Transcription Research Record
2020, Vol. XX(X) 1–21

DOI: 10.1177/ToBeAssigned
journals.sagepub.com/home/trr
SAGE

Developing An Automated Microscopic Traffic Simulation Scenario Generation Tool

Guanhao Xu[1], Abhilasha Saroj[1], Chieh (Ross) Wang[1], and Yunli Shao[2]

Abstract

Traffic simulation is an effective tool for urban planners, traffic engineers, and researchers to study traffic. In particular, microscopic traffic simulation, which simulates individual vehicles' movements within a transportation network, has demonstrated its importance in analyzing and managing transportation systems. However, integrating data from various sources, generating traffic scenarios, and importing information into traffic simulators to conduct microscopic simulations have always been a challenge. This paper presents a solution to overcome this challenge: RealTwin, a comprehensive tool for automated scenario generation for microscopic traffic simulation. Following a streamlined scenario generation and calibration workflow, RealTwin effectively bridges gaps between traffic data from various sources and traffic simulators, making microscopic traffic simulation more accessible for researchers and engineers across various levels of expertise. Using RealTwin to generate a real-world traffic scenario in SUMO, VISSIM, and AIMSUN, RealTwin's ability is demonstrated in the construction of realistic and consistent traffic scenarios in different simulators. Furthermore, this paper introduces and illustrates RealTwin's capability for technology (e.g., autonomous vehicle) scenario generation. This feature can contribute to more comprehensive microscopic simulations, facilitating the analysis of potential effects of various technological innovations on mobility, energy efficiency, and safety. Finally, RealTwin is used to calibrate a simulation in SUMO. The calibration module enhances RealTwin's ability to generate consistent simulations across different platforms and more realistic simulations that reflect real-world traffic operations.

Keywords

microscopic traffic simulation, scenario generation, simulation calibration, autonomous vehicle

Traffic simulation is an effective tool for urban planners, traffic engineers, and researchers to study traffic. In particular, microscopic traffic simulation, which provide high-fidelity modeling by simulating the movements of individual vehicles within a transportation network, has demonstrated its importance in analyzing and managing transportation systems. By modeling each element of traffic separately and considering how they interact with each other, microscopic traffic simulation enables a realistic representation of the traffic system (*1*–*3*).

Microscopic traffic simulators are computer-based tools that provide a user-friendly platform to develop microscopic traffic simulation models to study attributes of transportation entities or systems. Numerous traffic simulators are available today, each offering different features. Choosing a particular tool depends heavily on the specific requirements of the project (*4*). Examples of such tools include Simulation of Urban Mobility (SUMO) (*5*), VISSIM (*6*), SimTraffic (*7*), TransModeler (*8*), and AIMSUN (*9*). These traffic simulators differ from each other in terms of user input requirements, driving behavior models (e.g., car-following model and lane-changing model), simulation output, and other parameters (*10*).

To carry out a traffic simulation in a traffic simulator, a traffic scenario needs to be constructed first. Some of the most essential elements of a traffic scenario include network (e.g., geometry, road type, and lane configuration), traffic demand (e.g., trips between origin and destination [OD]), and traffic infrastructure (e.g., traffic signals) (*5*). These components must be well defined to create a realistic traffic scenario.

[1]Buildings and Transportation Science Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
[2]School of Environmental, Civil, Agricultural and Mechanical Engineering, University of Georgia, Athens, GA 30602, USA

Corresponding author:
Guanhao Xu, xug1@ornl.gov

However, such process is often not straightforward because these elements come from different data sources, thus making the integration of data from multiple sources and their post-processing to generate a traffic scenario a challenging and time-consuming task (*11*–*13*). Even after data integration is complete, importing them into the traffic simulator poses another set of challenges. On one hand, different simulators may require specific formats or data structures, so that converting raw data into the format supported by the simulator may take considerable effort (*10, 14, 15*). On the other hand, an element in a traffic scenario may not have a corresponding direct input in a traffic simulator. For example, having 50% of Level 2 (partial driving automation defined in the SAE J3016 standard (*16*)) autonomous vehicles in a traffic scenario is not a direct input into any simulator but may affect other elements input into the simulator, such as road capacity and car-following behavior. The conversion of traffic scenario elements into inputs that can be taken by different simulators can dramatically increase the complexity of the process. Due to these complexities, building a traffic simulation for a simple intersection with traffic lights can take 30 minutes, while developing complex simulations for large regions can take weeks or even months (*17*).

Furthermore, once a traffic scenario is constructed and simulated in a simulation platform, it is challenging for other research teams to reproduce or extend the same scenario, say in a different simulator, for analyzing different technologies. This makes it difficult to cross-validate the simulation results in different simulators, weakening the conclusions drawn from the simulation. Besides, consistent and comparable simulation (e.g. similar vehicle composition, traffic and control characteristics, similar vehicle behavior, etc.) can be particularly important for energy analysis; lack of such consistency and transferability among simulation tools prevents researchers and transportation and energy agencies to gain a critical understanding of the potential positive and/or negative impacts of emerging mobility technologies (e.g., connected and autonomous vehicles.)

The complexities of scenario generation for microscopic traffic simulation, along with the challenges of ensuring scenario consistency across different simulators, have given rise to a great demand for tools that automate and streamline the scenario generation process and construct scenarios that lead to realistic and comparable simulation scenarios in different simulators.Therefore, this work aims to introduce a tool, i.e., **RealTwin**, which addresses the aforementioned challenges in the generation of microscopic traffic simulation scenarios.

The key contribution of this paper is to demonstrate a tool that is enabled by several key elements proposed by the authors:

- a highly automated scenario generation and calibration workflow that bridges the gaps between data from different sources and simulators. This capability not only significantly saves the time and effort needed for developing and calibrating microscopic traffic simulations, but also makes microscopic traffic simulations more accessible to researchers and engineers with different levels of expertise. Users with limited simulation experience can access Real-Twin's basic functionality by simply providing the required inputs to generate simulation-ready scenarios while advanced users with deeper simulation knowledge can directly call Real-Twin's scripting functions to refine scenarios or customize the simulation setup;
- a database and pipeline to enable the generation of traffic scenarios to study the presence of emerging technologies, with current support for AV scenario generation in SUMO and planned extensions to VISSIM, AIMSUN, and other emerging technologies such as CAVs and EVs. This feature, once fully developed and extended to different simulation platforms, can contribute to the development of more comprehensive microscopic traffic simulation scenarios that incorporate the presence of emerging technologies to examine their impacts on mobility, energy efficiency, and safety;
- a scenario definition and generation workflow to generate comparable simulation scenarios across different microscopic traffic simulators. Real-Twin currently supports SUMO, VISSIM, and AIMSUN, providing users the ability to conduct benchmarking and cross-validation that are crucial for ensuring the reliability and reproducibility of simulation results.

The rest of the paper is organized as follows. The next section provides a review of the literature on microscopic traffic simulations. This is followed by a detailed explanation of the RealTwin workflow and functions. Next, a case study is conducted to illustrate the use of RealTwin and its capability for application modeling and simulation calibration. Finally, the conclusion summarizes the main contributions of the RealTwin tool to the field of traffic simulation and discusses future research directions.

Literature Review on Microscopic Traffic Simulation

Microscopic Traffic Simulation

RealTwin currently focuses on the scenario generation for microscopic traffic simulations. Microscopic traffic simulation captures the dynamics of all vehicles and their interactions in intricate detail. Typically, each vehicle is depicted using driver behavior models (e.g., car-following model, lane-changing model) with various parameters—such as desired speed, acceleration capabilities, driver aggressiveness, and reaction times (*18*). Owing to the high level of

detail, microscopic traffic simulations are ideal for traffic operation analysis enabling the evaluation of complex traffic dynamics (*19*–*22*). In safety studies, microscopic traffic simulations provide critical insights into near-misses, collisions, and conflict points, allowing for the assessment of advanced safety interventions and environmental impacts on driver behavior (*23*–*25*). Moreover, microscopic traffic simulations also support testing traffic and vehicle control technologies like traffic signal control (*26, 27*), Eco-Driving strategies (*28*–*30*), Connected and Automated Vehicles (CAV) (*31*–*33*), offering a controlled environment for safe and cost-effective experimentation.

Microscopic Traffic Simulation Software

Microscopic traffic simulation software provides detailed and dynamic modeling of traffic systems by simulating the individual behaviors of drivers and vehicles. Some commonly used microscopic traffic simulation software include SUMO, AIMSUN, PTV VISSIM, Corsim, Paramics, and TransModeler. Each software has distinct strengths; for example, SUMO excels in flexibility and extensibility owing to its open-source nature, allowing users to modify and expand its features to suit specific project requirements. VISSIM and AIMSUN have user-friendly interfaces and offer versatility with their ability to switch between microscopic and mesoscopic simulations, enabling efficient handling of both detailed traffic behavior studies and broader traffic pattern analyses over large areas. Currently, RealTwin supports the generation of scenarios for SUMO, AIMSUN, and VISSIM.

SUMO is an open-source, highly portable platform that supports detailed modeling of vehicular, pedestrian, and bicycle traffic. It allows for extensive customization and scripting through the use of extensible markup language (XML) documents, making it adaptable for a range of scenarios from small roads to entire cities (*34, 35*).

AIMSUN is a commercial traffic simulation software used in transportation planning and engineering, providing a single, united network representation of how people move in various modes and at various scales. It responds to the complexity or simplicity of various projects and comes with an extensive built-in toolkit that can be further enhanced and extended with Python scripts, APIs, and software development kits (*9*).

PTV VISSIM is also a widely used commercial microscopic traffic simulation tool developed by the PTV Group. It comes with built-in Wiedemann car-following models and rule-based algorithms for lateral vehicle movement (*36*). In addition, it offers a VISSIM component object model (COM) interface that allows access to the object model hierarchy, with network elements such as vehicles, links, and vehicle inputs using various programming languages such as C, C++, Python, and VB.Net (*36, 37*).

Microscopic traffic simulations in PTV VISSIM are time-step-oriented discrete event simulations, while microscopic traffic simulations in SUMO and Aimsun are discrete-time simulations. In time-step-oriented discrete event simulation, updates occur at fixed time steps, triggered by events, while in discrete-time simulation, updates are triggered only at regular, fixed time intervals.

Automated Microscopic Traffic Simulation Generation Tools

Although many tools and platforms have been developed to streamline the microscopic traffic simulation generation process, very few tools have emerged to automate it.

SUMO OpenStreetMap (OSM) Web Wizard is one of the most popular microscopic traffic simulation creation tools and has been widely used in many studies (*38*–*41*). OSM Web Wizard enables users to configure a randomized traffic demand and run and visualize the scenario in the sumo-gui (*5*). However, OSM Web Wizard has many limitations.

First, although it is excellent for quick and easy scenario creation, it struggles with generating long-period, larger-scale simulation scenarios. This is partly due to the request limitations on the OSM database side, which can result in connection issues for large areas, requiring multiple smaller requests that significantly extend the processing time. Second, OSM Web Wizard generates synthetic traffic demand using SUMO's randomTrips.py, which does not incorporate daily or periodic flow curves, reducing the representativeness of long-term traffic demand. Third, OSM Web Wizard lacks certain elements of traffic scenario; for instance, it does not allow users to set traffic signals in its automated scenario generation process, and users need to rely on separate tools like netedit for this purpose. Finally, OSM Web Wizard is designed exclusively for SUMO and does not support other simulators.

Flow is a traffic control benchmarking framework and it provides a suite of traffic control scenarios (benchmarks), tools for designing custom traffic scenarios, and integration with deep reinforcement learning and microscopic traffic simulation libraries (*42*). However, this tool is designed specifically for generating traffic control scenarios and does not provide functions that automate the ingestion of other real world data such as network and traffic demand.

Another recently available tool comprises the two packages "osm2gmns" and "CAMLite" proposed in (*43*). The "osm2gmns" package automatically processes OpenStreetMap transport networks in the General Modeling Network Specification (GMNS) format, whereas "CAMLite" supports connected and automated mobility (CAM) system modeling. However, these tools only support the creation and simulation of the transportation networks in the GMNS format.

There are also some existing tools that support automated microscopic traffic simulation calibration. For example, a

genetic algorithm tool in MATLAB was designed in (*44*) to automatically calibrate microscopic traffic simulation models based on speed–density relationships in Aimsun. Another example is the methodology proposed in (*45*) that trains an artificial neural network (ANN) to learn the behavior of the transport network of interest and automatically calibrate microscopic traffic simulations.

The crucial aspects that often need to be handled manually by users include data integration from diverse sources, filling in missing or incomplete information (e.g., traffic signal plan) to build a traffic scenario, importing the traffic scenario into microscopic traffic simulators (e.g., configuring simulator parameters to reflect the traffic scenario), and calibrating the scenarios based on real-world data (e.g., calibrate driving behavior parameter) (*5, 6, 46*). These existing tools automate only specific processes in developing traffic scenarios for microscopic traffic simulations, leaving other crucial aspects to be handled manually by users. In addition, none of these tools support the generation of scenarios ready to be simulated in multiple prevailing traffic simulators. Supporting multiple simulators is important because users may be familiar with only one simulator, and if the tools do not support that specific simulator, it becomes unusable for them. Moreover, supporting multiple simulators provides flexibility in choosing the best tool for specific research or practical needs and ensures robustness by allowing cross-validation of results. Without this support, the tool's usability is significantly restricted.

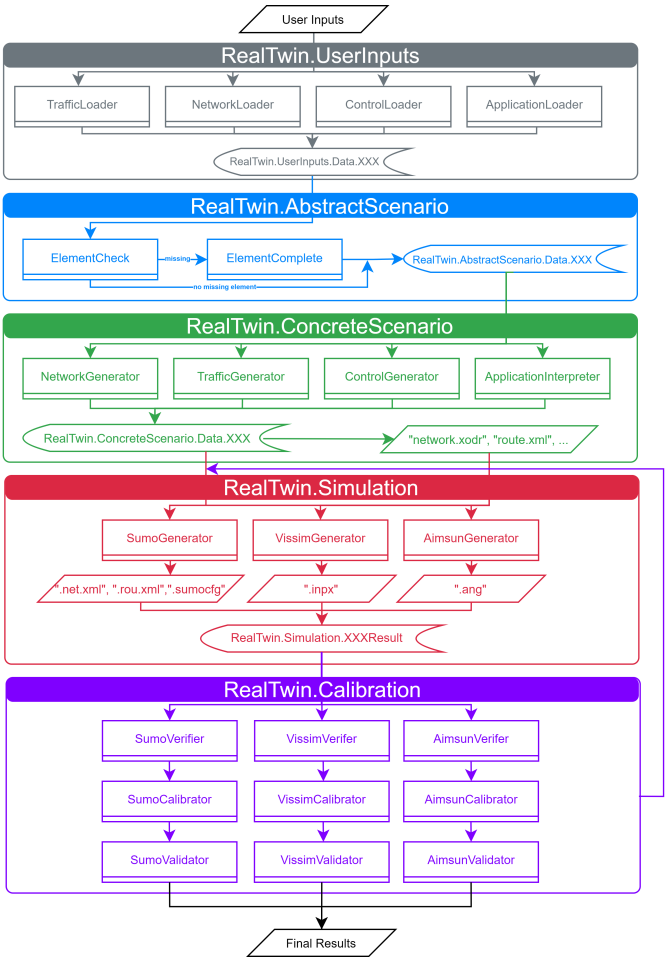


Figure 1. RealTwin workflow.

RealTwin workflow

Overview

Constructing a simulation scenario based on real-world data can be time-consuming due to complexity in the integration of various data formats and in the post-processing needed to align with specific simulation parameters. In addition, the need for specialized knowledge in traffic simulation presents a challenge for researchers who do not have a simulation background to successfully use traffic simulation for developing solutions for traffic management and urban planning. This paper aims to overcome these challenges by offering an automated way for simulation scenario generation that reduces the amount of effort required, thereby making traffic simulations accessible to a broader audience.

We present RealTwin, a comprehensive tool for automating the process of scenario generation and calibration for microscopic traffic simulation. RealTwin accommodates user-customized input, processes that input to construct abstract and concrete scenarios, imports the input to multiple simulation platforms, and calibrate the generated simulation. The RealTwin workflow is shown in Figure 1 and is discussed in detail in subsequent sections. This workflow is designed to be modular, allowing for the extension and modification of different modules to meet diverse research and simulation needs.

RealTwin.UserInputs: Module for Ingesting User Inputs

Element of User Input

The `RealTwin.UserInputs` module serves as the initial module of RealTwin. It contains the "IngestUserInput" function that reads user-supplied data and converts it into a format that RealTwin can understand. Once converted, these data are then stored in an intermediate variable called `RealTwin.UserInputs.Data` and passed along to the subsequent module in the workflow.

Format of User Input for Minimum Requirement

The RealTwin user input interface is a YAML file that stores scenario parameters or data file paths. YAML, short for yet another markup language, is a lightweight human readable serializing language designed primarily to be easy to read and edit; therefore, it is often used to create automation processes

(47). An example of a YAML file that meets the minimum user input requirements is depicted in Figure 2.

For the "traffic" category, RealTwin currently support traffic volume and intersection turning ratio as inputs. The input for volume should be a comma-separated value (CSV) file that contains at least the name of the road, the starting and ending intervals of each measurement, and the count of vehicles in each measurement interval. The turning ratio contains the name of the intersection, the starting interval of each measurement, the ending interval of each measurement, and the vehicle count or turning percent for each turning movement in the corresponding interval. If the input for TurningRatio is missing, RealTwin can still proceed with the workflow assuming [0.8,0.1,0.1] for a [thru,left,right] approach, [0.8,0.2] for a [thru,left/right] approach, and [0.5,0.5] for a [left,right] approach. However, TurningRatio input is highly recommended. Currently, RealTwin can directly accept output files from GRIDSMART (48), an innovative smart camera system that provides intersection volume and turning ratios at fixed intervals (e.g., 15 minutes). Users may also transform their own data into this format to be ingested by our tool. In future expansions, we plan to support additional data sources and formats, including origin-destination data and vehicle trajectory data, to further enhance RealTwin's versatility.

For the "network" category, the minimum requirement is the longitudes and latitudes of a set of vertices of the polygon that bounds the network. Using these coordinates, RealTwin queries OpenStreetMap (49), downloads the .osm network bounds by the polygon, and converts it to an OpenDRIVE network. The ASAM OpenDRIVE format, with the file extension ".xodr", provides a common standard for describing road networks using the Extensible Markup Language (XML) syntax (50). It can be easily imported to many popular traffic simulation software (e.g., SUMO, VISSIM, and AIMSUN). Therefore, the OpenDRIVE network is used as the network file for scenario generation.

For the "IntersectionControl" in the "control" category, RealTwin currently supports processing the universal traffic data format (UTDF), an open standard data specification for traffic signals and traffic-related data for intersections promoted by Cubic Trnasportation Systems' Trafficware, the developer of Synchro (51). A detailed description of UTDF data can be found in (52).

Format of User Input for ApplicationLoader—Function for Technology Scenario Generation

The RealTwin tool is also designed to include technology applications that aim to enable users to generate simulation scenario for the selected technology. With the emergence of AV and connected AV (CAV) (53) technologies, a simulation scenario often needs to be further tailored to be able to study the effects of these applications. In this work, we focus on illustrating the capabilities of the RealTwin tool and demonstrate the capability of generating AV scenarios as an example first step. Based on technology-specific parameters provided by users, the developed AV technology scenario extension generates a simulation scenario with AV in the fleet. The generated AV scenario consists of a vehicle type for AV with corresponding driving behavior model parameters to represent it. Figure 3 shows the part of parameters of user input for the AV technology scenario in the user input interface YAML.

Example Application—AV Scenario

The key user input parameters for a technology scenario are technology scenario identifier name, vehicle types preferred in the fleet, penetration level of the technology, and car-following model parameters for the vehicle types. Currently, RealTwin supports the scenario generation for AV, but we plan to expand the capability to enable scenario generation for CAVs and electric vehicles (EVs) in the near future.

AV User Input: If driver behavior parameters are not provided by the user, as in the case in Figure 3, RealTwin generates driver behavior parameters based on values suggested in the literature for the corresponding car-following model to represent AV driving behavior described under *AV Parameter Database* next. The default value for "VehicleTypes" is set to "Human" and "AVnormal" to represent human-driven vehicles and AV vehicle types. In addition, "carFollowingModelforVehicleTypes" is set to the user-defined car following model. It is to be noted that for simulation software that do not provide access to multiple car following model the default car following model will be used even if the user provides a different user-defined car following model for the vehicle type. For "PenetrationLevelPercent," the default value is set to 100 percent. This user interface design for the AV application can be extended to add more vehicle types in the future, such as "AVSafe" or "AVAggressive" to represent safe or aggressive driving behaviors of AVs. To ensure the tool to quickly identify inputs needed, the naming of parameters in YAML will be consistent with those of each simulator, and detailed documentation of RealTwin will be provided.

AV Parameter Database: We create an AV Parameter Database for RealTwin that stores the parameter values to define the AV and human driving behavior. The AV Parameter Database includes parameters for the car-following model, cooperative driving, lane-changing model, and speed distribution parameter values. It is developed based on the parameter values used in the literature to simulate AV driving behavior in the simulation software of choice.

The AV Parameter Database currently supports AV scenario generation in VISSIM and SUMO. This will be extended to AIMSUN in the future. For VISSIM, the Wiedemann 99 car-following parameter values used primarily in VISSIM to define AV driving behavior based on previous studies (54–65) are provided next. The magnitude of these values for AV in comparison to human driving behavior is provided in italicized text.

```
Traffic:
  Volume: volume.csv
  TurningRatio: turn.csv
Network:
  NetworkVertices: (longitude1, latitude1), (longitude2, latitude2),
                   (longitude3, latitude3), (longitude4, latitude4),
Control:
  IntersectionControl: control.csv
```

Figure 2. A sample YAML file that meets minimum requirements for user input.

```
applicationLoader:
  technology: AV #Mention application name
  penetrationLevelPercent: 50
  vehicleTypes: ['Human', 'AVnormal']  #Human, AVnormal, AVcautious, AVaggressive
  carFollowingModelforVehicleTypes: ['Wied 99', 'Wied 99'] #Wied99, Krauss, IDM
  Wied99Parameters:
    cc0: [[],[2]]
    cc1: [[],[]]
    cc2: [[],[]]
    cc3: [[],[]]
    cc4: [[],[]]
    cc5: [[],[]]
    cc6: [[],[]]
    cc7: [[],[]]
    cc8: [[],[]]
    cc9: [[],[]]

  KraussParameters:
    minGap: [[],[]]
    accel: [[],[]]
    decel: [[],[]]
    sigma: [[],[]]
    tau: [[],[]]
    emergencyDecel: [[],[]]
```

Figure 3. Example YAML file extension for an AV scenario.

- CC0—Standstill distance (m): 1 *(lower)*
- CC1—Headway time (s): 0.5 *(lower)*
- CC2—"Following" variation (s): 1.66 *(lower)*
- CC3—Threshold for entering the following phase (s): −10 *(lower)*
- CC4—Negative "following" threshold (m/s): −0.35 *(no change)*
- CC5—Positive "following" threshold (m/s): 0.35 *(no change)*
- CC6—Speed dependency of oscillation (1/ms): 0 *(lower)*
- CC7—Oscillation acceleration (m/s^2): 0.33 *(higher)*
- CC8—Standstill acceleration (m/s^2): 3.8 *(higher)*
- CC9—Acceleration at 80 km/hr (m/s^2): 1.8 *(higher)*

For SUMO, the AV Parameter Database contains parameter values to represent AV driving behavior for Krauss—the default car-following model in SUMO (*5*). To define human driving behavior, the default values of the car-following behavior parameter for Krauss in SUMO are used. The parameter values to define the driving behavior of an AV considered based on previous efforts (*66*–*68*), along with comparisons of their magnitude with respect to human driving behavior, are:

- minGap—Minimum gap when standing (m): 0.5 *(lower)*
- tau—The driver's desired (minimum) time headway (s): 0.4 *(lower)*
- sigma—The driver imperfection where 0 denotes perfect driving: 0 *(lower)*
- accel—The acceleration ability of vehicles of this type (in m/s^2): 3.8 *(higher)*
- decel—The deceleration ability of vehicles of this type (in m/s^2): 4.4 *(no change)*
- emergencyDecel—The maximum deceleration ability of vehicles of this type in case of emergency (in m/s^2): 9 *(no change)*

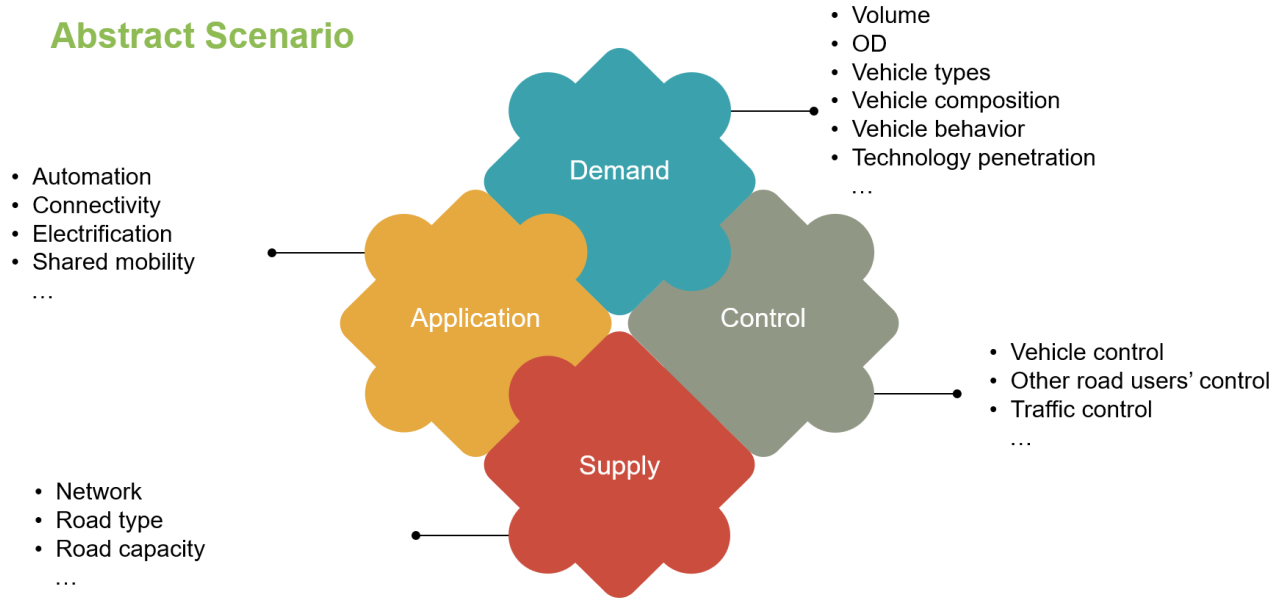


Figure 4. Abstract scenario definition.

RealTwin.AbstractScenario: Module for Generating Abstract Scenario

Abstract Scenario Definition

When researchers come up with a scenario to study a technology or application, they essentially have a definition of an abstract scenario. The abstract scenario definition can be descriptive. It is a direct transformation of user inputs into RealTwin that is composed of four sets of elements, listed in Figure 4.

Workflow of RealTwin.AbstractScenario

In the `RealTwin.AbstractScenario` module, user input data will be converted to a complete definition of an abstract scenario. Specifically, functions in the four classes—"TrafficLoader," "NetworkLoader," "ControlLoader," and "ApplicationLoader"—will transform the raw data stored in "RealTwin.UserInputs.Data" into intermediate variables that fit into each of the elements defined in the abstract scenario definition. Then, the "ElementCheck" function will check if all elements required for simulation construction are fulfilled. If any element is not provided by the user, the RealTwin.Database, which is composed of open-source datasets, will be queried to complete the abstract scenario definition via the "ElementComplete" function, as shown in Figure 5. Finally, all elements of the abstract scenario will be stored in `RealTwin.AbstractScenario.Data`.

RealTwin.ConcreteScenario: Module for Generating Concrete Scenario

Concrete Scenario Definition

In the `RealTwin.ConcreteScenario` module, an abstract scenario is converted into a concrete scenario, which describes all the critical attributes of traffic and will be used as direct inputs to different traffic simulators. The major differences between an abstract scenario and a concrete scenario are the following.

- In an abstract scenario, some elements, though critical to a simulation, cannot be used as direct inputs into simulation tools due to their lack of specific information or tangible characteristics. However, in a concrete scenario, all elements are well-defined and can be used as direct input into different simulation platforms. For instance, in a concrete scenario, cars and buses are considered as road users to be simulated. However, the concept of "road user" itself might not be a direct input that can be fed into simulation tools. Therefore, in the corresponding concrete scenario, the properties are well defined and are input into simulators (e.g., maximum acceleration/deceleration, desired speed distribution).

- In an abstract scenario, elements come from different data sources and thus are independent. However, in a concrete scenario, all elements are interconnected through the functions in the `RealTwin.ConcreteScenario` module. For example, in an abstract scenario, the variable that stores volume may contain the volume of the entire city, depending on what users provide. By contrast, in a concrete scenario, volume is assigned to each specific road within the area of interest, as detailed in the network file.

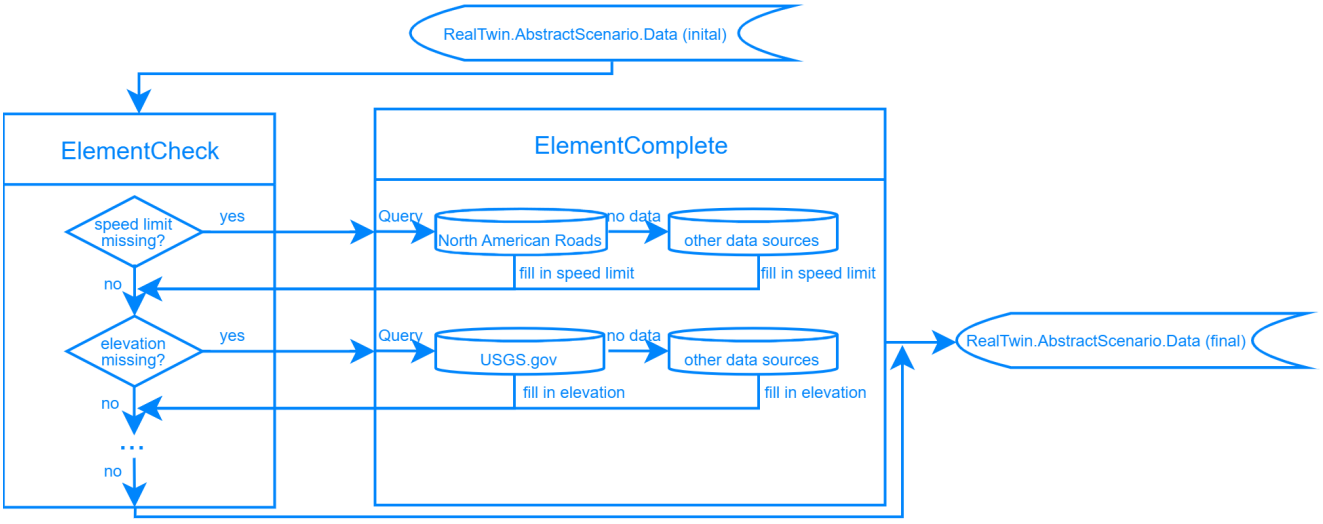


Figure 5. Workflow for filling in missing elements.

Workflow of RealTwin.ConcreteScenario

In the `RealTwin.ConcreteScenario` module, the key task is to link all elements together. This integration is realized using functions in the class "NetworkGenerator," which finalize the network geometry and properties and encapsulate them in an OpenDRIVE (.xodr) file. OpenDRIVE object identifiers, such as road IDs and junction IDs, are correlated with the corresponding IDs in each of the other abstract scenario elements by matching the latitude and longitude. This builds a connection between the network and all other elements. Next, the functions in the classes "TrafficGenerator," "ControlGenerator," and "ApplicationInterpreter" are called to finalize all remaining concrete scenario elements. These consolidated components are finally stored in `RealTwin.ConcreteScenario.Data`.

RealTwin.Simulation: Module for Generating Simulation in Different Simulators

The next phase of the RealTwin workflow is to load the concrete scenario into different simulators (e.g., SUMO, VISSIM, and AIMSUN) and create simulation runs. This process occurs as follows: The "ImportNetwork" function is first called to import the OpenDRIVE network into the simulator of choice. Then, the "MatchJunctionID," "MatchRoadID," and "MatchMovementID" functions are used to create an object ID mapping table between the OpenDRIVE network and the simulator of choice. Next, "ImportSpeedLimit" updates the road speed limit inside the simulation platform. This is followed by the generation of demand using the "ImportDemand" function. Finally, "GenerateSimulation" functions are used to conduct simulation runs with different random seeds. Table 1 shows the details of the RealTwin functions in the `RealTwin.Simulation` module.

RealTwin.Calibration: Module for Calibrating Simulations in Different Simulators

The final phase of RealTwin is to calibrate the simulation scenario generated. Calibration is an important element of microscopic traffic simulation development. It ensures model accuracy by optimizing model parameters to achieve the closest possible match between simulated outputs and real-world traffic measurements (e.g., flow, speed, and travel time) (*69*). Techniques such as Genetic Algorithms (GA), Simulated Annealing (SA), and Tabu Search (TS) are commonly used to optimize calibration, addressing the complexities of high-dimensional parameter spaces (*70, 71*). Recent advancements such as the use of vehicle trajectory data have further enhanced calibration precision (*72, 73*).

RealTwin currently has the capability of calibrating: 1) turning ratios and traffic inflows at minor intersections (with turning ratios and traffic inflows at major intersections provided by the user);2) driving behavior parameters, chosen based on a literature review (*70, 74–80*), which we identify as widely used parameters in microscopic traffic simulation calibration. Detailed examples for each calibration are provided in a case study in the next section.

Table 1. Functions in RealTwin.Simulation

Function	Purpose	Input	Output
ImportNetwork	Import the OpenDrive (`.xodr`) network to the simulator	`.xodr` network	• SUMO: `.net.xml` network • VISSIM: `.inpx` network • AIMSUN: `.ang` network
MatchJunctionID	Match junction ID from OpenDrive network with junction ID in simulator	`.xodr` network	Junction ID mapping tables
MatchRoadID	Match road ID from OpenDrive network with junction ID in simulator	`.xodr` network; junction ID mapping table	Road ID mapping tables
ImportSpeedLimit	Import speed limit	`.xodr` network; road ID mapping table	• SUMO & AIMSUN: updated speed limit • VISSIM: "Desired Speed Decision" objects created and placed at the corresponding network locations
ImportStopControl	Import intersection stop control	`.xodr` network; traffic control; junction&road ID mapping tables	• SUMO: updated junction type and edge priority • VISSIM: "StopSign" object placed at the location(s) • AIMSUN: updated warning and priorities in each node configuration
ImportSignal	Import intersection signal plan	`.xodr` network; traffic control; junction&road ID mapping tables; movement ID mapping tables	• SUMO: `.add.xml` additional file defining traffic light program • VISSIM: "Signal Controller" and "Signal Head" objects created; "Signal Heads" for corresponding "Signal Group" placed at the network location • AIMSUN: MasterControl Plan
ImportDemand	Import demand (currently support inflow + turning ratio)	Inflow; turning ratio; junction ID mapping table; road ID mapping table	• SUMO: `.rou.xml` route file • VISSIM: Network objects "Vehicle Inputs," "Vehicle Composition," and "TimeIntervals" created and set up to represent the demand for the assigned time interval • AIMSUN: "Traffic State" and "Traffic Demand" set up
GenerateSimulation	Generate simulations with different random seeds	`.xodr` network, random seeds	• SUMO: `.sumocfg` configuration file • VISSIM: Parameters for "Simulation," "Simulation Period," "Number of Runs," and "Random Seed" set up • AIMSUN: Parameters for "Scenario," "Experiment," and "Replication" set up

Workflow of RealTwin.Calibration

A detailed workflow of `RealTwin.Calibration` is shown in Figure 6. It includes three primary steps:

- **Verification:** Check if the real-world input data match with the input data given to the model, and verify if the model is responding to the inputs the same way it is expected to respond in the simulation.
- **Calibration:** Tune selected calibration parameters (e.g., turn movement ratio, car-following parameters, lane-changing parameters) so that the calibration targets are met. This process includes a comparison of simulated results and ground truth data (*81*).
- **Validation:** A verified and calibrated model is validated with ground truth data, comparing simulation-generated performance measure and ground truth measure. This performance measure is usually different from the one used to evaluate a calibration target.

Objective Function

The first objective function used in `RealTwin.Calibration` is minimizing the average Geoffrey E. Havers statistic (GEH), a widely accepted measure to evaluate discrepancies between observed and simulated traffic flows given by the equation below. We use it to calibrate turning ratios and traffic inflows in RealTwin, as it balances relative and absolute error. This is particularly important because traffic volumes can vary widely across locations, and GEH provides a robust measure without overemphasizing small differences in low-flow areas.

$$\min GEH_{avg} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\frac{2(M_i - C_i)^2}{M_i + C_i}} \tag{1}$$

where

GEH_{avg}	mean GEH
M_i	metric of observation i from simulation
C_i	actual metric of observation i
N	total number of field observations

The second objective function used in RealTwin is to minimize the mean absolute error (MAE). We use it to calibrate the driving behavior parameters in RealTwin because these parameters are often calibrated by comparing simulated travel time/speed with real world travel time/speed, which tend to be less variable compared to traffic volumes. Travel time or speed differences have a more uniform significance across scenarios, making absolute error measures like MAE appropriate. This is particularly relevant because travel times for long routes are typically used for calibration, where the differences are more consistent and meaningful across the dataset. The MAE measures the average magnitude of the errors between the observed and simulated traffic data and is given by:

$$\min MAE = \frac{1}{N}\sum_{i=1}^{N}|M_i - C_i| \tag{2}$$

where

MAE	mean absolute error
M_i	metric of observation i from simulation
C_i	actual metric of observation i
N	total number of field observations

Measures of Effectiveness

Currently, RealTwin supports link flow and travel time as measures of effectiveness (MOEs) for evaluating simulation performance from the mobility perspective. In future versions, we plan to expand the range of MOEs to include metrics that capture detailed driving behavior (e.g., lane change frequency, acceleration and deceleration profiles, speed variance), safety indicators (e.g., time-to-collision), as well as energy consumption and emissions estimates.

Calibration Target

For the simulation calibration target, `RealTwin.Calibration` uses Wisconsin Department of Transportation freeway model calibration criteria (*81*), which defines calibration acceptance targets for hourly flow and travel time. Specifically, GEH of individual link flow should be below 5 in over 85% of cases and simulated travel times should be within 15% of real-world travel times in over 85% of cases.

Calibration Algorithm

Three parameter optimization algorithms are integrated for calibration in RealTwin for users to choose from: Tabu Search, Genetic Algorithm, and Simulated Annealing. For each algorithm, default values for key parameters have been provided, which can be adjusted by the user.

- **Tabu Search (TS)** explores new solutions by moving from a solution to the solution with the best objective function in its neighborhood at each iteration until some stopping criterion has been satisfied (*70, 71, 82*). Key parameters include:
  - Tabu List Size: 100 (number of solutions to keep in the tabu list to avoid revisiting them).
  - Maximum Iterations: 30 (stopping criterion if no improvement is found).
  - Neighborhood Size: 50 (number of solutions evaluated in each iteration).
- **Genetic Algorithm (GA)** starts from a random population set and evaluates candidate solutions at

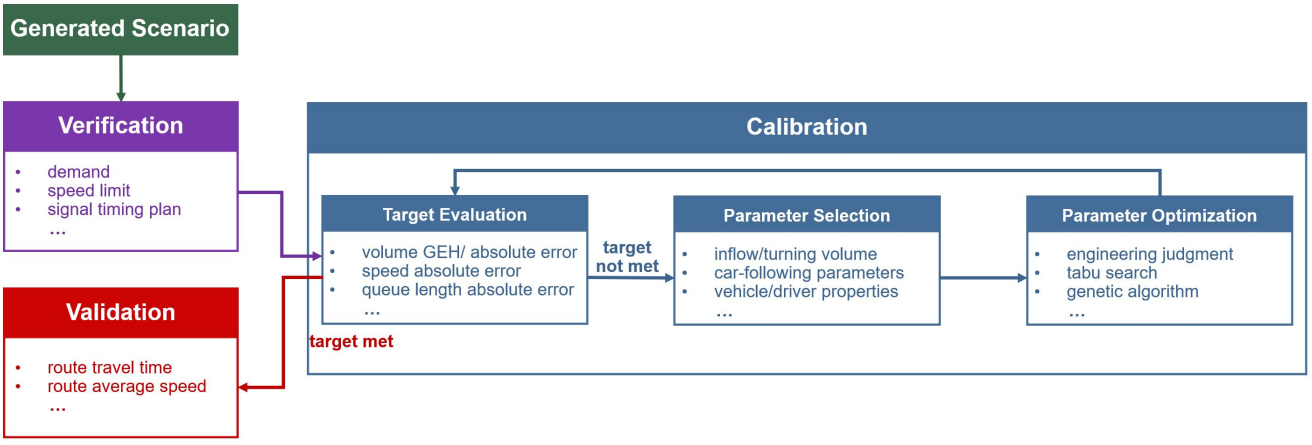


Figure 6. RealTwin calibration workflow (GEH = Geoffrey E. Havers statistic).

every generation. If stopping criteria are not met, its population is evolved by selection, crossover, and mutation (*70*, *71*, *82*). Key parameters include:

  - Population Size: 50 (number of candidate solutions in each generation).
  - Crossover Rate: 0.75 (probability of combining two solutions).
  - Mutation Rate: 0.1 (probability of mutating a solution).
  - Maximum Generations: 30 (stopping criterion if no improvement is found).

- **Simulated Annealing (SA)** is inspired by the metallurgical process of annealing, where a system explores potential solutions by initially allowing higher probabilities of accepting worse solutions to escape local optima and then gradually reducing this probability as it "cools" down (*83*, *84*). Key parameters include:

  - Initial Temperature: 100 (starting probability of accepting worse solutions).
  - Cooling Rate: 0.99 (factor by which the temperature is reduced in each iteration).
  - Maximum Iterations: 1000 (stopping criterion if no improvement is found).

RealTwin Demonstration Using Case Study

This section demonstrates the use of the developed RealTwin tool to generate a calibrated traffic microscopic simulation scenario through a case study. Specifically, the RealTwin tool was used to automatically

- develop a complete simulation including road network, traffic demand, routes, and traffic controls;
- generate a simulation scenario to study the effect of AVs in traffic fleet composition on vehicle travel times using a RealTwin application extension
- evaluate network and vehicle level metrics from the simulation results; and
- calibrate simulation parameters using real-world data.

Implementation of RealTwin for Scenario Development

In this section, a case study was conducted to demonstrate the capability of RealTwin in automating microscopic traffic simulation scenario generation in SUMO, VISSIM, and AIMSUN. This case study demonstrates automated microscopic simulation scenario generation of a 0.58-mile corridor along Shallowford Road in Chattanooga, Tennessee, USA. This network has 10 intersections: 6 intersections controlled by actuated signal controllers and 4 unsignalized intersections. The traffic simulation scenario for a real-world historic days' morning peak was automatically generated using RealTwin, given user input data for "Traffic," "Network," and "Control." This scenario generation for each simulator takes about 10-15 seconds on a system with an Intel i9-13900 CPU and an NVIDIA RTX 4090 GPU.

Ingestion of User Input

In this case study, the user input for "network" is the longitude and latitude of the four vertices of the polygon bounding this corridor, as shown in Figure 7. The corridor spans approximately 0.58 miles and includes 10 intersections (6 signalized and 4 unsignalized). The intersection traffic count data from GRIDSMART (*48*) were used as user input for both "Volume" and "TurningRatio" in the "Traffic" category. These data were collected between 8:00 a.m. and 9:00 a.m., with traffic flows on the major road reaching up to 1,371 veh/hr and minor roads ranging from 64 to 377 veh/hr.

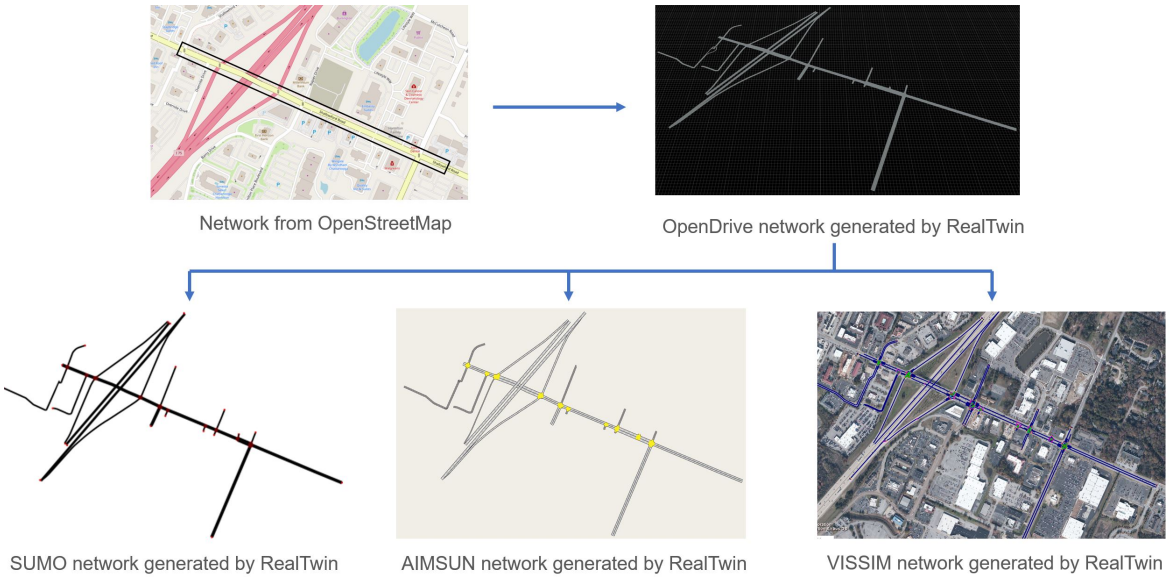


Figure 7. RealTwin generated network in different simulators.

Table 2. Form of Elements in Each RealTwin Module

	UserInput	**AbstractScenario**	**ConcreteScenario**	**Simulation**
Network	Longitude and latitude of bounding box vertices	OpenDrive network	OpenDrive network	SUMO network (`.net.xml`)
Volume	Volume data from traffic studies or sensors (`.csv`)	Intermediate variables including: • RoadID • RoadName • IntervalStart • IntervalEnd • Volume	Updated intermediate variable with OpenDrive ID	SUMO route file (`.rou.xml`)
TurningRatio	Turning ratio data from traffic studies or sensors (`.csv`)	Intermediate variables: • IntersectionID • IntersectionName • IntervalStart • IntervalEnd • TurnMovement • TurnRatio	Updated intermediate variable with OpenDrive ID	SUMO route file (`.rou.xml`)
IntersectionControl	UTDF file (`.csv`)	UTDF file containing intersection control type and signal timing	UTDF file with OpenDrive ID	SUMO additional file defining traffic light program (`.add.xml`)

Finally, the synchro UTDF file was used as the user input for "IntersectionControl" in the "Control" category. This file provides detailed TS2 actuated signal plans with a cycle length of 120 seconds for the 6 signalized intersections. As an

example, the current form of each element in each RealTwin module during the SUMO simulation generation is shown in Table 2.

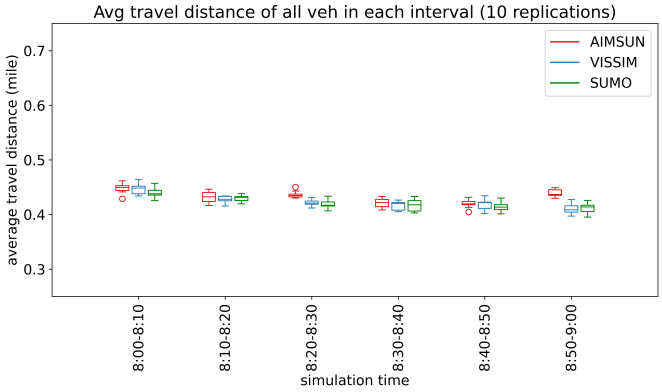
Network Output of RealTwin

The OpenDRIVE network generated by `RealTwin.ConcreteScenario`, SUMO network generated by `RealTwin.Simulation.SUMO`, AIMSUN network generated by `RealTwin.Simulation.AIMSUN`, and VISSIM network generated by `RealTwin.Simulation.VISSIM` were compared with the network from OpenStreetMap in Figure 7. Comparison of the three networks with the real network from OpenStreetMap shows that RealTwin is capable of replicating actual road geometry. This comparison demonstrates the effectiveness of RealTwin in converting complex real-world roadway network data into simulation-ready formats, saving significant network creation time and resources. This indicates RealTwin's potential to serve as a powerful tool for traffic simulation studies.

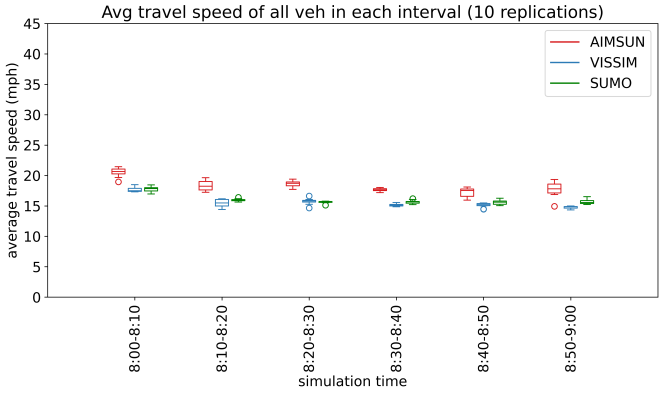
Comparison of Simulation in SUMO, VISSIM, and AIMSUN Generated by RealTwin

The generated scenario was then simulated for 10 runs with different random seeds in SUMO, VISSIM, and AIMSUN. The network-level metrics from the simulations in SUMO, VISSIM, and AIMSUN were first compared using box plots, shown in Figure 8. Specifically, the average travel distance and the average travel speed of all vehicles every 10 minutes during the simulation run were compared. The comparison of box plots from 10 replications shows the closeness in the performance metric despite the stochastic nature of different simulation runs. Good consistency in the three software can be observed across the simulations from both figures, demonstrating that RealTwin can generate consistent and comparable simulations across different simulation platforms.

In addition, the vehicle-level metrics from a single run of simulation in SUMO, VISSIM, and AIMSUN were compared using kernel density plots of individual vehicle travel distance distribution and speed distribution, shown in Figure 9. A slight difference is seen in the individual vehicle travel speed distributions, likely due to different car-following models and their default parameters (e.g. maximum acceleration, maximum deceleration, minimum headway) in different simulation software. Nevertheless, the similar shapes in both figures further demonstrated RealTwin's capability of generating consistent, comparable simulations in different simulators. This capability enables RealTwin users to cross-validate the conclusions drawn from different simulators when using microscopic simulations for traffic studies (e.g., network mobility analysis, energy efficiency analysis, and traffic impact studies). It also reduces the time and effort needed to adapt models across different simulation platforms.



(a) Average travel distance



(b) Average travel speed

Figure 8. Comparison of network metrics from SUMO, VISSIM, and AIMSUN.

Implementation of RealTwin Workflow for AV Scenario Development

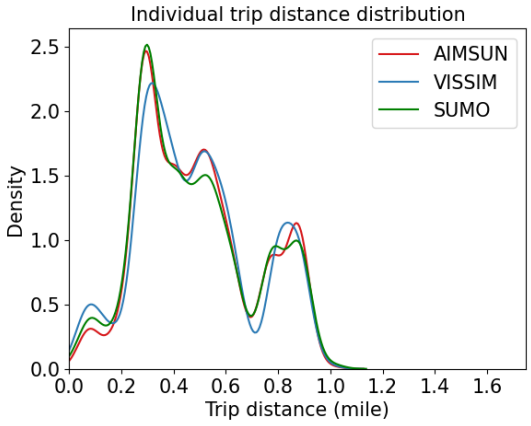
RealTwin was then used to generate AV scenarios on the developed network of the Shallowford Road corridor. Five simulation scenarios for different AV penetration levels were generated— 0%, 25%, 50%, 75%, and 100%. For this experiment, the AV scenario was implemented in SUMO, where the parameters of the Krauss car-following model for AV driving behavior were considered.

Ingestion of User Input

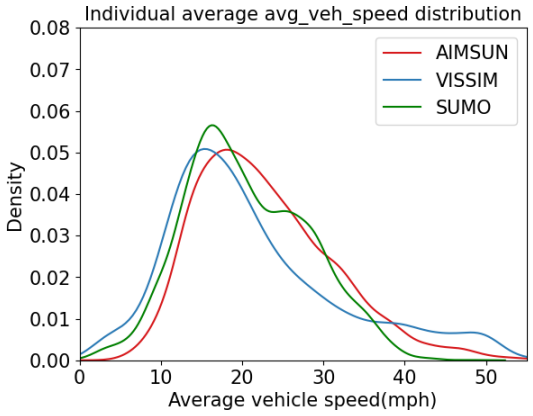
The technology application parameters in the user input interface were set as follows:

- Application: "AV"
- Vehicle types: "Human" and "AVnormal"
- Penetration level: 0%, 25%, 50%, 75%, and 100%

Further, it was assumed that the user did not provide any input on the driving behavior parameter value.



(a) Travel distance distribution



(b) Speed distribution

Figure 9. Comparison of vehicle metrics from SUMO, VISSIM, and AIMSUN.

Updating Driving Behavior Data Using AV User Input and Parameter Database

In `RealTwin.AbstractScenario`, the user input parameter values were used along with the AV parameter database described previously in ApplicationLoader to create a final dataset of parameter values reflecting the Human and AV driving behaviors. In this case study where the AV scenario was generated in SUMO, the AV driving behavior was defined by setting the Krauss car-following model parameter values. Because no input was available from users on the driving behavior parameters for the two vehicle types—Human and AVnormal, the final set of parameter values used the driving behavior parameter values provided in the AV parameter database.

Finalizing AV and Human Driving Behavior Parameter

In `RealTwin.ConcreteScenario`, the final dataset generated by `RealTwin.AbstractScenario` was referred to set the RealTwin driving behavior parameter variable values for Human and AV vehicle types for this scenario. These variables were then ready to be called by the functions in the `RealTwin.Simulation` module and assigned to SUMO software-specific variables.

Generating Simulations

In the final step, the `RealTwin.Simulation` module was used to generate a simulation scenario in SUMO. The driving behavior parameter values finalized for variables in the concrete scenario were used to define the two vehicle types—Human and AVnormal—in the SUMO flow XML file in which vehicle type distribution parameter was also defined.

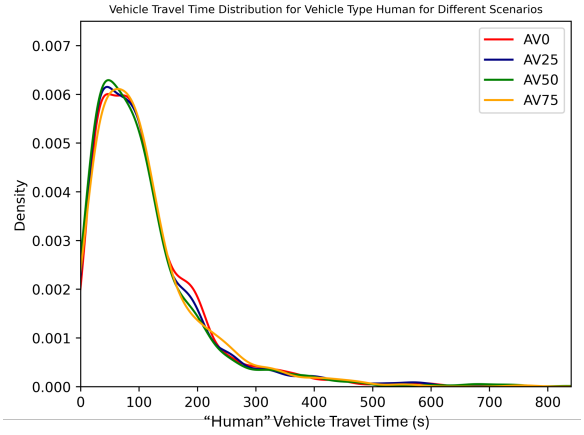
In this case, the SUMO flow file, net file, and turn file for the Shallowford Road corridor were already available from the steps described in Table 2. SUMO `jtrrouter` was used to generate the route file along with the two defined vehicle types considering their composition in the fleet. This process was used to create five simulation scenarios with different AV penetration levels.

Simulation Results

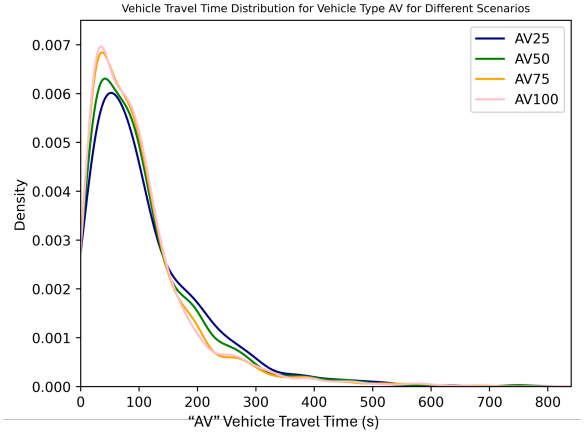
The simulation run results obtained from the five penetration levels of AV in fleet composition were then analyzed. The vehicle travel times for the two vehicle types—AVnormal and Human—were compared for the five sets of scenarios of different AV penetration levels. The results in Figure 10 show the distribution of travel times.

The results indicate that although differences are hardly observed in the travel time distribution of AV and Humans when AV penetration is low, for higher AV penetration levels, more vehicles experience lower travel times. The travel time distribution curves for AV shifted to the left and have a higher peak for the higher penetration level on AV75 and AV100 compared to travel time distribution for human vehicles. This result is in alignment with the expectation of AV driving behavior. For example, the AVnormal driving behavior parameter "minimum gap" is less than and the "accel" parameter is greater than those of the Human driving parameter. The results from this sample simulation experiment generated by RealTwin demonstrate the potential of RealTwin to generate multiple scenarios with significantly reduced manual effort.

The application extension for AV scenario generation was also developed for VISSIM and will be extended to AIMSUN in the future. Figure 11 shows the distribution of travel times in AV scenarios generated by RealTwin in VISSIM. Similar to the results in SUMO, the increase in AV penetration leads to a larger proportion of shorter vehicle travel times in the travel time distribution for AV vehicles when compared to the travel time distribution for human vehicles. Although the range of vehicle travel time for both VISSIM and SUMO scenarios are similar, differences in car-following models and driving behavior parameter values between the two simulators may have impacted the travel times for each vehicle type differently. In the future, for both SUMO and VISSIM, the AV parameter set may be calibrated to the same

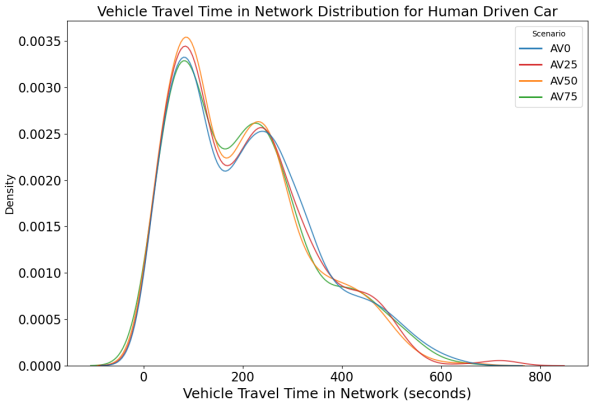


(a) Vehicle type: "Human"

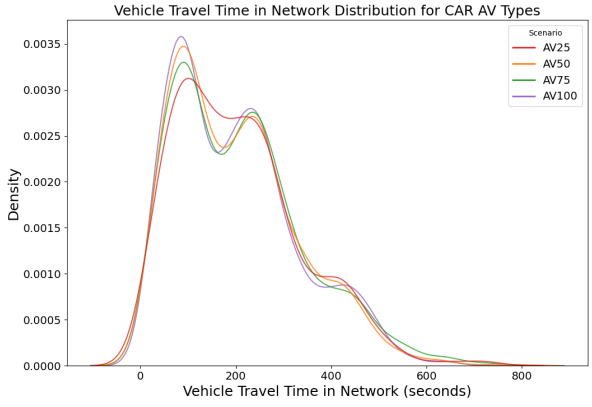


(b) Vehicle type: "AVnormal"

Figure 10. Vehicle travel time distribution for different vehicle types under varying penetration levels of AVs in SUMO.



(a) Vehicle type: "Human"



(b) Vehicle type: "AV"

Figure 11. Vehicle travel time distribution for different vehicle types under varying penetration levels of AVs in VISSIM.

data (real or synthetic) to ensure consistency in impact for the same technology scenario.

Implementation of RealTwin Workflow for Simulation Calibration

Calibrate Turning Ratios and Inflow

To demonstrate RealTwin's calibration capability, a case study was conducted in SUMO using the Shallowford Road network, shown in Figure 12. The same calibration module and algorithms can also be applied to other simulators. This network has 10 intersections: 6 signalized intersections with traffic demand data available through traffic cameras and 4 unsignalized intersections with no demand data (marked by red arrows). In SUMO, no inflow was assigned to edges without data being provided, and traffic volume will be evenly assigned to each movement of the intersection approach if the turning ratios were not defined by the user. Therefore, RealTwin was used to calibrate inflow and turning ratios at those unsignalized intersections where data were unavailable. Specifically, the average GEH of volume was minimized at approaches where data were available using Tabu Search (TS), Genetic Algorithm (GA), and Simulated Annealing (SA), respectively.

The results of the three algorithms are shown in Table 3. As shown in the table, GA outperformed TS and SA in terms of final objective value (mean GEH). However, TS equally minimized the GEH of each approach, leading to much higher percentages of GEH that are smaller than 5. Therefore, results from TS were selected. Finally, Figure 13 compares the approach of GEH at different intersections before and after calibration. RealTwin's calibration module significantly reduces the GEH of traffic volume at most approaches, demonstrating its effectiveness in simulation calibration.

Calibrate Driving Behavior

Next, using the simulation with calibrated turning ratios and inflows, we calibrates the driving behavior, specifically

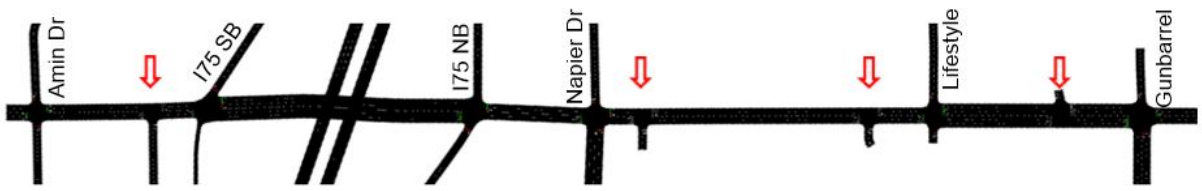


Figure 12. Intersections for calibration.

Table 3. Comparison of turning ratio and inflow calibration results

Algorithm	Mean GEH before calibration	% of GEH ¡5 before calibration	Mean GEH after calibration	% of GEH ¡5 after calibration
TS	10.32	50.00%	1.40	90.90%
GA	10.32	50.00%	1.23	68.18%
SA	10.32	50.00%	5.71	68.18%

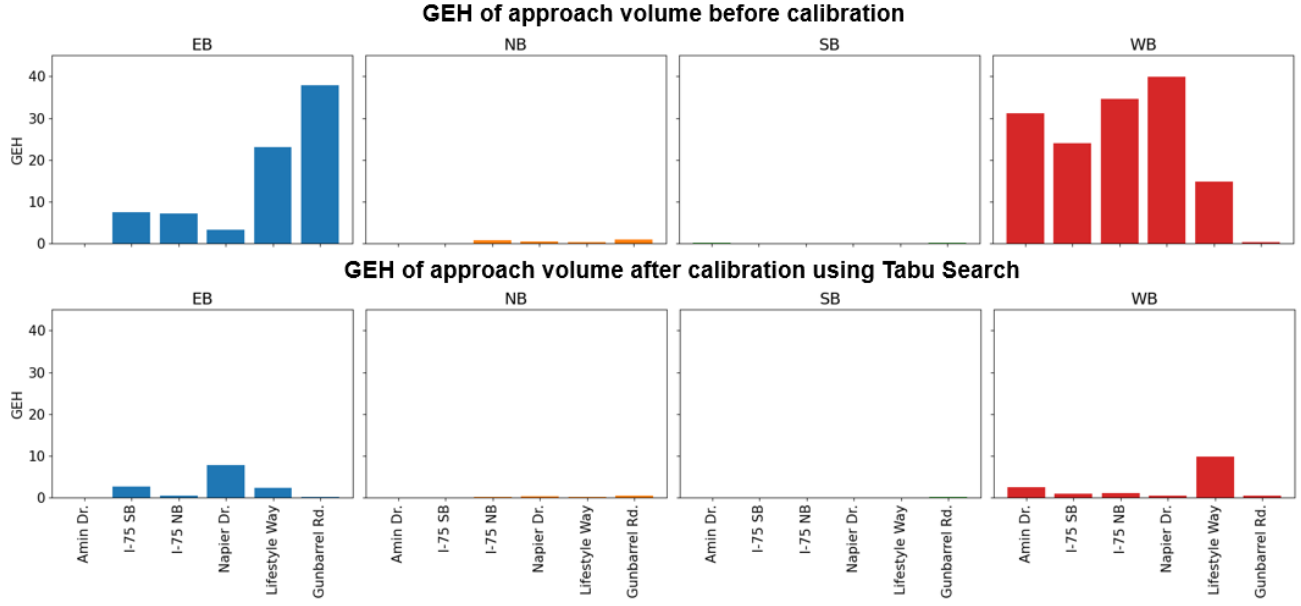


Figure 13. Comparison of approach GEH at different intersections before and after calibration.

car-following parameters, by minimizing the MAE of average eastbound and westbound travel time. The calibrated parameters in SUMO are listed below and their calibration ranges are determined from literature review (*74*, *75*).

- **minGap:** minimum gap when standing (m), [1.00, 3.00];
- **accel:** acceleration (m/s^2), [2.50, 3.00];
- **decel:** deceleration (m/s^2), [4.50, 5.30];
- **emergencyDecel:** maximum deceleration (m/s^2), [5.00, 9.30];
- **sigma:** driver's imperfection (0 denotes perfect driving), [0.00, 1.00];
- **tau:** desired headway (s), [0.25, 1.25].

According to Google Maps, the average travel time of vehicles traveling eastbound on the same day and time was 240 s, and the average travel time of vehicles traveling westbound was 180 s. Before the calibration, the eastbound travel time from the simulation is 200.29 s, and the westbound travel time from the simulation is 207.36 s. These travel times in simulation were significantly different from the real-world travel time. As shown in Table 4, RealTwin's calibration process significantly reduces the MAE for all methods.

Table 4. Comparison of driving behavior calibration results using different algorithms

	MAE		Eastbound travel time (s)			Westbound travel time (s)		
	before cal	**after cal**	**real world**	**before cal**	**after cal**	**real world**	**before cal**	**after cal**
TS	33.54	23.28	240	200.29	212.86	180	207.36	197.61
GA	33.54	15.21	240	200.29	241.67	180	207.36	208.85
SA	33.54	17.57	240	200.29	237.50	180	207.36	212.65

Conclusion

In this paper, we presented RealTwin, a comprehensive tool that can automatically ingest real-world data and generate scenarios for microscopic traffic simulations. Following a streamlined scenario generation and calibration workflow, RealTwin effectively bridges gaps between traffic data from various sources and traffic simulators, making microscopic traffic simulation more accessible for researchers and engineers across various levels of expertise. Using SUMO, VISSIM, and AIMSUN as examples, the implementation of RealTwin for scenario development of a case study was presented. The consistency observed in the SUMO, VISSIM, and AIMSUN simulations created by RealTwin demonstrated its ability to generate comparable simulations across different simulators. In addition, RealTwin's capability for technology scenario generation was introduced and illustrated. This feature can contribute to more comprehensive microscopic simulations facilitating the analysis of how various technological innovations can potentially influence mobility, energy efficiency, and safety. Finally, RealTwin was employed to calibrate a simulation in SUMO. The calibration module enhances RealTwin's ability to generate realistic simulations that reflect real-world traffic operations.

Limitations and Future Work

While RealTwin demonstrates significant advancements in automating scenario generation for microscopic traffic simulations, several limitations remain. For example, the current reliance on OpenStreetMap for network data introduces potential inaccuracies, such as incorrect intersection lane configurations, which require manual corrections. Additionally, the tool presently supports a limited range of user input formats, including inputs for traffic signals. The user interface is still in its early stages, necessitating further development to improve accessibility and user experience. To address these limitations, future efforts will focus on improving and enhancing the following aspects of RealTwin:

- **Automation:** Advance RealTwin workflow's automation for more streamlined and accurate scenario generation. This includes but is not limited to:
  - *Automatic validation and correction of problematic user input.* For instance, the coordinates provided by users for network generation may not form a valid polygon. Similarly, mismatches between road names in the volume input file and the network map may cause scenario generation failure. Future developments will focus on creating tools to assist users in validating and correcting such inputs to ensure successful scenario generation.
  - *Automatic correction of network layout.* The network data from OpenStreetMap often has issues with intersection lane configuration. We are in the process of developing tools for RealTwin to use computer vision and deep learning techniques to automatically process satellite or street-level images of intersections and correct the network.
  - *Automatic identification and correction of inconsistent data.* Inconsistent data, such as mismatched traffic volumes or intersection turning ratios from different data sources, can affect the realism of the simulation scenario. Future developments aim to integrate automated algorithms to identify and resolve such data inconsistencies.
- **Integration:** Improve RealTwin capabilities to integrate more:
  - *Emerging applications.* RealTwin currently supports scenario generation for Autonomous Vehicles (AV). Expanding support for other emerging technologies, such as Connected and Automated Vehicles (CAVs) and electrification, is planned as part of future developments.
  - *Multi-modal traffic.* In future versions, we plan to support multi-modal traffic (e.g., bus, truck, etc.) and allow vehicle composition to be a user-defined input. Additionally, we will provide recommended driving behavior parameters for each traffic mode for user reference.
  - *User input data formats.* RealTwin currently uses OpenStreetMap for network generation and supports direct outputs from GridSmart as inputs for demand and UTDF as inputs for traffic signal. In the future, ingestion of additional data formats, such as United States Geological Survey (USGS) map data for network and OD data or per vehicle

record data (trajectory data) for traffic demand, is planned to be added to the tool to provide greater flexibility.

  - *Algorithms and performance measures.* RealTwin will include additional algorithms and performance measures in its scenario generation and calibration process. For example, RealTwin currently uses `jtrrouter` to generate the route file for SUMO. SUMO *routeSampler.py* could be another option and will be considered as an alternative way to generate SUMO routes in our future version. Another example is that the calibration module of RealTwin in future versions will utilize additional calibration algorithms (e.g., Artificial Neural Networks (ANNs)) and objective functions (e.g. minimizing Root Mean Square Error (RMSE), minimizing root mean square normalized error (RMSNE), and multi-objective calibration function, etc.), and expand the range of MOEs to include metrics that capture detailed driving behavior (e.g., lane change frequency, acceleration and deceleration profiles, speed variance), safety indicators (e.g., time-to-collision), as well as energy consumption and emissions estimates.

- **Ease of use:** Enable user-friendliness and ease of use of the RealTwin tool.

  - *RealTwin user interface (UI).* we will develop user-accessible metrics that provide characteristics of the comprehensive RealTwin database such as accuracy and completeness, and develop a user-friendly and flexible UI prototype to generate scenarios. This UI prototype will support users to select, locate, and generate scenarios with ease while including transparency on the accuracy and completeness of data and scenarios.
  - *Customization.* We will enhance RealTwin's flexibility by enabling users to customize various aspects of the tool. For example, currently, RealTwin supports calibrating predefined parameters, but users cannot define or choose additional parameters. Future updates will include this functionality, allowing users to adapt the tool to their specific research needs.

In addition to improving RealTwin's capabilities, we will use RealTwin to generate and calibrate scenarios for networks of different scales to perform further in-depth performance analyses in future work.

Acknowledgements

We thank the City of Chattanooga for sharing their city-wide historical and real-time traffic and signal data.

Declaration of conflicting interests

Chieh (Ross) Wang is a member of Transportation Research Record's Editorial Board. All other authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: Xu, Saroj, Wang, and Shao; analysis and interpretation of results: Xu, Saroj, Wang, and Shao; draft manuscript preparation: Xu, Saroj, Wang, and Shao. All authors reviewed the results and approved the final version of the manuscript.

Funding

This work was supported by the US Department of Energy, Vehicle Technologies Office, Energy Efficient Mobility Systems (EEMS) program, under project Real-Twin (EEMS114).

References

1. Ehlert, P. A. and L. J. Rothkrantz. Microscopic traffic simulation with reactive driving agents. In *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585)*. IEEE, 2001, pp. 860–865.
2. Hidas, P. Modelling lane changing and merging in microscopic traffic simulation. *Transportation Research Part C: Emerging Technologies*, Vol. 10, No. 5-6, 2002, pp. 351–371.
3. Chu, L., H. X. Liu, J.-S. Oh, and W. Recker. A calibration procedure for microscopic traffic simulation. In *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*, Vol. 2. IEEE, 2003, pp. 1574–1579.
4. Passos, L. S., R. J. Rossetti, and Z. Kokkinogenis. Towards the next-generation traffic simulation tools: a first appraisal. In *6th Iberian Conference on Information Systems and Technologies (CISTI 2011)*. IEEE, 2011, pp. 1–6.
5. Lopez, P. A., M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. Microscopic Traffic Simulation using SUMO. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. URL https://elib.dlr.de/124092/.
6. *VISSIM Manual*. PTV AG, 2022.
7. Trafficware. SimTraffic, 2023. Available from: https://simtraffic.net/#[object%20Object]/.
8. Caliper Corporation. TransModeler, 2023. Available from: https://www.caliper.com/transmodeler/default.htm/.
9. Aimsun. *Aimsun Next 22 User's Manual*. Barcelona, Spain, aimsun next 22.0.1 ed., 2022. [Online]. URL https://docs.aimsun.com/next/22.0.1/.
10. Jones, S. L., A. J. Sullivan, N. Cheekoti, M. D. Anderson, and D. Malave. Traffic simulation software comparison study. *UTCA report*, Vol. 2217.

11. Chen, D., M. Zhu, H. Yang, X. Wang, and Y. Wang. Data-driven Traffic Simulation: A Comprehensive Review. *IEEE Transactions on Intelligent Vehicles*.
12. Guastella, D. A., B. Cornelis, and G. Bontempi. Traffic Simulation with Incomplete Data: The Case of Brussels. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Methods for Enriched Mobility Data: Emerging issues and Ethical perspectives 2023*. 2023, pp. 15–24.
13. Renninger, A., S. Ameen Noman, T. Atkison, and J. Sussman. Live Intersection Data Acquisition for Traffic Simulators (LIDATS). *Sensors*, Vol. 24, No. 11, 2024, p. 3392.
14. Ratrout, N. T. and S. M. Rahman. A comparative analysis of currently used microscopic and macroscopic traffic simulation software. *The Arabian Journal for Science and Engineering*, Vol. 34, No. 1B, 2009, pp. 121–133.
15. Lin, Y., M. Ratzel, and M. Althoff. Automatic traffic scenario conversion from OpenSCENARIO to CommonRoad. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2023, pp. 4941–4946.
16. SAE. J3016C: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles, 2021. URL https://www.sae.org/standards/content/j3016_202104/.
17. PTVGroup. Traffic Simulation. https://www.ptvgroup.com/en/application-areas/traffic-simulation#:~:text=How%20long%20does%20it%20take,out%20more%20in%20this%20webinar, 2024. Accessed: November 18, 2024.
18. Ferrara, A., S. Sacone, S. Siri, A. Ferrara, S. Sacone, and S. Siri. Microscopic and mesoscopic traffic models. *Freeway traffic modelling and control*, 2018, pp. 113–143.
19. Xu, G. and V. V. Gayah. Non-unimodal and non-concave relationships in the network Macroscopic Fundamental Diagram caused by hierarchical streets. *Transportation Research Part B: Methodological*, Vol. 173, 2023, pp. 203–227.
20. Xu, G., Z. Yu, and V. V. Gayah. Analytical method to approximate the impact of turning on the macroscopic fundamental diagram. *Transportation research record*, Vol. 2674, No. 9, 2020, pp. 933–947.
21. Xu, G., P. Zhang, V. V. Gayah, and X. Hu. Opposing hysteresis patterns in flow and outflow macroscopic fundamental diagrams and their implications. *Transportation Research Record*, Vol. 2677, No. 8, 2023, pp. 100–117.
22. Taglieri, D., H. Liu, and V. V. Gayah. Network-wide implementation of roundabouts versus signalized intersections on urban streets: analytical and simulation comparison. *Transportation research record*, Vol. 2678, No. 5, 2024, pp. 719–735.
23. Shiledar, A., V. Sujan, A. Siekmann, and J. Yuan. *Enhanced Safety of Heavy-Duty Vehicles on Highways through Automatic Speed Enforcement–A Simulation Study*. Tech. rep., SAE Technical Paper, 2024.
24. Wang, C., C. Xu, J. Xia, Z. Qian, and L. Lu. A combined use of microscopic traffic simulation and extreme value methods for traffic safety evaluation. *Transportation Research Part C: Emerging Technologies*, Vol. 90, 2018, pp. 281–291.
25. Garg, M. and M. Bouroche. Can connected autonomous vehicles improve mixed traffic safety without compromising efficiency in realistic scenarios? *IEEE Transactions on Intelligent Transportation Systems*, Vol. 24, No. 6, 2023, pp. 6674–6689.
26. Yu, Z., G. Xu, V. V. Gayah, and E. Christofa. Incorporating phase rotation into a person-based signal timing optimization algorithm. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 1, 2020, pp. 513–521.
27. Liu, H., V. V. Gayah, and M. W. Levin. A max pressure algorithm for traffic signals considering pedestrian queues. *Transportation Research Part C: Emerging Technologies*, Vol. 169, 2024, p. 104865.
28. Kavas-Torris, O. and L. Guvenc. A Comprehensive Eco-Driving Strategy for CAVs with Microscopic Traffic Simulation Testing Evaluation. *Sensors*, Vol. 23, No. 20, 2023, p. 8416.
29. Ahmad, A., A. S. Al-Sumaiti, Y.-J. Byon, and K. Al Hosani. Eco-Driving Framework for Autonomous Vehicles at Signalized Intersection in Mixed-Traffic Environment. *IEEE Access*, Vol. 12, 2024, pp. 85291–85305.
30. Jamil, U., M. Malmir, A. Chen, M. Filipovska, M. Xie, C. Ding, and Y.-F. Jin. Developing an eco-driving strategy in a hybrid traffic network using reinforcement learning. *Science Progress*, Vol. 107, No. 3, 2024, p. 00368504241263406.
31. Shi, Y., Z. Wang, T. J. LaClair, C. Wang, and Y. Shao. Real-time control of connected vehicles in signalized corridors using pseudospectral convex optimization. *Optimal Control Applications and Methods*, Vol. 44, No. 4, 2023, pp. 2257–2277.
32. Zhang, J., C. Chang, Z. He, W. Zhong, D. Yao, S. Li, and L. Li. CAVSim: A microscopic traffic simulator for evaluation of connected and automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 24, No. 9, 2023, pp. 10038–10054.
33. Vishnoi, S. C., J. Ji, M. Bahavarnia, Y. Zhang, A. F. Taha, C. G. Claudel, and D. B. Work. CAV Traffic Control to Mitigate the Impact of Congestion from Bottlenecks: A Linear Quadratic Regulator Approach and Microsimulation Study. *Journal on Autonomous Transportation Systems*, Vol. 1, No. 2, 2024, pp. 1–37.
34. Lopez, P. A., M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2575–2582.
35. SUMO. SUMO at a Glance. https://sumo.dlr.de/docs/SUMO_at_a_Glance.html, 2024. Accessed: 2024-07-21.
36. VISSIM, P. *VISSIM Manual*. PTV Group, 2022.
37. VISSIM, P. *VISSIM COM Manual*. PTV Group, 2022.

38. Artal-Villa, L. and C. Olaverri-Monreal. Vehicle-pedestrian interaction in sumo and unity3d. In *New Knowledge in Information Systems and Technologies: Volume 2*. Springer, 2019, pp. 198–207.
39. Oliveira, A. and T. Vazão. Generating synthetic datasets for mobile wireless networks with sumo. In *Proceedings of the 19th ACM international symposium on mobility management and wireless access*. 2021, pp. 33–42.
40. Nayak, R. P., S. Sethi, and S. K. Bhoi. PHVA: a position based high speed vehicle detection algorithm for detecting high speed vehicles using vehicular cloud. In *2018 International Conference on Information Technology (ICIT)*. IEEE, 2018, pp. 227–232.
41. Soumya, S. and N. Bappalige. Performance analysis of mobile ad hoc routing protocols in vehicular ad hoc networks using NS3. *International Journal of Applied Engineering and Management Letters*, Vol. 5, No. 1, 2021, pp. 19–28.
42. Wu, C., A. R. Kreidieh, K. Parvate, E. Vinitsky, and A. M. Bayen. Flow: A modular learning framework for mixed autonomy traffic. *IEEE Transactions on Robotics*, Vol. 38, No. 2, 2021, pp. 1270–1286.
43. Lu, J. and X. S. Zhou. Virtual track networks: A hierarchical modeling framework and open-source tools for simplified and efficient connected and automated mobility (CAM) system design based on general modeling network specification (GMNS). *Transportation Research Part C: Emerging Technologies*, Vol. 153, 2023, p. 104223.
44. Chiappone, S., O. Giuffrè, A. Granà, R. Mauro, and A. Sferlazza. Traffic simulation models calibration using speed–density relationship: An automated procedure based on genetic algorithm. *Expert Systems with Applications*, Vol. 44, 2016, pp. 147–155.
45. Daguano, R. F., L. R. Yoshioka, M. L. Netto, C. L. Marte, C. A. Isler, M. M. D. Santos, and J. F. Justo. Automatic Calibration of Microscopic Traffic Simulation Models Using Artificial Neural Networks. *Sensors*, Vol. 23, No. 21, 2023, p. 8798.
46. Casas, J., J. L. Ferrer, D. Garcia, J. Perarnau, and A. Torday. Traffic simulation with aimsun. *Fundamentals of traffic simulation*, 2010, pp. 173–232.
47. Eriksson, M. and V. Hallberg. Comparison between JSON and YAML for data serialization. *The School of Computer Science and Engineering Royal Institute of Technology*, 2011, pp. 1–25.
48. Cubic Transportation Systems, Inc. GRIDSMART, 2023. Available from: https://transportation.cubic.com/.
49. OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017.
50. Association for Standardisation of Automation and Measuring Systems (ASAM). ASAM OpenDRIVE Standard (1.7.0), 2021. URL https://www.asam.net/standards/detail/opendrive/.
51. Sabra, W. *Signal Timing Process Final Report*. Tech. rep., 2003.
52. Cubic Transportation Systems. *Synchro Studio 12 User Guide*, 2023. URL https://support.trafficware.com/support/solutions/articles/69000836741-synchro-studio-12-user-guide. Accessed: 2023-07-31.
53. Blog, S. SAE levels of driving automation refined for clarity and international audience. *SAE International*, Vol. 3.
54. Seth, D. and M. L. Cummings. Traffic efficiency and safety impacts of autonomous vehicle aggressiveness. *simulation*, Vol. 19, 2019, p. 20.
55. Deluka Tibljaš, A., T. Giuffrè, S. Surdonja, and S. Trubia. Introduction of Autonomous Vehicles: Roundabouts design and safety performance evaluation. *Sustainability*, Vol. 10, No. 4, 2018, p. 1060.
56. Elvarsson, A. B. Modelling urban driving and stopping behavior for automated vehicles. *Semester Proj. IVT, ETH Zürich, Zürich*, 2017, pp. 1–43.
57. Espinosa, M. Safety evaluation of signalized intersections with automated vehicles at various penetration levels based on conflict analysis of simulated traffic. doi: 10.32920/ryerson.14652957.v1. URL https://rshare.library.torontomu.ca/articles/thesis/Safety_evaluation_of_signalized_intersections_with_automated_vehicles_at_various_penetration_levels_based_on_conflict_analysis_of_simulated_traffic/14652957.
58. Morando, M. M., L. T. Truong, and H. L. Vu. Investigating safety impacts of autonomous vehicles using traffic micro-simulation. In *Australasian transport research forum*. 2017, pp. 1–6.
59. Maryam Mousavi, S., D. Lord, B. Dadashova, and S. Reza Mousavi. Can autonomous vehicles enhance traffic safety at unsignalized intersections? In *International Conference on Transportation and Development 2020*. American Society of Civil Engineers Reston, VA, 2020, pp. 194–206.
60. Rao, R. S., S. Yoon Park, and G.-L. Chang. Developing the guidelines for managing autonomous vehicle flows on congested highways: A case study of MD-100. *Simulation*, Vol. 97, No. 6, 2021, pp. 367–382.
61. Rezaei, A. and B. Caulfield. Simulating a transition to autonomous mobility. *Simulation Modelling Practice and Theory*, Vol. 106, 2021, p. 102175.
62. Sukennik, P., J. Lohmiller, and J. Schlaich. Simulation-based forecasting the impacts of autonomous driving. In *Proceedings of the International Symposium of Transport Simulation (ISTS'18) and the International Workshop on Traffic Data Collection and its Standardization (IWTDCS'18), Matsuyama, Japan*. 2018, pp. 6–8.
63. Stanek, D., R. T. Milam, E. Huang, and Y. A. Wang. *Measuring autonomous vehicle impacts on congested networks using simulation*. Tech. rep., 2018.

64. Tafidis, P., A. Pirdavani, T. Brijs, and H. Farah. Intersection control type effect on automated vehicle operation. In *CICTP 2019*. 2019, pp. 2742–2750.
65. Wang, Y. and L. Wang. Autonomous vehicles' performance on single lane road: A simulation under VISSIM environment. In *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. IEEE, 2017, pp. 1–5.
66. Li, D. and P. Wagner. Impacts of gradual automated vehicle penetration on motorway operation: a comprehensive evaluation. *European transport research review*, Vol. 11, 2019, pp. 1–10.
67. Lu, Q., T. Tettamanti, D. Hörcher, and I. Varga. The impact of autonomous vehicles on urban traffic network capacity: an experimental analysis by microscopic traffic simulation. *Transportation Letters*, Vol. 12, No. 8, 2020, pp. 540–549.
68. Richter, G., L. Grohmann, P. Nitsche, and G. Lenz. Anticipating Automated Vehicle Presence and the Effects on Interactions with Conventional Traffic and Infrastructure. In *SUMO*. 2019, pp. 230–243.
69. Hourdakis, J., P. G. Michalopoulos, and J. Kottommannil. Practical procedure for calibrating microscopic traffic simulation models. *Transportation research record*, Vol. 1852, No. 1, 2003, pp. 130–139.
70. Lidbe, A. D., A. M. Hainen, and S. L. Jones. Comparative study of simulated annealing, tabu search, and the genetic algorithm for calibration of the microsimulation model. *Simulation*, Vol. 93, No. 1, 2017, pp. 21–33.
71. Yu, M. and W. D. Fan. Calibration of microscopic traffic simulation models using metaheuristic algorithms. *International Journal of Transportation Science and Technology*, Vol. 6, No. 1, 2017, pp. 63–77.
72. Igene, M., Q. Luo, K. Jimee, M. Soltanirad, T. Bataineh, and H. Liu. Integrating LiDAR Sensor Data into Microsimulation Model Calibration for Proactive Safety Analysis. *Sensors*, Vol. 24, No. 13, 2024, p. 4393.
73. Hale, D. K., A. Ghiasi, F. Khalighi, D. Zhao, X. Li, and R. M. James. Vehicle Trajectory-Based Calibration Procedure for Microsimulation. *Transportation Research Record*, Vol. 2677, No. 1, 2023, pp. 1764–1781.
74. Langer, M., M. Harth, L. Preitschaft, R. Kates, and K. Bogenberger. Calibration and assessment of urban microscopic traffic simulation as an environment for testing of automated driving. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021, pp. 3210–3216.
75. Mecheva, T., R. Furnadzhiev, and N. Kakanakov. Modeling driver behavior in road traffic simulation. *Sensors*, Vol. 22, No. 24, 2022, p. 9801.
76. Karimi, M., M. Miriestahbanati, H. Esmaeeli, and C. Alecsandru. Multi-Objective Stochastic Optimization Algorithms to Calibrate Microsimulation Models. *Transportation Research Record*, Vol. 2673, No. 4, 2019, pp. 743–752. doi:10.1177/0361198119838260. URL https://doi.org/10.1177/0361198119838260. https://doi.org/10.1177/0361198119838260.
77. Bandi, M. M. and V. George. Calibration of Vehicle and Driver Characteristics in VISSIM and ANN-based Sensitivity Analysis. *IJM*, Vol. 13, No. 2, 2020, pp. 79–101. doi:10.34196/ijm.00219. URL https://doi.org/10.34196/ijm.00219.
78. Wisconsin Department of Transportation. TEOpS 16-20 Attachment 6.3: Traffic Signal Coordination Guidelines, 2020. URL https://wisconsindot.gov/dtsdManuals/traffic-ops/manuals-and-standards/teops/16-20att6.3.pdf. Accessed: 2025-04-01.
79. Lokesh, Y., C. N S, and N. Tr. Calibration and Validation of VISSIM Driving Behavior Parameters for Signalized and UN-Signalized Intersections in Bangalore City. *GIS Science Journal*, 2020, pp. 911–936.
80. Kentucky Transportation Cabinet. Appendix B – VISSIM Development and Calibration Report, 2020. URL https://transportation.ky.gov/Planning/Planning%20Studies%20and%20Reports/Appendix%20B%20-%20VISSIM%20Development%20and%20Calibration%20Report.pdf. Accessed: 2025-04-01.
81. Dowling, R., A. Skabardonis, J. Halkias, G. McHale, and G. Zammit. Guidelines for calibration of microsimulation models: framework and applications. *Transportation Research Record*, Vol. 1876, No. 1, 2004, pp. 1–9.
82. Hu, T.-Y. and L.-W. Chen. Traffic signal optimization with greedy randomized tabu search algorithm. *Journal of transportation engineering*, Vol. 138, No. 8, 2012, pp. 1040–1050.
83. Pinheiro, M., X. Emery, A. M. A. Rocha, T. Miranda, and L. Lamas. Boreholes plans optimization methodology combining geostatistical simulation and simulated annealing. *Tunnelling and Underground Space Technology*, Vol. 70, 2017, pp. 65–75.
84. Gamboa-Venegas, C., S. Gómez-Campos, and E. Meneses. Calibration of Traffic Simulations Using Simulated Annealing and GPS Navigation Records. In *Annual International Conference on Information Management and Big Data*. Springer, 2021, pp. 17–33.