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

Lara CODECÁ[1], Jakob ERDMANN[2], Vinny CAHILL[1], and Jérôme HÄRRI[3]

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

Abstract

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

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

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

1 Introduction

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

[1] United Nations Sustainable Development Goals: https://www.un.org/sustainabledevelopment/sustainable-development-goals/ Access: September, 2020

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

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

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

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

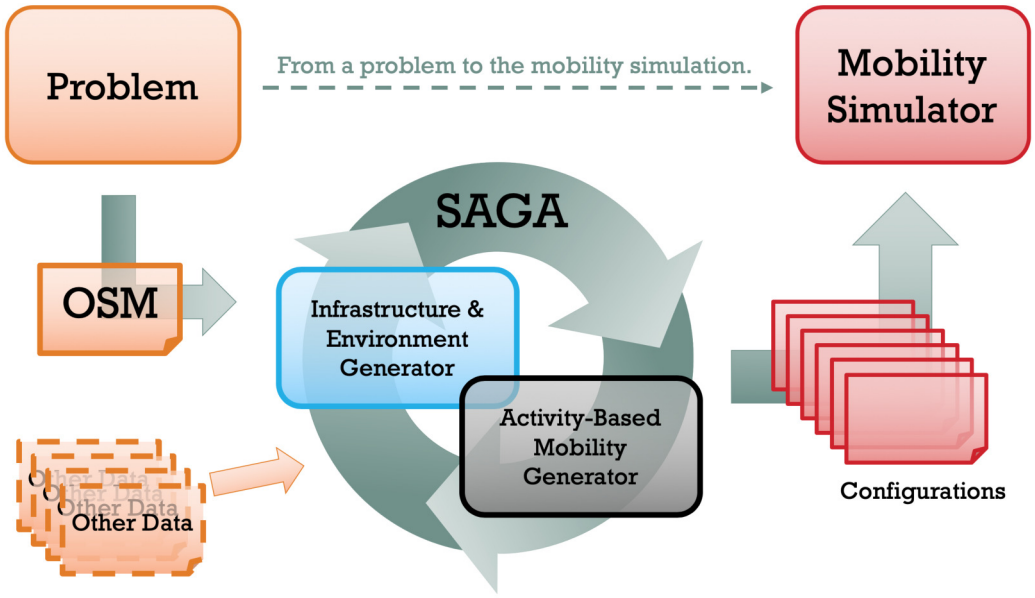


Figure 1: Overview of SAGA's interaction model.

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

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

2 Related Work

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

[2]VISSIM: http://vision-traffic.ptvgroup.com/en-us/training-support/support/ptv-vissim/ Access: September, 2020
[3]Paramics: https://www.paramics.co.uk/en/ Access: September, 2020
[4]Cube: https://www.citilabs.com/software/ Access: September, 2020

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

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

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

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

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

[5]SUMO Wiki: SUMOPy https://sumo.dlr.de/docs/Contributed/SUMOPy.html Access: September, 2020
[6]SUMO Wiki: ACTIVITYGEN https://sumo.dlr.de/docs/ACTIVITYGEN.html Access: September, 2020
[7]SUMO Wiki: OSMWebWizard http://sumo.sourceforge.net/userdoc/Tutorials/OSMWebWizard.html Access: September, 2020
[8]SAGA on GitHub: https://github.com/lcodeca/SUMOActivityGen Access: September, 2020

3 Modelling Smart Mobility

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

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

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

3.1 Activity-based Mobility Plans

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

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

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

3.2 Components of a Mobility Simulation

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

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

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

3.3 Automation Process

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

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

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

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

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

the resulting simulations.

4 Mobility Scenario Generation Framework

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

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

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

4.1 Requirements

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

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

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

4.2 Top-down Overview

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

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

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

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

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

4.3 Workflow and Capabilities

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

4.3.1 Iterative Scenario Generation Process

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

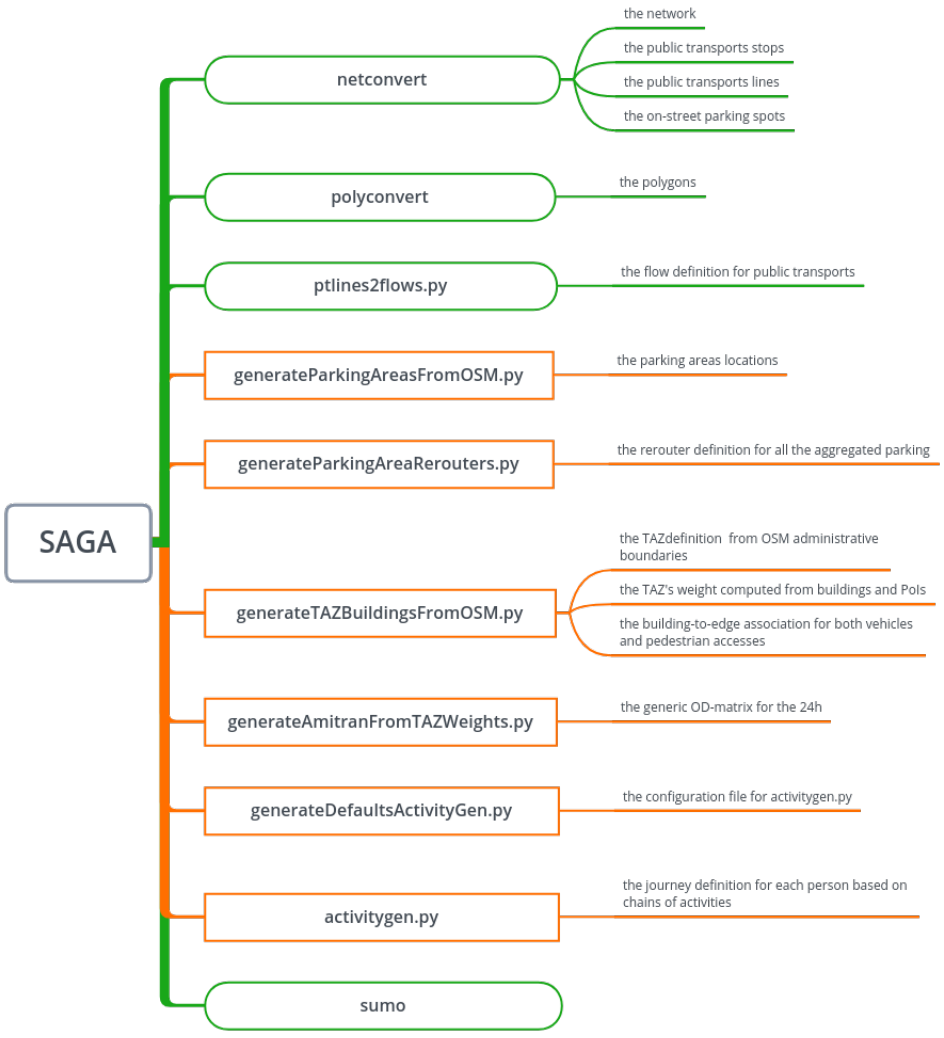


Figure 2: Overview of the scenario generation process. The rounded boxes in green are the tools provided by SUMO. The squared boxes in orange are the tools provided by SAGA. On the right of each box is listed the generated output.

1. The OSM file is processed using `netconvert`[10] to obtain (i) the network definition, (ii) the public transportation stops, (iii) the public transportation lines, and (iv) the on-street parking spots.

2. The OSM file and the newly generated network definition is processed using `polyconvert`[11] to extract all the environmental geometrical features available (e.g., buildings, PoIs, and

[10]SUMO Wiki: NETCONVERT https://sumo.dlr.de/docs/NETCONVERT.html Access: September, 2020
[11]SUMO Wiki: POLYCONVERT https://sumo.dlr.de/docs/POLYCONVERT.html Access: September, 2020

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

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

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

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

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

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

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

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

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

[12] SUMO Wiki: ptlines2flows.py https://sumo.dlr.de/docs/Tutorials/PT_from_OpenStreetMap.html Access: September, 2020

[13] SUMO Wiki: Parking areas https://sumo.dlr.de/docs/Simulation/ParkingArea.html Access: September, 2020

[14] SUMO Wiki: Parking Area Rerouters https://sumo.dlr.de/docs/Simulation/Rerouter.html#rerouting_to_an_alternative_parking_area Access: September, 2020

[15] OpenStreetMap Wiki: administrative boundaries https://wiki.openstreetmap.org/wiki/Tag:boundary=administrative Access: September, 2020

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

[17] SUMO Wiki: Amitran https://sumo.dlr.de/docs/Demand/Importing_O/D_Matrices.html#the_amitran_format Access: September, 2020

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

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

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

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

4.3.2 Activity-Based Mobility Generation in Detail

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

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

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

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

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

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

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

[19] SUMO Wiki: TraCI Python APIs https://sumo.dlr.de/docs/TraCI/Interfacing_TraCI_from_Python.html Access: September, 2020

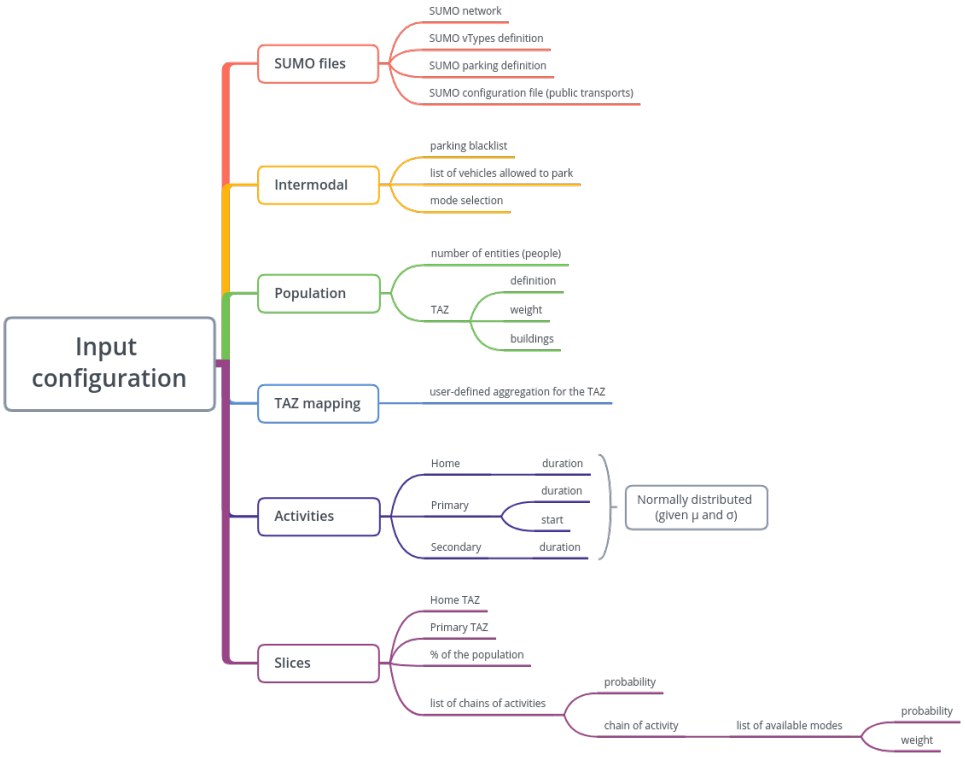


Figure 3: Input required by the activity-based mobility generator.

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

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

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

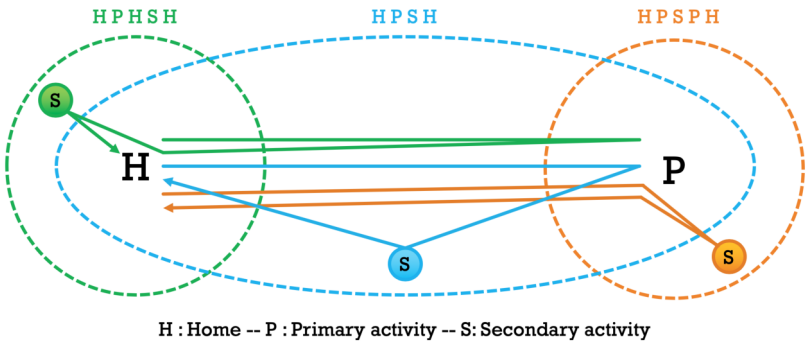


Figure 4: Diagram showing how the location for the Secondary activities is chosen.

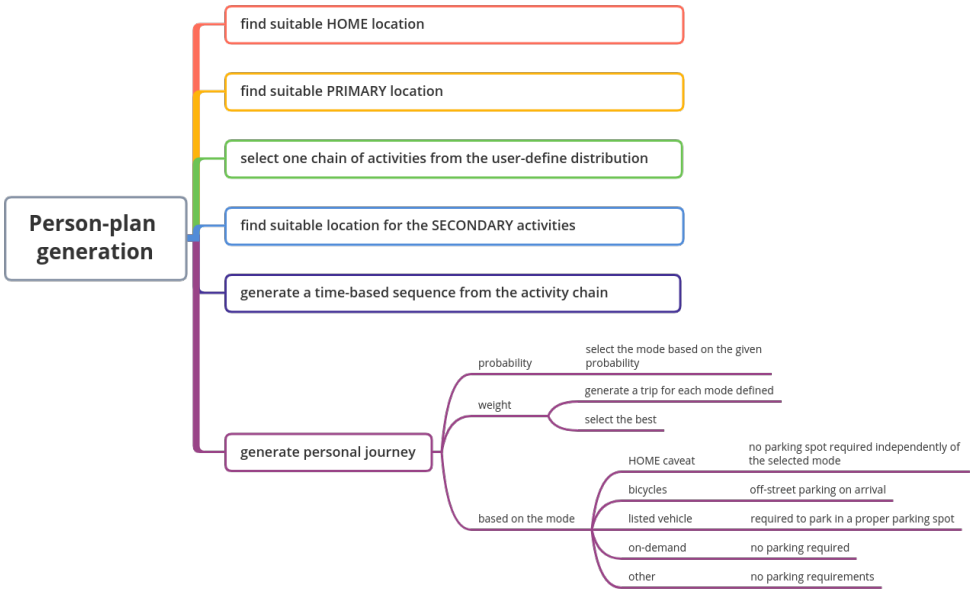


Figure 5: Overview of the implementation choices made during the activity-based mobility generation.

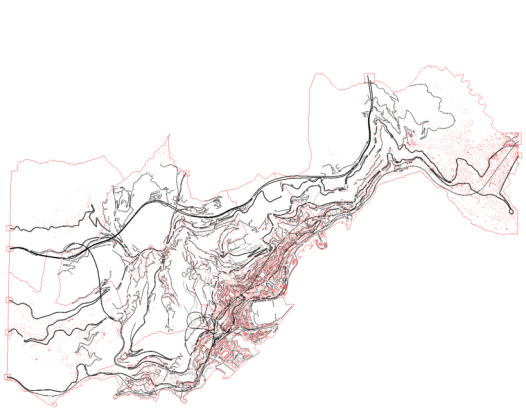
$S \rightarrow H$ (in green), the location of S is chosen to be within the area of the circle centered at H with a parametrized radius. Similarly, in the chain $H \rightarrow P \rightarrow S \rightarrow P \rightarrow H$ (in orange), the location for S is within the parametrized radius of the circle centered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

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

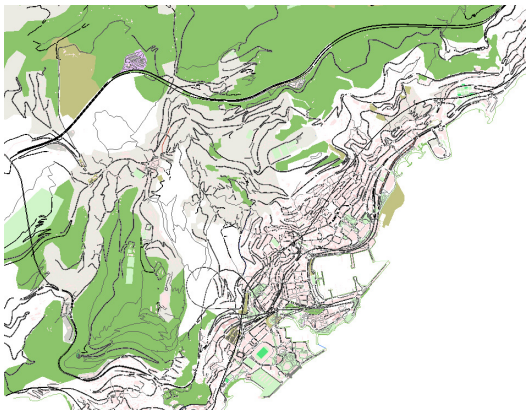
1. The Home location is computed based on the given TAZ. The exact location is selected by choosing a building based on its weighted probability, and the street accesses associated with it are saved.
2. The location for the Primary activity is computed following the same principle as the Home location. All the Primary activities in a single chain share this location.
3. The chain of activities associated with the plan is chosen from the distribution defined in the given configuration.
4. Following the sequence of activities, the areas of interest associated to each Secondary activity are computed. The process required to compute the actual location is similar to the one described for the other activities, but in this case, the area of interest, instead of being a TAZ, follows the definition presented in Figure 4.
5. An estimation of the time-based sequence of locations is generated using starting times and durations retrieved from the generic activities, where the Estimated Travel Time (ETT) used for the transit journeys is computed by SUMO based on a generic vehicle.
6. Based on the available transportation modes defined for the selected activity chain, the complete plan associated with a person is generated by composing consecutive trips.
   - The trip itself is generated using the TraCI API `findIntermodalRoute`[20] provided by SUMO.
   - In case the parameter associated with the mode is its probability, a single transportation mode is chosen. In case the trip is impossible with the given mode, the journey is discarded.
   - If the parameter associated with the mode is its weight, a trip is generated for all the given modes, their cost is multiplied by the given weight, and finally, the best mode (minimum cost) is selected for the trip. The default value defined in SUMO for the cost of a trip is the ETT. In order to model different user preferences associated with each mode of transport, the cost is multiplied by the given weight, skewing the selection of the best mode, if required.
   - Independently of the transport mode chosen, there is no requirement for parking associated with the Home activity. The assumption is that people have the means to park at home.
   - Each mode presents different parking behaviours:
     - Bicycles are parked off-street on arrival.
     - The vehicles listed in the configuration file that require parking will look for the parking area closest to the destination, leaving SUMO at simulation time to find an alternative parking area in case the given one is full.
     - On-demand vehicles have no parking requirements.
     - Other vehicles, such as emergency, have no parking requirements unless explicitly inserted in the configuration file.

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

[20] SUMO Wiki: findIntermodalRoute API https://sumo.dlr.de/docs/TraCI/Simulation_Value_Retrieval.html#command_0x87_find_intermodal_route Access: September, 2020



(a) MoST Scenario



(b) Principality of Monaco

(c) Adlershof (DE)

(d) Docklands (IE)

Figure 6: Examples of SAGA usage.

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

5 Use Case: Fast Prototyping

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

The workflow presented in this paper was developed while hand-crafting the MoST Scenario [9] (Figure 6a). The MoST Scenario covers an area of approximately 70 km^2 that includes three logical areas and 20 TAZ. It provides the locations of PoIs, more than 100 parking areas, the

shape and location for the buildings, and the elevation of buildings and streets. Buses and trains comprise the public transport network, with more than 150 stops and more than 20 routes. Based on previous experience and expert knowledge, to achieve the first working prototype of MoST Scenario, it took approximately 960 hours (one person working for six months). Only afterward, the iterative process of hand-tuning the multi-modal scenario could begin.

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

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

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

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

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

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

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

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

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

[21] Building City Dashboard https://dashboards.maynoothuniversity.ie/ Access: September, 2020
[22] Dublin Dashboard http://www.dublindashboard.ie/ Access: September, 2020
[23] MoST Scenario on GitHub https://github.com/lcodeca/MoSTScenario

6 Conclusion and Future Work

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

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

Acknowledgments

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

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

References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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