire to understand the impact of information-aware routing and traffic-calming measures (e.g., traffic signal timing changes) on congestion and business attractivity in the Mission San Jose district in Fremont, CA. To the authors' knowledge, within the digital twin options, only microsimulations can help understand the impact of a change in traffic signal timing plans on road traffic at a small network scale.

This article aims to give city planners and traffic engineers the necessary tools and methodology to create, calibrate, and validate a large-scale road traffic microsimulation (i.e., where routing behaviors impact the state of traffic). An accurate traffic simulation model will facilitate transportation engineering in multiple aspects, from traffic congestion improvement (*9*) to applications for autonomous driving (*10*). Existing literature provides a high-level overview and comparison of traffic simulation and its development (*9–12*), the benefits and applications of traffic simulation (*10*), and an abstract framework for constructing a simulation (*13–15*). Existing literature also includes a closer examination of simulation models with various techniques for calibration

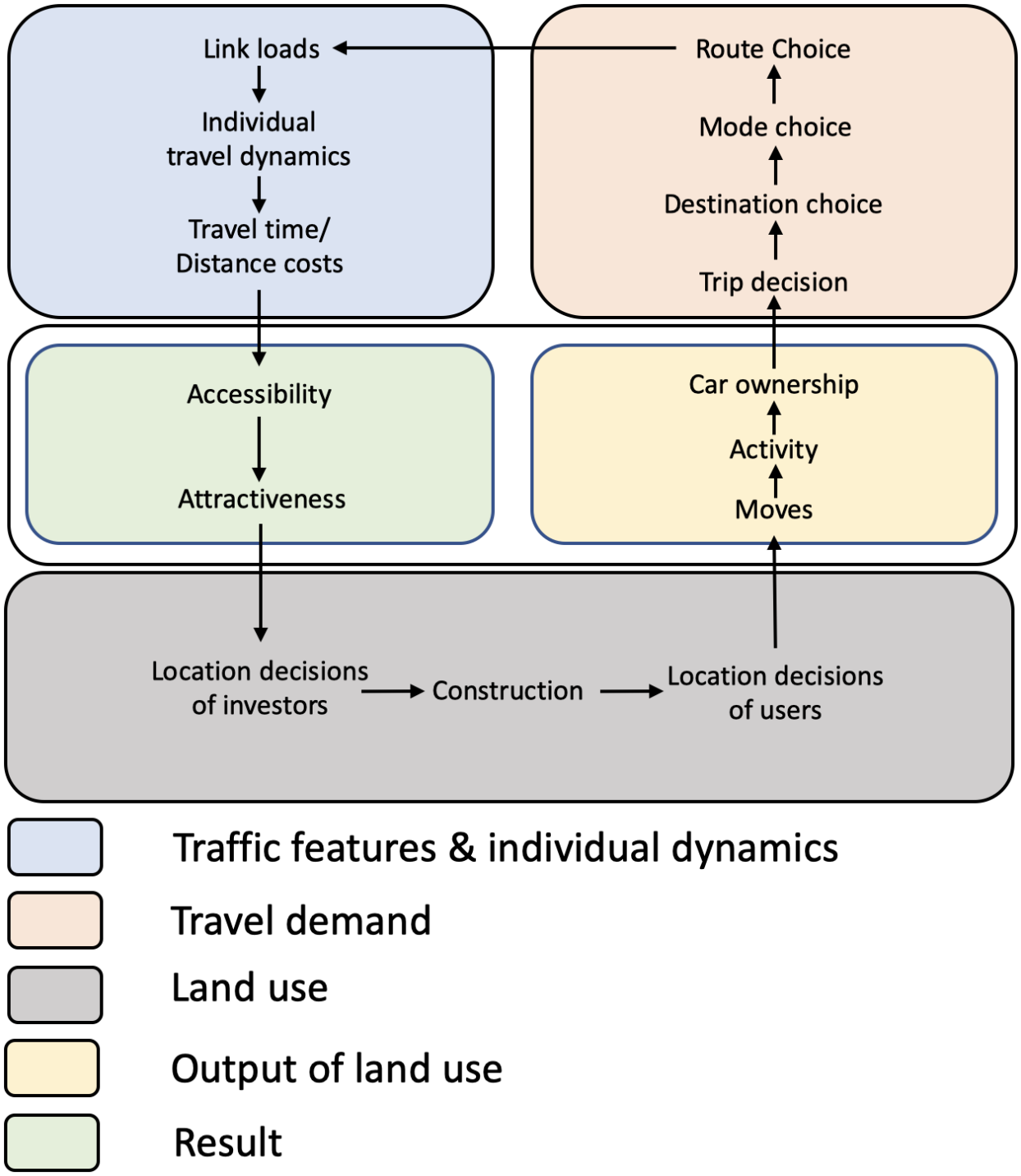

FIGURE 1 The Land Use Cycle (*7*) shows the relationship between transportation and city planning. Each color represents a type of model that planners can use. Microsimulations fall across the red, blue, and green boxes. Using OD demand input, it models route choice, link loads, as well as time and distance costs. Its output can be used to compute accessibility and attractiveness indicators, among others.

and validation of simulations under different scenarios (*16–21*). This article has a similar pattern of creating a microsimulation compared to (*14*), but it also introduces a machine learning based method for model calibration (7.6). The authors found that none of the above-mentioned literature shares any generalizable or transferable blueprint for the end-to-end process of creation, calibration, and validation of a microsimulation. Nonetheless, the authors are aware that some transportation consulting firms have internal methodologies that are not public. Because the processes of developing traffic microsimulations are very similar across different cities, this article aims to provide a detailed handbook and publicly available code source for creating a microsimulation in general.

Note that traffic microsimulation only makes sense when fine-grain traffic data needs to be modeled and when case studies or A/B experiments are unavailable or unrealistic. Traffic microsimulations cannot be used for demand analysis (such as for assessing the impact of ride-hailing companies with the respect to the number of trips (*22*)) or mode shift analysis (*8, 15, 23*). In addition, microsimulations are not relevant when data is missing to calibrate the simulation.

Contributions of the article

The first contribution of this work stands in sharing knowledge, processes, and lessons learned about creating a traffic microsimulator with those interested in designing one of their own. It aims to help such individuals assess the work needed by offering an overview of what end-to-end traffic microsimulation design entails. The authors believe that most of the challenges of running a traffic simulation are not apparent until one creates their own, and thus illustrate here in detail the necessary steps involved.

The second contribution of this work is to provide a realistic, open-source simulation with an example network with nearly 5,000 links where traffic conditions dictate the dynamic routing of an estimated afternoon peak demand (2 PM-6 PM) of 130,000 vehicles – namely, the Mission San Jose district in Fremont – to researchers that seek to try out and test ideas in a similar environment.

Finally, this article aims to present a unified pipeline for the creation, calibration, validation, and execution of a traffic microsimulation by sharing open-source code to reproduce every step the authors describe herein.

Organization

This article first gives an overview of the different type of simulations a city can perform (Section 3. It describes the process to create, calibrate and validate traffic microsimulations (Section 4). Then, the article specifies the input data needed (Section 5) and the steps to follow (Section 6) to run an Aimsun simulation. Finally, it describes the calibration process (Section 7) and the output analysis that can be done after running the simulation (Section 8). Each section is accompanied by references to the open-source code repository[4] such that the reader can replicate the processes discussed in this article. In addition, data and results from Fremont, California, USA, are included to provide motivating examples for the reader.

EXISTING TRAFFIC SIMULATOR

To understand the type of simulations that cities can use, it is important to understand how traffic is shaped by cities.

[4] `https://github.com/Fremont-project/traffic-microsimulation`

First, transportation planning is inherently related and interconnected to urban planning. Specific interaction within the land use cycle (Figure 1) can be cast using different type of models, depending on the problem a planner might want to solve.

Land use models
Land use models (*7*) estimate how accessibility and attractiveness impact the land use. As an illustration, using traffic data and demographics as input, UrbanSim (*24*) models household, employment, real estate, and potential evolution in job location, urban density, and building constructions.

Travel demand models and simulations
Using land use data coupled with demographics data, travel demand models estimate the need for transportation (*15, 25*). More specific, they estimate and forecast the travel and traffic demand. These models are relevant to estimate mode-shifting opportunities or latent demand, for example. There are three main approaches of transport demand modeling.

Trip-based models (*15*) are the historical way to generate facility demand from sociodemographic data. In particular, since the 1950s, the classical four-step model (*26*) has been used extensively by planners. In this model, a trip is an origin, and a destination. Trips are aggregated across space in origin-destination demand matrix (**OD matrix**) using transportation analysis zones (**TAZs**). The model first generates a distribution of outbound and inbound trips across transportation analysis zones (TAZs) (*trips generation*). Second, the inbound trips and outbound trips across all TAZs are matched into origin-destination (OD) matrices (*trip distribution*). Third, for each OD, the modal split is estimated (*modal split*). Finally, the traffic demand is assigned to routes using the static traffic assignment (*route assignment*). The four-step model is a static model, that estimates total travel during a day without modeling any dynamics in the network.

Activity-based models (*8*) extends the trip-based models by associating to each trip a purpose (work, shopping, or leisure). This feature enables modeling heterogeneous departure times. It also enables taking into account the demand elasticity with respect to the network traffic conditions. MATSim (*27*), TransCAD (*28*), Visum, or CEMDAP (*29*) are few software products that enable solving activity-based models.

Finally, **agent-based models** extend the activity-based model by modeling each agent, instead of each trip (*23*). Therefore, agent-based models can reproduce carpooling behaviors, and model that most commuters come back to their home after working during the day. MATSim (*27*), BEAM (*30*) (that was built on MATSim), or ActivitySim (*31*) are few software products that enable solving agent-based models. Agent-based models can also be solved by researchers using their own algorithm, like (*6*).

The travel demand models do not represent the traffic dynamics on individual links.

Traffic models and simulators
Assigning the OD demand to dynamic routes is done by **dynamic traffic assignment** models (*32, 33*). Once the route of each vehicle is known, computing the dynamic road section traffic loads in the network is called **dynamic network loading** (*32*).

In this dissertation, we refer to the traffic demand as the demand that travels using the road network, without a route being specified. However, in the transportation engineering academic literature, the traffic demand might refer to a different notion: in the four-step model (*26*), it includes the route assignment. Our choice is motivated by the fact that we are interested in understanding

the impact of information-aware routing on traffic. With the rise of information-aware routing behaviors, the author believes that the **traffic demand should not include the individual route choice** anymore. Consequently, traffic models might refer to models that perform both the dynamic traffic assignment and the dynamic network loading, but it might also refer to models that only perform dynamic network loading.

Three types of traffic flow modeling exist (*10, 32*).

Macroscopic models aggregate vehicles into traffic flow, making the internal assumption that traffic behaves like a fluid (*32*). These models can be static (like the four-step model (*26*), or the static traffic assignment ((*34*))) or dynamic. Aimsun (*35*), Cube, POLARIS (*36*), Visum, TransCAD (*28*) are few macrosimulation software products.

Microscopic models represent each vehicle in the network (*32*). Microsimulation reproduces individual driving behaviors and fine-grain traffic resolution. The interaction of each vehicle with its environment is modeled using car-following models, lane-changing models, and route-choice models (*32*). Aimsun (*35*), Cube, TransModeler, POLARIS (*36*), SUMO (*37*), Vissim (*38*), MISIMLab, Synchro, BEAM (*30*), Paramics (*39*), ActivitySim (*31*) are few microsimulation software products.

Mesoscopic models fit in between and represent a compromise between macroscopic and microscopic modelling (*32*). They model every vehicle, but only the interaction between a vehicle and the traffic flow is modeled. The interaction between a vehicle and the traffic flow can be modeled with fundamental diagrams of traffic flow, exit-functions, or queuing models (*32*). Some mesoscopic models are event-based models: they model interaction between vehicles and the traffic flow as events that might not require to model time continuously. As an example, the mean-field routing game introduced in (*40*) is an event-based mesoscopic model that uses fundamental diagrams of traffic flow. Aimsun (*35*), Cube, TransModeler, POLARIS (*36*), SUMO (*37*), Vissim (*38*), MISIMLab, Synchro, BEAM (*30*), Paramics (*39*), ActivitySim (*31*) are few mesosimulation software products.

In the team's knowledge, within the traffic simulations options, only microsimulations can help understand the impact of a change in traffic signal timing plans on road traffic at scale. Within the available traffic microsimulator, Aimsun is used in this work because it includes macroscopic, mesoscopic and microscopic simulation options, and it can be used with an academic license.

SIMULATION OVERVIEW AND ITS CREATION PROCESS

Traffic simulations (*10*) provide traffic information visualizations and related figures, which include vehicle hours traveled (VHT), vehicle miles traveled (VMT), mean delay per vehicle, gas emission, accessibility index, etc., for a comprehensive analysis of the design and efficiency of the transportation system in question (Section 8).

In this article, the authors refer to traffic simulations as the simulation of road traffic (vehicle flows in the network over time) with key inputs of traffic demand (people's origin, destination, and departure time grouped by timed origin-destination matrices) and a road network (including road sections, lanes, intersections, road signs, and traffic signal timing plans).

Simulations can be aggregated macroscopically, mesoscopically, and microscopically (*10*). Macrosimulation focuses on the aggregation of traffic flow and demand, while mesosimulation breaks traffic flow into smaller groups and examines the behavior of the whole in those groups. In this work, Aimsun Next 22 (*35*) is used to perform microsimulations, where the focus is on the individual elements in a transportation system.

In a microsimulation, individual vehicles are generated and assigned to a route, which is then simulated across the lanes of the input network's road sections (*10*). Before being generated, each vehicle is defined by an origin, destination, departure time, and optionally, a vehicle type. The vehicle input data are aggregated across space and time into timed origin-destination (OD) matrices for each vehicle type. Space aggregation uses transportation analysis zones (TAZs), while time aggregation uses time buckets. Lanes of contiguous road sections are connected through unsignalized intersections (with yield or stop signs) or signalized intersections (with given traffic signal timing plans and a master control plan). Assigning each vehicle to a route is sometimes referred to as route assignment, while the simulation of vehicle movement through the network is commonly referred to as dynamic network loading (*10*). Simulated link flows and network traversal times can be compared to ground data with which to calibrate and validate a proposed model. For the example Fremont San Jose Mission district microsimulation, input data are described in (Section 5).

Generally, when it comes to modeling transportation systems, there exists a notable tradeoff between the number of model variables and the risk of overfitting, a result of the large quantity of data needed to calibrate complex transportation models (*21*). With this in mind, real data set aside for model calibration should be further split into training and testing data to decrease the overfitting risk (*41*).

Before calibrating a microsimulation (Section 7), one needs first to fix any existing network and demand issues (connectivity issues, wrong number of lanes, wrong traffic signal plans, small mistakes in the master traffic control plan, obvious error in the demand data). The first phase of calibration is done without simulation by matching simulated and ground total counts of vehicles entering or exiting the network. This is then followed up by the second phase of calibration, done through macrosimulation. Once the OD demand is calibrated, the driving behaviors (routing, car-following, lane-changing models), and microsimulation parameters (like simulation time step) can be calibrated using optimization algorithms that work with expensive function evaluations (this work uses a genetic algorithm that is highly parallelizable).

Once calibrated, the microsimulation can be validated using eyeball estimation or concrete metrics alike. Eyeballing here mainly consists of understanding where and when the congestion occurs in the input network and checking for consistency with any prior knowledge about the network's congestion. Metrics of effectiveness (MOE) can then be used for a more rigorous second validation. For example, the mean delay per vehicle over time in the network indicates when the peak hour happens in the network and is a strong indicator of the global quality of the simulation. Finally, more specific data like detector flows and network traversal times can similarly be used to validate the simulation against ground data.

Once the simulation is created, calibrated, and validated, it can be used for analysis (Section 8). For example, causes of congestion can be derived and policies to mitigate these can be tested.

INPUT DATA DESCRIPTION

The required inputs for a microsimulation are a network (Section 5.1) and a dataset of timed origin-destination demand (Section 5.2). Traffic data (Section 5.3) may also be used to calibrate and validate the simulation. The data used by the authors to simulate the traffic in the Mission San Jose district around Interstate 680 and Mission Boulevard (State Route 238/262) in Fremont, CA is openly available and the process of calibrating the input data and importing it in Aimsun is

reproducible[4].

Network

The road network is made up of constituent road sections connected through signalized or unsignalized intersections. To create this network, the authors downloaded the OpenStreetMap (OSM)(*42*) network model using the bounding box defined by the following coordinates: North: 37.5524, East: -121.9089, South: 37.4907, and West: -121.9544 and first cleaned it in ArcGIS(*43*) (see figure 2). After importing the network into Aimsun (*35*), Google satellite, Maps, and StreetView images were used to perform manual adjustments to ensure the accuracy of connectivity, yield and stop sign locations, and lane counts (see figure 3). Speed limits are inferred using the data provided by the City of Fremont and road capacities are adjusted using the data from the Behavior, Energy, Autonomy, and Mobility (BEAM) model which is an open-source agent-based regional transportation model (*30*). Then, traffic signal plans (including the ramp meters and the master control plan) from the city and CalTrans were added using the Aimsun graphical user interface (GUI). Finally, traffic-calming measures (primarily turn-restrictions) were created in the simulator. In summary, the modeled network has 4,705 links with a total of 393.27 km section length. This includes 111 freeway sections, 373 primary road sections, 2,916 residential road sections, and 2,013 nodes (intersections), 313 of which have stop signs and 37 of which have traffic lights (26 operated by the city and 11 operated by CalTrans). The overall process to create and fix the network (with traffic signal plans) took our team about 600 person-hours to complete.

Origin-Destination Demand

The origins, destinations, and departure times for every vehicle are aggregated into timed origin-destination demand (TODD) matrices. Origins (or destinations) are clustered into transportation analysis zones (TAZ), which are bijective to the set of centroids connected to internal or external entry/exit points in the network. The connections between the centroids and these points (nodes) in the network are called centroid connectors (*44*). The 20.8 square kilometers network area is divided into 76 internal centroids and 10 external centroids (see figure 4). Departure times are aggregated into 15-minutes time intervals. Between 2pm and 8pm, 130,000 vehicles are modeled (including 62,000 commuters and 68,000 residents). In this work, the demand data was derived from the SF-CHAMP demand model (*45*) from the San Francisco County Transportation Authority and from a StreetLight study performed for the City of Fremont. Unfortunately, there is no reproducible process to create accurate demand data as of now, which is where most of the major challenges of realistic traffic simulation remain. However, demand data accuracy can still be slightly improved through calibration against ground data (Section 7).

The overall process to create and calibrate the origin-destination demand took our team around 600 person-hours to complete.

Traffic data

To calibrate against ground data, one can utilize ground flow, speed, or/and travel time data, each of which can be directly imported within Aimsun as a Real Data Set. In this study, flow data is generated from 56 city flow detectors and 27 CalTrans Performance Measurement Systems (PeMS) detectors (*46*). Speed and travel time data can be acquired using the Google Maps API[5]. In this study, travel time data was gathered from driving in the area. The overall process to create traffic

[5] `https://developers.google.com/maps/documentation/directions`

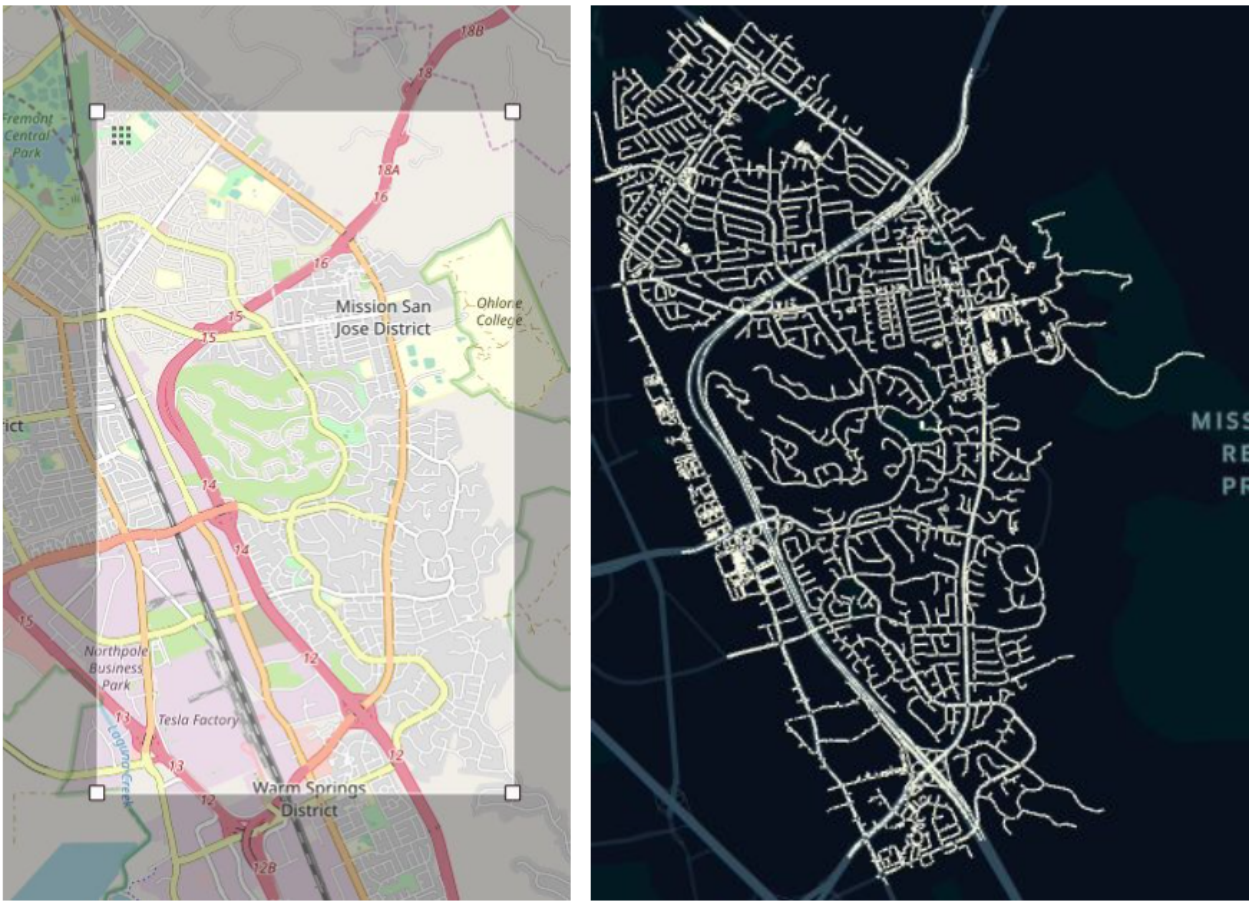

FIGURE 2 OSM network with the bounding box on the left and corresponding Aimsun network after cleaning on the right.

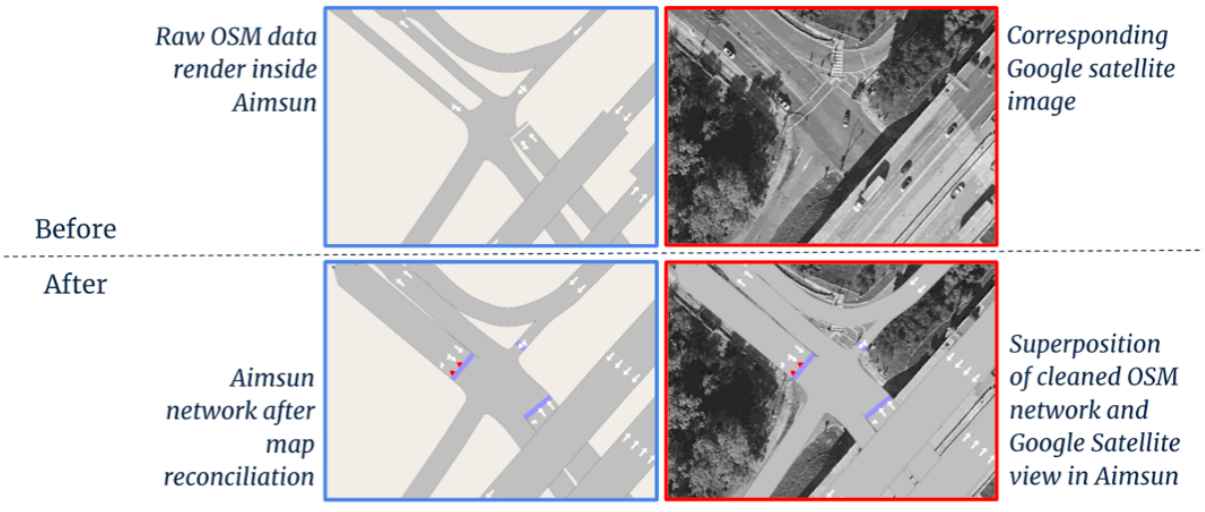

FIGURE 3 Before *(top)* and after *(bottom)* manual editing of the OSM network in Aimsun with comparison to the Google Satellite image.

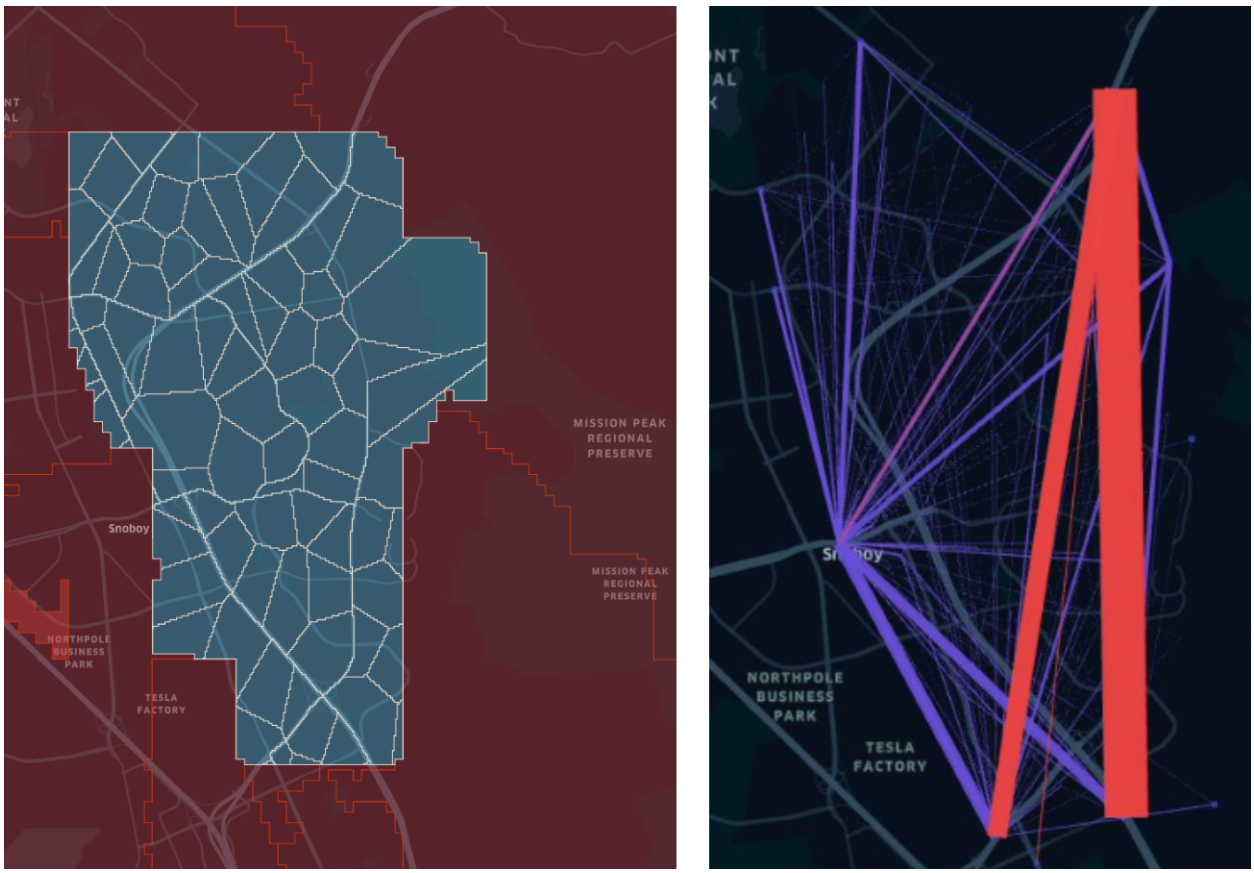

FIGURE 4 Transportation analysis zones (TAZ) (on the left) and demand plotted with desire lines (on the right). A commuter (aggregated into red lines on the right plot) is a vehicle with an origin and a destination, which are both external centroids (red TAZ on the left plot). A resident (aggregated into blue lines on the right plot) is a vehicle departing or/and arriving from or/and to an internal centroid (blue TAZ on the left plot).

data took our team 400 person-hours to complete.

SIMULATION

Once the input data is imported into the Aimsun simulator, simulations can be run, generating simulated traffic data as output.

Running a simulation

To run a traffic simulation in Aimsun, one first needs to create/import the network. Second, the OD demand data should be imported, and a traffic demand (a list of timed OD demand matrices with scaling factors) should be generated. Optionally, traffic data can also be imported. Once all imports are complete, a traffic simulation can be created, with many mutable parameters (see full list in Section 7.5). From here, the simulation can be run.

Creating and running a simulation can be done using the Aimsun's GUI, but, for the sake of reproducibility, the authors opted to write Python scripts for each step of the process[4]. The open-source repository is self-contained, and any readers with an Aimsun 22.0.1 license should be able to reproduce all the steps explained below and run the same simulation performed by the authors.

First, the network can be imported directly from OpenStreetMap (OSM) in Aimsun. For this work, the authors did some processing of the OSM network data in ArcGIS before importing the OSM network from an external file in Aimsun. Speed limits and vehicle capacity for each of the road sections can then be updated, followed by importing the ramp meters, traffic control plans, the master control plan, and traffic management strategies. Among these, correctly configuring the traffic control plans took most of the time because of the many parameters that needed to be changed for each traffic light to consider various settings and actuation. The demand data is also imported into Aimsun, which is created by importing the centroid data with the centroid connectors. The centroid data contains the OD demand data which is converted to traffic demand The real flow data, the final bit of input, is then imported as a real data set inside Aimsun.

Then, a Jupyter Notebook is designed to define any configurable model characteristics outside Aimsun (including time step size, routing model, driving behavior, or output database location) needed to generate the simulation. Finally, the simulation can be run. Running the 6-hour-long simulation takes between 30 minutes and 4 hours, with total runtime depending on the list of data to output and the input data size.

Simulation Output Data Description

Microsimulation models can generate detailed data for every vehicle corresponding to its car following, lane-changing, and gap acceptance behavior. These characteristics can be observed through simulation playback, using Aimsun's GUI to visualize each vehicle's motion through the network. These results can also be aggregated to compare the macroscopic simulation results vs. real data sets with respect to factors such as flow, speed, and travel time.

In Aimsun, while the outputs can be accessed using the GUI, it is also possible to save them as SQLite tables. The output tables[6] contained in the output database are defined in the microsimulation configuration. Each table contains different types of statistics, and it is important to identify which tables are necessary to generate before running the simulation to prevent data

[6]Described in `https://docs.aimsun.com/next/22.0.1/UsersManual/OutputDatabaseDefinition.html`

cluttering. For example, the MISYS (microsimulation system) table contains system-level statistics about the entire network, such as VMT, VHT, total gas emissions, or average delay across all road sections. The SQLite output databases are used as primary data sources for simulation outputs throughout calibration (Section 7), validation, and analysis (Section 8).

CALIBRATION

The challenges of creating a realistic microsimulation of city traffic lie in its calibration. To calibrate a microsimulation, one first needs to fix all the network issues (Section 7.1). Then, OD demand can be calibrated without simulation by matching the total counts of vehicles entering or exiting the network (Section 7.2) followed by calibration with macrosimulation by matching all detector counts in the network (Section 7.3). Once the OD demand is calibrated in this manner and after having chosen a route choice model (Section 7.4), the driving behaviors are then calibrated (Section 7.5) using a genetic algorithm (Section 7.6). The OD demand calibration and driving behavior calibration procedures can be reproduced from the provided open-source code[4].

Network and demand apparent issue fixes

A first test to ensure that the input network and demand data are not flawed can be done by running a simulation with 50% of the demand and checking that no congestion occurs in the network. Congestion can be detected with the Aimsun GUI by playing the simulation. It can also be detected with total delay in the network, or mean travel time per vehicle-mile. A second check – running the full demand – can be done by looking at gaps between the ground flow and the simulated flow that are below 50% or above 200%.

The authors used GitHub issues[7] to report, follow and solve any apparent network or demand issues. On average, such checks found 4 to 5 issues. Each issue was assigned to a team member, and around 2 issues were solved per team member per week. 43 issues of this variety were reported in total. The overall apparent issues fixes process took around 500 person-hours. The issues included:

- Updating incorrect lane connections at intersections.
- Changing erroneous road section geometries.
- Changing wrong numbers of lanes on road sections.
- Removing parking lots, where cut-throughs were performed in the simulation to avoid congestion at traffic lights.
- Updating improperly imported traffic signal plans.
- Updating master traffic control plans to solve missing synchronization between traffic signal plans. The authors found that there was one ground truth congestion issue that could be solved by synchronizing traffic lights operated by the state with the traffic lights operated by the city on Auto-Mall Pkwy and I-680.

The authors realized during this process that the input demand data used was biased towards the northwest, which is likely a byproduct of the demand being sourced from the SFCTA CHAMP demand model, which aims to replicate traffic in San Francisco (which is to the northwest of Fremont). An accurate initial demand data is key for a realistic simulation.

[7] `https://github.com/features/issues`

OD demand calibration without simulation

A first calibration of the OD demand described in (Section 5.2) can be done without simulation. To do so, the total demand is scaled up or down such that the ground flow data at every external entry or exit point matches the demand data that enter or exit the network at said point (each of which is represented by an external centroid derived from external TAZs). This approach was inspired by the scaling problem formulation approach presented in (*47*). The objective function of the minimization problem is shown in equation 1:

$$\min_{\alpha\in\mathbb{R}} \sum_{t\in\mathscr{T}} \sum_{d\in\mathscr{C}_{ext}} \left(f_{t,mapd(d)} - \alpha \sum_{o\in\mathscr{C}} \boldsymbol{d}(o,d,t) \right)^2 + \sum_{t\in\mathscr{T}} \sum_{o\in\mathscr{C}_{ext}} \left(f_{t,mapo(o)} - \alpha \sum_{d\in\mathscr{C}} \boldsymbol{d}(o,d,t) \right)^2 \quad (1)$$

Where:

- α is the demand scaling factor.
- $\mathscr{C}$ is the set of centroids.
- $\mathscr{C}_{ext}$ is the subset of external centroids.
- o and d are origin and destination centroids.
- $\mathscr{T}$ is the set of time bucket, and t is one time bucket.
- $mapd(d)$ is the detector associated with the destination centroid d.
- $mapo(o)$ is the detector associated with the origin centroid o.
- $f_{t,l}$ is the ground flow data for the time bucket t on the detector l
- $\boldsymbol{d}(o,d,t)$ is the number of vehicle that exit the origin o to reach the destination d during the time bucket t.

This approach can be used to derive how the demand changes over time in cases when the current OD demand matrices and the flow data over the years are available.

OD demand calibration with macrosimulation

Once the OD demand is calibrated against entry or exit flows, it can be calibrated against all detector flows in the network by assigning the OD demand to routes and counting the number of vehicles going over each detector. To assign the OD demand to routes without simulating each individual dynamics, the static traffic assignment can be used (*34*). OD-demand calibration aims to better align the simulated and ground detector flows. This step can be done inside Aimsun directly with the static OD demand adjustment scenarios. OD adjustment is done by solving the constrained generalized least-squares (GLS) described in equation 2 as adopted from (*48*):

$$\min_{\hat{\boldsymbol{d}}\in\mathbb{R}_+^{|\mathscr{C}|\times|\mathscr{C}|\times|\mathscr{T}|}} \sum_{t\in\mathscr{T}} \sum_{l\in\mathscr{L}} (f_{t,l} - \hat{f}_{t,l}(\hat{\boldsymbol{d}}))^2 + \gamma \sum_{t\in\mathscr{T}} \sum_{o,d\in\mathscr{C}} (\boldsymbol{d}(o,d,t) - \hat{\boldsymbol{d}}(o,d,t))^2 \quad (2)$$

Where:

- $\hat{\boldsymbol{d}}$ is the calibrated timed OD demand.
- $\mathscr{L}$ is the set of detectors.
- $\hat{f}_{t,l}(\hat{\boldsymbol{d}})$ is the simulated flow on detector l during the time bucket t given the demand $\hat{\boldsymbol{d}}$. In this subsection, the simulation is the static traffic assignment.
- $\boldsymbol{d}$ is the prior demand before the calibration with macrosimulation and after the calibration without simulation.
- γ is a scaling factor to avoid overfitting the flow data.

Because of the high number of variables that can be calibrated (namely, each element of each OD demand matrix) and the relatively low volume of ground data (flow for each time step

for each detector), overfitting is a major risk and must be accounted for. Therefore, the flow data is divided into training data and testing data. Then, the objective function is minimized against the training data and tested against testing data. If the demand is changed such that the training data is perfectly fitted, but the testing data is badly reproduced, then the calibration has overfitted the training data.

The comparison of results between the macrosimulation results after OD adjustment using the training and testing sets is shown in figure 5, where overfitting can be observed.

To reduce the risk of overfitting, a regularization term that penalizes large modifications of the prior demand was added in equation 2 to provide a balancing effect (*48*), formulated as the Frobenius norm (*49*) of the difference between the calibrated and the original OD demand matrices. This approach is not exclusive to the Frobenius norm – other norms such as the nuclear norm (*49*) could be considered for regularization. In (*50*) the l_1 norm is used to compare the OD matrix elements.

In this work, to avoid overfitting, the regularization term was scaled by a large factor $\gamma = 10$ such that the OD demand matrix after the macrosimulation calibration was very similar to the OD demand matrix after the calibration without simulation.

Finally, the validation of the calibrated matrix was done with flow regression plots. Flow regression plots compare simulated values with real-world values by scatter-plotting them as y and x-axes, respectively. A linear regression is then fitted onto to the data points to draw the line of best fit (*51*). The slope and intercept of the regression can then be compared to the ideal y=x line to determine whether the simulation model tends to over/underestimate the plotted metrics and whether there is bias in the model. Performance metrics of the linear regression (*51*), such as the coefficient of determination (R^2), root-mean-square error (RMSE, or nRMSE when normalized), and the mean absolute percentage error (MAPE) can be computed to determine the accuracy of the simulated flow.

Choice of the routing model

Once the network is bug-free and the OD demand is calibrated, the microsimulation can be run. By design, the microsimulation has many modifiable parameters to calibrate individual driver behaviors. Some of the most important microsimulation parameters are about routing behaviors.

A routing model assigns travelers to a series of links to get from one centroid (origin) to another (destination). There exist two types of routing models (*32*):

1. The one-shot assignment model assigns routes and runs the simulation once. When assigning vehicles to route, only past and current information are used, and no assumptions are made about the future. The route is given following a stochastic route choice (SRC) model.
2. The iterative assignment model assigns routes and runs the simulation iteratively until the travel cost experienced by each vehicle at the end of their trip cannot be minimized by unilaterally changing the route of the vehicle. This equilibrium state is referred to as the dynamic user equilibrium (*32*) (sometimes called Wardrop equilibrium or Nash equilibrium).

Because running many simulations iteratively takes a lot of time, the authors opted for a stochastic route choice (SRC) model. Several SRC models are available in Aimsun (fixed-route under free-flow conditions, fixed-route under warm-up period traffic conditions, binomial model, proportional model, logit model, and C-logit model). Considering the tradeoff between accuracy

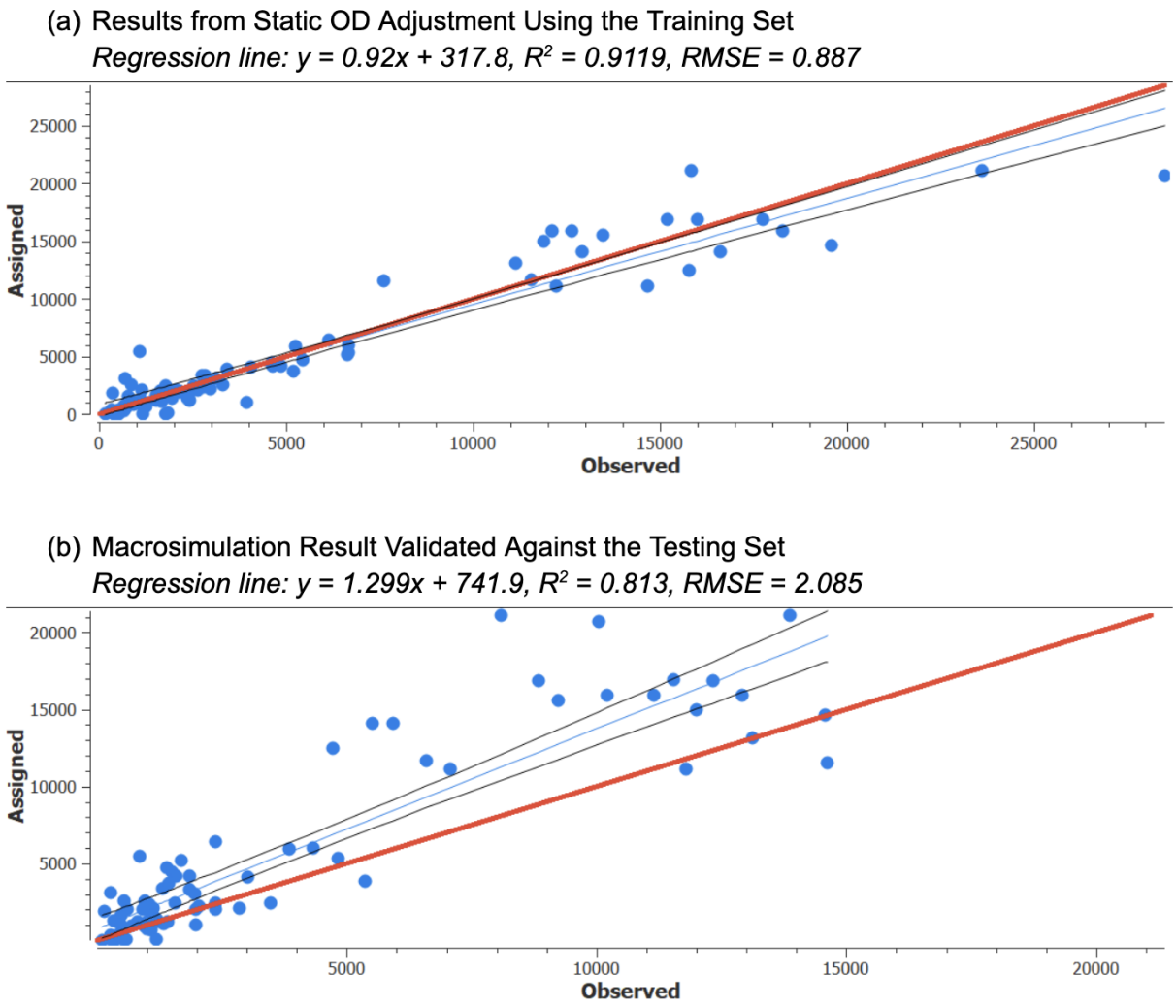

FIGURE 5 Assigned/Simulated Traffic vs. Observed/Actual Traffic Flow Linear Regression Plot after Aimsun's default OD-demand calibration with macrosimulation. Training results are reported on the top figure, while testing results are on the bottom figure. Very good training results (slope of 0.92 and R^2 of 0.9119, both close to 1) accompanying poor-quality testing results (slope of 1.299 and R^2 of 0.813, further away to 1) show that the calibration has over-fitted the training demand data.

and simulation run time, the authors chose to use the C-logit model (*52*) after experimenting with the different models.

The C-logit route choice parameters include:

- Number of alternate routes considered at each routing time step.
- Size of each rerouting time interval.
- Percentage of vehicles allowed to reroute en route.
- Averaging parameters for past and current (instantaneous) travel cost parameters.
- Route cost function parameters (utility, scaling cross-factor, overlap parameter).
- The calibration of the C-logit route choice parameters was done as part of the microsimulation calibration.

Driving behavior calibration

Once the routing choice model has been chosen, the microsimulation-specific parameters (like simulation time step length) and the driving behavior can be calibrated. The driving behavior parameters include the routing behavior parameters (like rerouting time interval), the car following parameters (like reaction time), and the lane changing parameters (like aggressiveness). The full list of parameters that can be calibrated can be found online[4].

The routing calibration aims to find the configuration of parameters that minimizes the difference between simulated data and ground truth data, without overfitting. A first manual calibration can be done based on intuition with simulation recording (for example, reaction time can be adjusted if the output to input flow ratio at some intersection seems low). Then, bounds can be set for each parameter based on physical intuitions (reaction time is between 0.2 to 3 seconds) and a systematic calibration can be performed.

In the systematic calibration, an objective function, described in equation 3, is set to be minimized (similar to OD demand calibration in equation 2):

$$\min_{\theta} \sum_{t \in \mathcal{T}} \sum_{l \in \mathcal{L}} (f_{t,l} - \tilde{f}_{t,l}(\theta))^2 \tag{3}$$

where:

- θ is a set of parameters to calibrate. If we have any prior on the set of parameters theta, a regularization term can be added to avoid overfitting.
- $\tilde{f}_{t,l}(\theta)$ is the simulated flow on detector l for the time bucket t given the calibrated demand, and the set of parameters θ.

An optimization algorithm can be run to find the optimal microsimulation-specific parameters and driving behavior parameters. Optimization algorithms include brute force algorithms like grid search, random search (*53*), classical optimization algorithms that can be found in the SciPy.Optimize toolbox (*54*), neural-network (*55*), or genetic algorithm (*56*). Because evaluating the objective function given the input parameters is costly, since it requires running a microsimulation, the authors decided to use a genetic algorithm. Genetic algorithms are particularly efficient for opaque box functions with a high stochastic effect. In addition, they are easily parallelizable and can handle multi-criteria optimization (*56*).

Genetic algorithm for microsimulation calibration

The authors followed the approach presented by (*16*) to fine-tune the driving and microsimulation parameters.

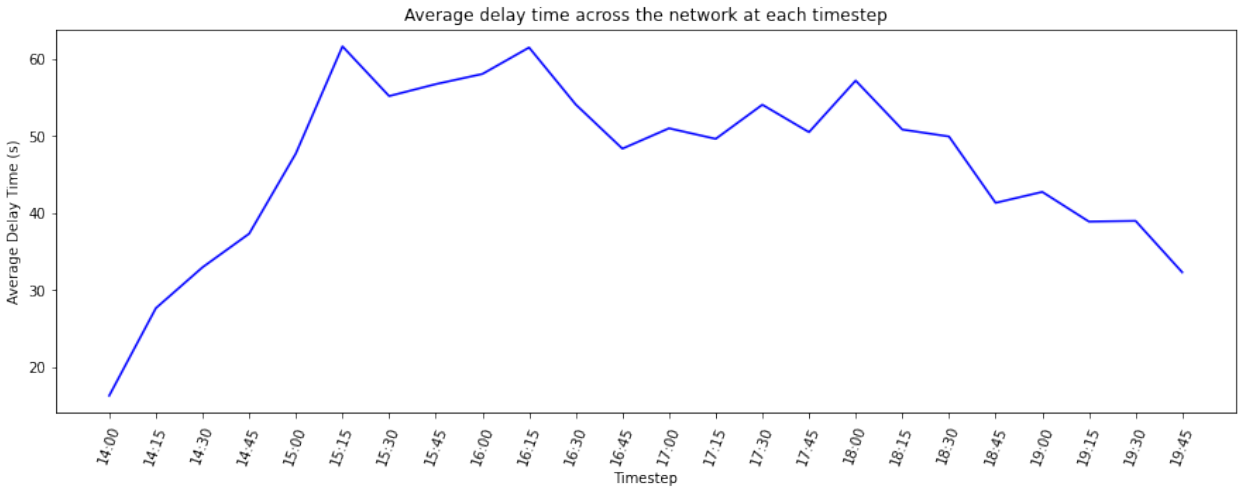

FIGURE 6 Time-series of average delay time across the entire network in Fremont, CA. We identify that peak congestion happens during 3:15PM to 4:15PM.

Because of the significant number of model parameters (35 for micro-simulation using the C-logit stochastic route choice model) and the run-time for each simulation on the authors' computers (30-40 minutes), the search space (*i.e.,* number of model parameters to calibrate) has been decreased through a sensitivity analysis. Only the 10 parameters that have the most impact on the measures of effectiveness have been selected by comparing the relative metric evolution over small increments of each parameter using the Latin hypercube sampling (LHS) algorithm (*57*). The calibration of the 10 selected parameters was done using the Non-dominated Sorting Genetic Algorithm-III (*58*) from the DEAP Python library (*59*) using the objective function described in equation 3. The initial value of the parameters was set to the Aimsun default ones if not updated based on intuition after few initial simulations. Similar to subsection 7.3, the RMSE over the flow bi-plots is minimized.

POST-PROCESSING ANALYSIS

After the simulation is created, one needs to validate the accuracy of the simulation to ensure the credibility of its results. Once the accuracy of the simulation is satisfactory, traffic analysis can be conducted to observe how certain metrics change in different scenarios.

Validation

The first step in validating the simulation is to use Measures of Effectiveness (MOE) (*13, 60*), which serve as indicators for the general correctness of system-wide results. Some examples of metrics that can be used for MOE are average delay time, total distance traveled by vehicles in the entire network, and the average number of vehicles in the virtual queue at each time step.

After determining that MOE correctness is validated, the next step is to validate system-wide and location-specific metrics for the simulation. System-wide metrics denote data encapsulating property or properties across the entire simulation network, such as flow at all detectors. Location-specific metrics, on the other hand, deal with data specific to a subset of the network, such as a corridor.

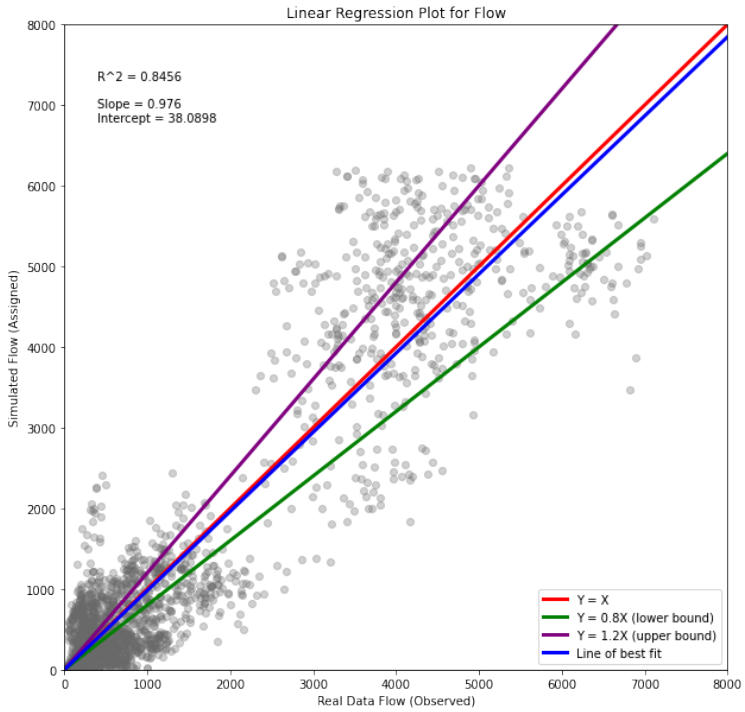

FIGURE 7 Linear regression plot of flow across 83 detectors in the entire network at each 15 minute time step.

Validation of system-wide metrics

For validation of system-wide metrics, linear regression plots comparing simulated versus real-world data are commonly used across previous literature (*9, 61, 62*). In this work, detector flow, OD travel times, and OD route distances were system-wide metrics used to validate the simulation results (Figure 7).

To conduct a more detailed analysis of linear regression plots, one can create a separate regression plot of system-wide data points for each time frame. Then, a time series of regression statistics for each regression plot can be drawn to gain insight into specific points in time when the simulation needs more calibration.

In addition, it might be useful to identify the times for each detector in which the simulated flow was over 1.2 or below 0.8 of the ground flow to know which detectors need more calibration.

Distributions of actual and simulated values can also be used for eyeball validation of the simulation. Though distributions show that the general trend that the simulated data is correctly approached, they provide less insight into the individuality of each data point than biplots.

Validation of specific location metrics

After validation of system-wide metrics, one can validate location-specific metrics in areas of high importance to ensure the accuracy of the simulation. To do so, we need to identify the scope of the location critical to our study. In this example, the authors set the I-680 corridor as the scope of specific location metric validation. Metrics one can validate include flow (veh/h), speed (km/h), delay time (h or h/veh), and density of vehicles (veh/km) observed at each detector within the corridor.

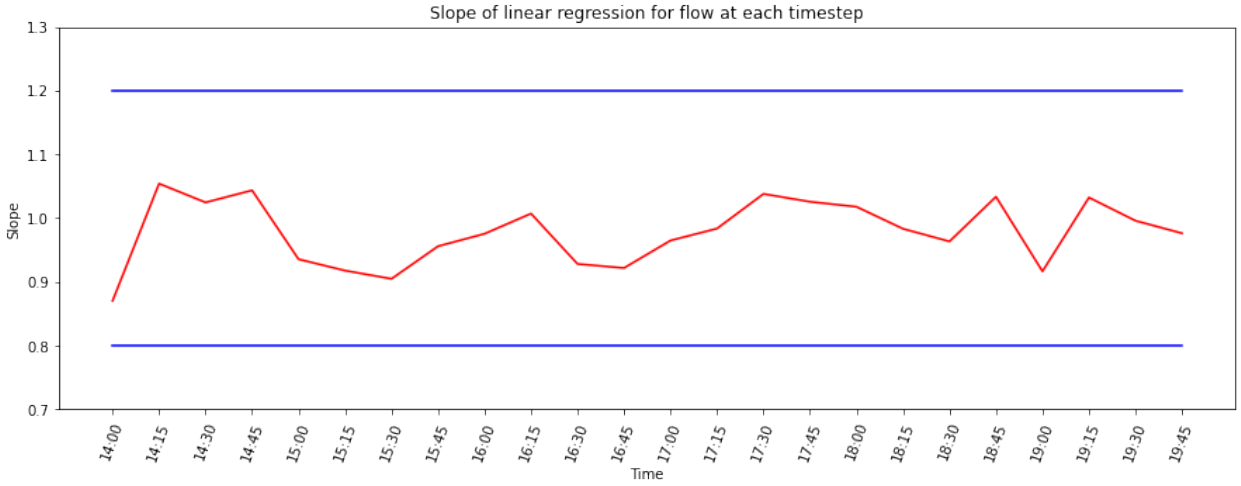

FIGURE 8 Slope of the linear regression for flow at each timestep (red). Setting arbitrary lower and upper bounds of 0.8 and 1.2 (blue), respectively, shows that the simulation does not over- or under-estimate the real flow.

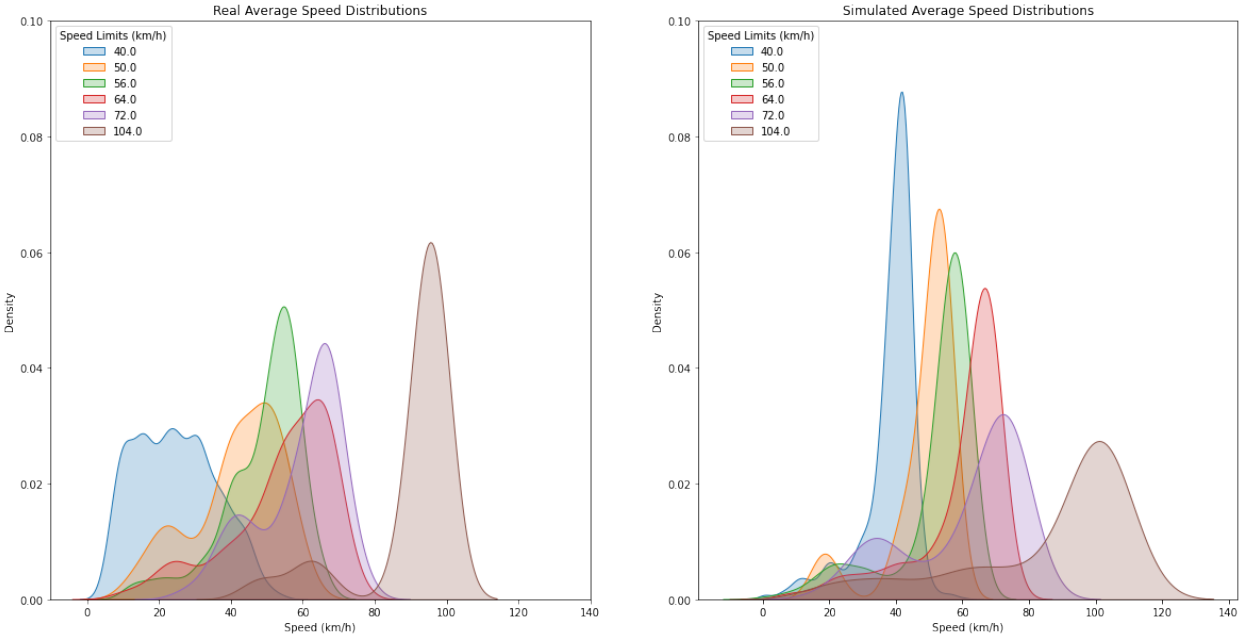

FIGURE 9 Kernel density estimation (KDE) plot of real (on the left) versus simulated (on the right) speeds at each road section. Each distribution is grouped by the speed limit on the road where the speed was observed. The simulation is able to capture a majority of trends seen in real average speed distributions. The simulation better performs on highway data than local road data.

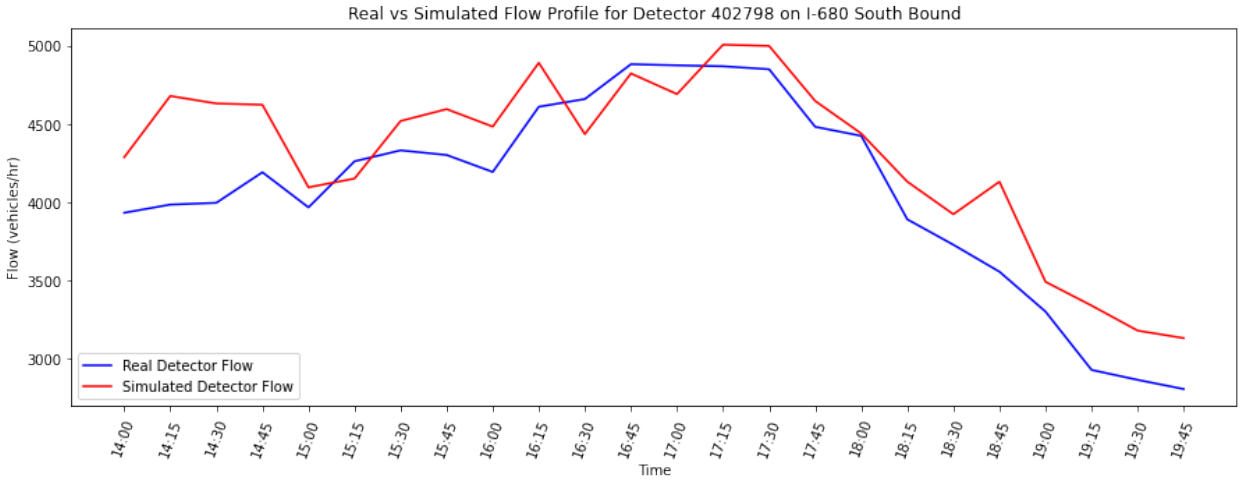

FIGURE 10 Flow profile for a detector on the I-680 South corridor. The trends of observed flow in real and simulated detectors align with each other.

The authors divided up validation of location-specific metrics into two processes. The first process is to visualize a time-series of each metric observed in each detector, like Figure 10. The trend and values observed at each timestep help determine the accuracy of each metric at each critical geographic point at the granular level.

Then, a time-space diagram is used to validate the macroscopic spatio-temporal relations across detectors in one corridor, like Figure 11. Time-space graphs describe the relationship between the location of vehicles in a traffic stream and the time as the vehicles progress along the highway (*63*). Note that on or off-ramp detectors are not included when creating a time-space diagram of detectors on a highway.

Analysis

After successful validation, the microsimulation can be used for traffic analysis study. The study can help understand the current traffic in a city or predict the impact of a "what-if" scenario. Note that microsimulation can only be used to study effects that do not require changing the input demand data. Changing the demand data will result in comparisons of two different traffic simulation models.

Some examples of scenario changes include implementing new traffic-calming strategies or changes in routing behaviors or driving behaviors. Motivating examples of scenarios that can be analyzed are as follows:

- Changing traffic signal timing plans (*3*).
- Changing speed limit or adding speed bumps (*3*).
- Adding turn and/or access restrictions (*3*).
- Understand the impact of increase in usage of Navigational apps on traffic (*64–66*)
- Changing cost function that a portion of the drivers minimizes in their routing choice to understand the impact of eco-routing adoption (*67, 68*).
- Changing the type of some vehicles to study the impact of mixed-autonomy in traffic (*69*).

Table 1 enumerates some of the metrics that can be used to analyze the impact of such

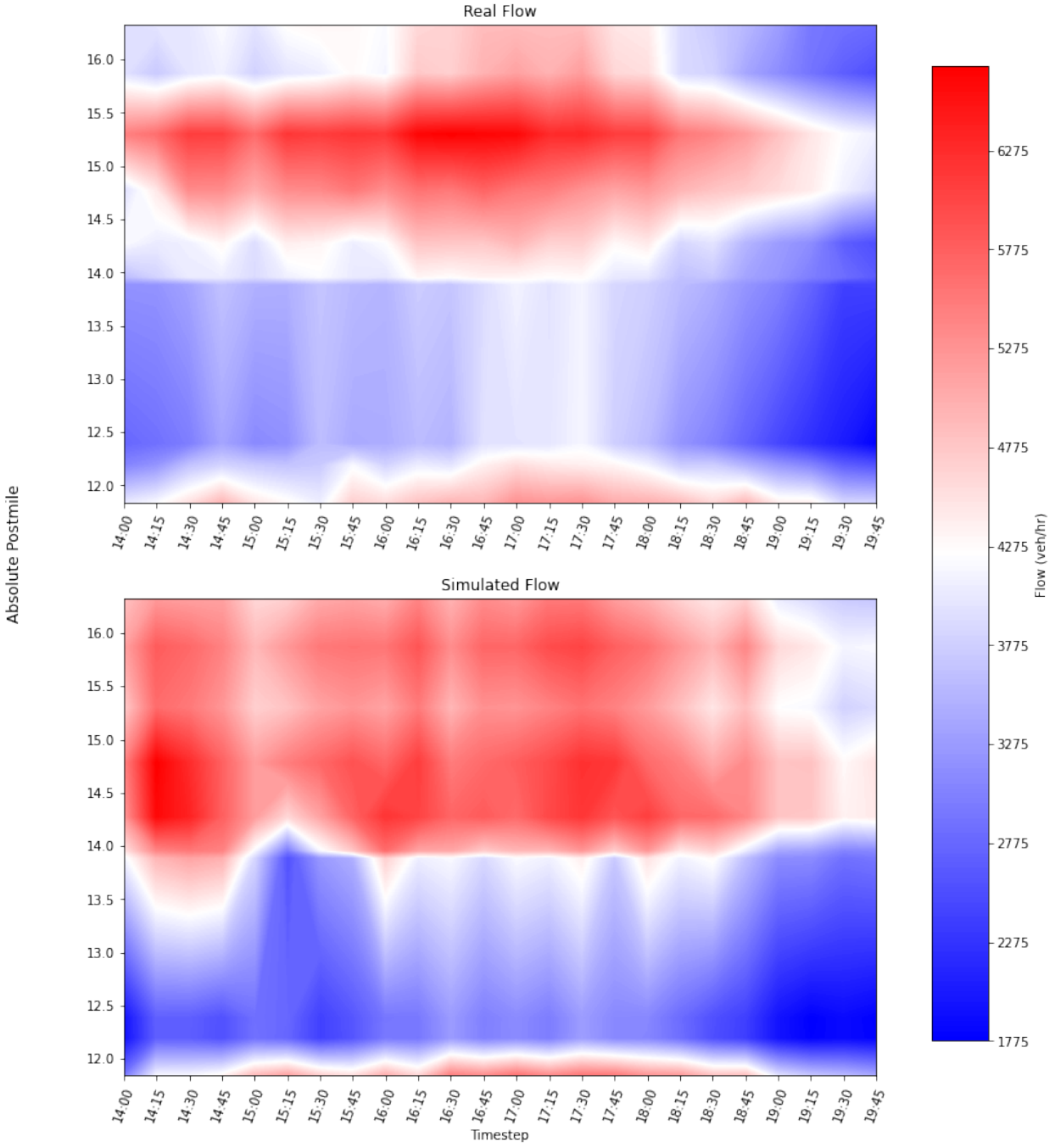

FIGURE 11 Time-space diagram of real versus simulated flow in I-680 South. Patterns of congestion are similar across real and simulated plots.

Type	Output Statistic (Units)	Granularity of Measurement
Transportation Effectiveness	Flow (veh/h)	Network, Centroid, Road, Lane, Turn, Detector
	Density (veh/km)	Network, Centroid, Road, Lane, Detector
	Travel Time (s)	Network, Centroid, Road, Lane, Turn
	Delay Time (s)	Network, Centroid, Road, Lane, Turn
	Stop Time (sec/km)	Network, Centroid, Road, Lane, Turn
	Mean Speed (km/h)	Network, Centroid, Road, Lane, Turn
	Number of Vehicles (veh)	Network, Centroid, Road, Lane, Turn
	Mean Virtual Queue (veh)	Network, Centroid, Road, Lane, Turn
	Number of Stops per Vehicle (#/veh)	Network, Centroid, Road
	Total Distance Traveled (km)	Network, Centroid
	Instantaneous Relative Gap	Network
	Experienced Relative Gap	Network
	Total Number of Lane Changes	Road
	Total Number of Missed Turns	Turn
	Total Green Light Time (s)	Turn
	Total Red Light Time (s)	Turn
Economic	Total Fuel Consumption (L)	Network, Centroid, Road, Turn
	Total Battery Consumption (kWh)	Network, Centroid, Road, Turn
Social & Environmental	Pollutant Weight (kg)	Network, Centroid, Road, Turn
	CO2 Emission (g/km)	Network, Road
	NO Emission (g/km)	Network, Road
	VOC Emission (g/km)	Network, Road
	Particulate Emission (g/km)	Network, Road
	Accessibility	Can be directly assessed through GUI

TABLE 1 Examples of metrics that can be used to analyze scenario changes. Relative Gap refers to the comparison between the current assignment solution to the ideal shortest-route time for all O-D pairs and all departure intervals (*70*). The granularity of measurement indicates at which scales the output statistic is computed by Aimsun. "Network" refers to the entire traffic network, "Centroid" each OD centroid pair, "Road" each road section in the network, "Lane" each lane in the road section, "Turn" each turn section of road intersections, and "Detector" each detector on road sections.

scenarios, alongside with the granularity of measurement.

CONCLUSION

Through the development of a traffic microsimulation of the San Jose Mission district in Fremont, CA, the authors designed and shared a reproducible process to create, calibrate and validate a large-scale microsimulation. The development of a large-scale traffic microsimulation is a tedious process that took the authors around 2,500 person-hours, and it is relevant only if case study or A/B experiments cannot be performed and if enough data is available to accurately reproduce demand data. A realistic traffic microsimulation can be used to understand current traffic and estimate policies that might impact routing or driving behaviors without changing the traffic demand (number of trips, departure times, and origins and destinations). If the simulation quality is very high compared to existing literature (flow nRMSE of 47%), it is not good enough yet to be used off-the-shelves by the city of Fremont traffic engineers.

The authors envision that future research directions about traffic microsimulation should include:

- Considering physical constraints on OD demand calibration regularization terms (like physics-informed neural network (*71*), or nuclear matrix norms (*72*)).
- Building on this work to develop a standardized validation toolbox for traffic microsimulation.
- Building on this work to develop a standardized calibration toolbox for microsimulation.
- Establishing clear use cases for when each type of traffic simulation are relevant and for when simulation are not relevant and case study, A/B experiment or simple models should be used instead of simulation.
- Continue developing large-scale simulation models and techniques to continue improving traffic at the metropolitan scale.

ACKNOWLEDGEMENTS

This work would not have been possible without the help of the City of Fremont, and especially Noe Veloso and Daniel Miller. This would not have been possible without the help of Jane MacFarlane that helped us set up the first OD demand data and refine them. The authors are thankful to the Aimsun team and their support team that consistently helped us to understand the software, how to interact with it through scripts, and for the postgraduate license. Thanks to the Connected Corridor people, especially Anthony Patire and Francois Dion, who shared their knowledge about creating, calibrating, and validating an Aimsun microsimulation; they could have written this article five years ago. Thanks to the France Berkeley Funds that funded the software used in this work. A. Bagabaldo is thankful for the Philippines CHED, DOST-SEI and Mapua University for his PhD scholarship. Thanks to the labmates that worked on understanding the impact of routing changes on traffic, especially the authors are grateful to Alex Keimer, Jessica Lazarus, Yashar Farid, Zhe Fu, Bingyu Zhao, Tania Veravelli, Pavan Yadevalli, Jiayi Li, Shuxia Tang, Arnaud De Guilhermier, Solene Olivier, Isabelle Zhou, and Henri Bataille. Finally, the authors wish to deeply thank all the team members that have been involved in this project: Anson Tiong, Daniel Macuga, Daniel Zhang, Edson Romero, Jasper Lee, Jiayi Li, Jinheng Xu, Jon Davis, Jose A La Torre, Lauren Zhou, Mengze Zhu, Michal Takac, Michael Zhang, Prakash Srivastava, Roham Ghotbi, Sayan Das, Shuli Yang, Trevor Wu, Xuan Su, Yanda Li, and Zixuan Yang.

REFERENCES

1. Lomax, T., D. Schrank, B. Eisele, et al., *2021 urban mobility report*. Texas Transportation Institute, 2021.
2. *World Energy Outlook 2018*. World Energy Outlook, OECD, 2018.
3. The Fremont Mobility Task Force, *Fremont Mobility Action Plan*, 2019.
4. John Cichowski for North Jersey, *How Leonia found smooth sailing with its last-ditch traffic fix to stop bridge commuters*, 2018.
5. Jeremy Walsh for Pleasanton Weekly, *City OKs partial street closure to curtail cut-through traffic*, 2018.
6. McElwee, M., B. Zhao, and K. Soga, Real-time Analysis of City Scale Transportation Networks in New Orleans Metropolitan Area using an Agent Based Model Approach. *MATEC Web of Conferences*, Vol. 271, No. 8, 2019, p. 06007.
7. Wegener, M., Overview of Land Use Transport Models, 2004, pp. 127–146.
8. Bhat, C. R. and F. S. Koppelman, Activity-Based Modeling of Travel Demand. In *Handbook of Transportation Science*, Kluwer Academic Publishers, Boston, 2006, pp. 39–65.
9. Shafiei, S., Z. Gu, and M. Saberi, Calibration and validation of a simulation-based dynamic traffic assignment model for a large-scale congested network. *Simulation Modelling Practice and Theory*, Vol. 86, 2018, pp. 169–186.
10. Chao, Q., H. Bi, W. Li, T. Mao, Z. Wang, M. C. Lin, and Z. Deng, A Survey on Visual Traffic Simulation: Models, Evaluations, and Applications in Autonomous Driving. *Computer Graphics Forum*, Vol. 39, No. 1, 2020, pp. 287–308.
11. Persula, M., Simulation of Traffic Systems - An Overview. *Journal of Geographic Information and Decision Analysis*, Vol. 3, No. 1, 1999, pp. 1–8.
12. Antoniou, C., J. Barcelo, M. Brackstone, H. B. Celikoglu, B. Ciuffo, V. Punzo, P. Sykes, T. Toledo, P. Vortisch, and P. Wagner, *Fundamentals of Traffic Simulation*, Vol. 145 of *International Series in Operations Research & Management Science*. Springer New York, New York, NY, 2010.
13. Lieberman, E. and A. K. Rathi, Traffic simulation. *Federal Highway Administration*, 1992.
14. of Transportation Federal Highway Administration, U. D., *Traffic Analysis Toolbox Volume III: Guidelines for Applying Traffic Microsimulation Modeling Software 2019 Update to the 2004 Version*, 2019.
15. National Academies of Sciences, Engineering, and Medicine and others, Travel demand forecasting: Parameters and techniques, 2012.
16. Maheshwary, P., K. Bhattacharyya, B. Maitra, and M. Boltze, A methodology for calibration of traffic micro-simulator for urban heterogeneous traffic operations. *Journal of Traffic and Transportation Engineering (English Edition)*, Vol. 7, No. 4, 2020, pp. 507–519.
17. Bert, E., A. Torday, and A. Dumont, Calibration of Urban Network Microsimulation Models. *Proc., 5th Swiss Transport Research Conf., Ascona, Switzerland*, , No. January, 2005.
18. Hollander, Y. and R. Liu, The principles of calibrating traffic microsimulation models. *Transportation*, Vol. 35, No. 3, 2008, pp. 347–362.
19. Otković, I. I., A. Deluka-Tibljaš, and S. Šurdonja, Validation of the calibration methodology of the micro-simulation traffic model. *Transportation Research Procedia*, Vol. 45, No. 2019, 2020, pp. 684–691.

20. Tawfeek, M. H., M. El Esawey, K. El-Araby, and H. Abdel-Latif, Calibration and validation of micro-simulation models using measurable variables. *12th International Transportation Specialty Conference 2018, Held as Part of the Canadian Society for Civil Engineering Annual Conference 2018*, , No. 6, 2019, pp. 12–22.
21. Brockfeld, E., R. D. Kühne, and P. Wagner, Calibration and Validation of Microscopic Traffic Flow Models. *Transportation Research Record*, Vol. 1876, No. 1, 2004, pp. 62–70.
22. Gehrke, S. R., A. Felix, and T. Reardon, Fare choices: A survey of ride-hailing passengers in metro Boston. *Metropolitan Area Planning Council*, 2018, p. 19.
23. Zheng, H., Y.-J. Son, Y.-C. Chiu, L. Head, Y. Feng, H. Xi, S. Kim, M. Hickman, et al., A primer for agent-based simulation and modeling in transportation applications, 2013.
24. Waddell, P., UrbanSim: Modeling urban development for land use, transportation, and environmental planning. *Journal of the American planning association*, Vol. 68, No. 3, 2002, pp. 297–314.
25. San Francisco County Transportation Authority and Cambridge Systematics Inc., San Francisco Travel Demand Forecasting Model Development: Executive Summary. Vol. 29, 2002.
26. McNally, M. G., The four-step model. In *Handbook of transport modelling*, Emerald Group Publishing Limited, 2007.
27. Balmer, M., M. Rieser, K. Meister, D. Charypar, N. Lefebvre, and K. Nagel, MATSim-T: Architecture and simulation times. In *Multi-agent systems for traffic and transportation engineering*, IGI Global, 2009, pp. 57–78.
28. Lu, H. Q. and P. Nimbole, Intro to TransCAD GIS. *Model Research and Development Unit Transportation Planning Branch*, 2002.
29. Pinjari, A., N. Eluru, S. Srinivasan, J. Y. Guo, R. Copperman, I. N. Sener, and C. R. Bhat, Cemdap: Modeling and microsimulation frameworks, software development, and verification. In *Proceedings of the transportation research board 87th annual meeting, Washington DC*, 2008.
30. Bae, S., C. Sheppard, A. Campbell, R. Waraich, S. Feygin, Z. Bilal, M. Asif, D. Aria, D. Serdyuk, A. Balayan, et al., *Behavior, Energy, Autonomy, Mobility Modeling Framework*. 7 Summits IT AG LTD, Zurich, Zurich; Skylite Networks, Fremont, CA; Lawrence . . . , 2019.
31. Galli, E., L. Cuellar, S. Eidenbenz, M. Ewers, S. Mniszewski, and C. Teuscher, ActivitySim: large-scale agent-based activity generation for infrastructure simulation. In *Proceedings of the 2009 spring simulation multiconference*, 2009, pp. 1–9.
32. Chiu, Y.-C., J. Bottom, M. Mahut, A. Paz, R. Balakrishna, S. Waller, and J. Hicks, Dynamic traffic assignment: A primer (transportation research circular e-c153), 2011.
33. U.S. Department of Transportation Federal Highway Administration, *Guidebook on the Utilization of Dynamic Traffic Assignment in Modeling*, 2020.
34. Patriksson, M., *The traffic assignment problem: models and methods*. Courier Dover Publications, 2015.
35. Transport Simulation Software, *Aimsun Next 22*, 2022.
36. Auld, J., M. Hope, H. Ley, V. Sokolov, B. Xu, and K. Zhang, POLARIS: Agent-based modeling framework development and implementation for integrated travel demand and network and operations simulations. *Transportation Research Part C: Emerging Technologies*, Vol. 64, 2016, pp. 101–116.

37. Behrisch, M., 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*, ThinkMind, 2011.
38. Fellendorf, M. and P. Vortisch, Microscopic traffic flow simulator VISSIM. In *Fundamentals of traffic simulation*, Springer, 2010, pp. 63–93.
39. Cameron, G. D. and G. I. Duncan, PARAMICS—Parallel microscopic simulation of road traffic. *The Journal of Supercomputing*, Vol. 10, No. 1, 1996, pp. 25–53.
40. Cabannes, T., M. Laurière, J. Perolat, R. Marinier, S. Girgin, S. Perrin, O. Pietquin, A. M. Bayen, E. Goubault, and R. Elie, Solving N-Player Dynamic Routing Games with Congestion: A Mean-Field Approach. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2022, pp. 1557–1559.
41. Xu, Y. and R. Goodacre, On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of analysis and testing*, Vol. 2, No. 3, 2018, pp. 249–262.
42. Haklay, M. and P. Weber, Openstreetmap: User-generated street maps. *IEEE Pervasive computing*, Vol. 7, No. 4, 2008, pp. 12–18.
43. Booth, B., A. Mitchell, et al., *Getting started with ArcGIS*, 2001.
44. Friedrich, M. and M. Galster, Methods for generating connectors in transport planning models. *Transportation Research Record*, , No. 2132, 2009, pp. 133–142.
45. Khani, A., E. Sall, L. Zorn, and M. Hickman, *Integration of the FAST-TrIPs person-based dynamic transit assignment model, the SF-CHAMP regional, activity-based travel demand model, and san francisco's citywide dynamic traffic assignment model*, 2013.
46. Chen, C., K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, Freeway performance measurement system: mining loop detector data. *Transportation Research Record*, Vol. 1748, No. 1, 2001, pp. 96–102.
47. Gordon, J. B., H. N. Koutsopoulos, and N. H. Wilson, Estimation of population origin–interchange–destination flows on multimodal transit networks. *Transportation Research Part C: Emerging Technologies*, Vol. 90, 2018, pp. 350–365.
48. Bell, M. G., The estimation of origin-destination matrices by constrained generalised least squares. *Transportation Research Part B: Methodological*, Vol. 25, No. 1, 1991, pp. 13–22.
49. Belitskii, G. et al., *Matrix norms and their applications*, Vol. 36. Birkhäuser, 2013.
50. Menon, A. K., C. Cai, W. Wang, T. Wen, and F. Chen, Fine-grained OD estimation with automated zoning and sparsity regularisation. *Transportation Research Part B: Methodological*, Vol. 80, 2015, pp. 150–172.
51. Draper, N. R. and H. Smith, *Applied regression analysis*. John Wiley, third edition ed., 1998.
52. Cascetta, E., A. Nuzzolo, F. Russo, and A. Vitetta, A modified logit route choice model overcoming path overlapping problems. Specification and some calibration results for interurban networks. In *Transportation and Traffic Theory. Proceedings of The 13th International Symposium On Transportation And Traffic Theory, Lyon, France, 24-26 July 1996*, 1996.
53. Rastrigin, L., Random search as a method for optimization and adaptation. In *Stochastic Optimization*, Springer, 1986, pp. 534–544.

54. Virtanen, P., R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, Vol. 17, 2020, pp. 261–272.
55. LeCun, Y., Y. Bengio, and G. Hinton, Deep learning. *nature*, Vol. 521, No. 7553, 2015, pp. 436–444.
56. Mirjalili, S., Genetic algorithm. In *Evolutionary algorithms and neural networks*, Springer, 2019, pp. 43–55.
57. Huntington, D. and C. Lyrintzis, Improvements to and limitations of Latin hypercube sampling. *Probabilistic engineering mechanics*, Vol. 13, No. 4, 1998, pp. 245–253.
58. Deb, K. and H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE transactions on evolutionary computation*, Vol. 18, No. 4, 2013, pp. 577–601.
59. Fortin, F.-A., F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, DEAP: Evolutionary Algorithms Made Easy. *Journal of Machine Learning Research*, Vol. 13, 2012, pp. 2171–2175.
60. Board, T. E. C. T. R., *Transportation Benefit-Cost Analysis*, 2022.
61. Arora, N., Y.-f. Chen, S. Ganapathy, Y. Li, Z. Lin, C. Osorio, A. Tomkins, and I. Tsogsuren, An Efficient Simulation-Based Travel Demand Calibration Algorithm for Large-Scale Metropolitan Traffic Models. *arXiv*, 2021.
62. Chan, C., A. Kuncheria, B. Zhao, T. Cabannes, A. Keimer, B. Wang, A. Bayen, and J. Macfarlane, Quasi-Dynamic Traffic Assignment using High Performance Computing. *Open Access Publications from the University of California*, 2021.
63. Garber, N. and L. Hoel, *Fundamental Principles of Traffic Flow*, Cengage Learning, p. 213–214. 4th ed., 2009.
64. Cabannes, T., M. A. S. Vincentelli, A. Sundt, H. Signargout, E. Porter, V. Fighiera, J. Ugirumurera, and A. M. Bayen, *The Impact of GPS-Enabled Shortest Path Routing on Mobility: A Game Theoretic Approach*, 2018.
65. Cabannes, T., F. Shyu, E. Porter, S. Yao, Y. Wang, M. A. S. Vincentelli, S. Hinardi, M. Zhao, and A. M. Bayen, Measuring regret in routing: assessing the impact of increased app usage. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 2589–2594.
66. Cabannes, T., M. Sangiovanni, A. Keimer, and A. M. Bayen, Regrets in routing networks: Measuring the impact of routing apps in traffic. *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, Vol. 5, No. 2, 2019, pp. 1–19.
67. Arora, N., T. Cabannes, S. Ganapathy, Y. Li, P. McAfee, M. Nunkesser, C. Osorio, A. Tomkins, and I. Tsogsuren, Quantifying the sustainability impact of Google Maps: A case study of Salt Lake City. *arXiv preprint arXiv:2111.03426*, 2021.
68. Google Maps, *Google Maps Eco-Friendly Routing: How it works*, 2021.

69. Wu, C., A. Kreidieh, K. Parvate, E. Vinitsky, and A. M. Bayen, Flow: Architecture and benchmarking for reinforcement learning in traffic control. *arXiv preprint arXiv:1710.05465*, Vol. 10, 2017.
70. *Guidebook on the Utilization of Dynamic Traffic Assignment in Modeling*, 2020.
71. Shi, R., Z. Mo, and X. Di, Physics-informed deep learning for traffic state estimation: A hybrid paradigm informed by second-order traffic models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, Vol. 35, pp. 540–547.
72. Candès, E. J., X. Li, Y. Ma, and J. Wright, Robust principal component analysis? *Journal of the ACM (JACM)*, Vol. 58, No. 3, 2011, pp. 1–37.