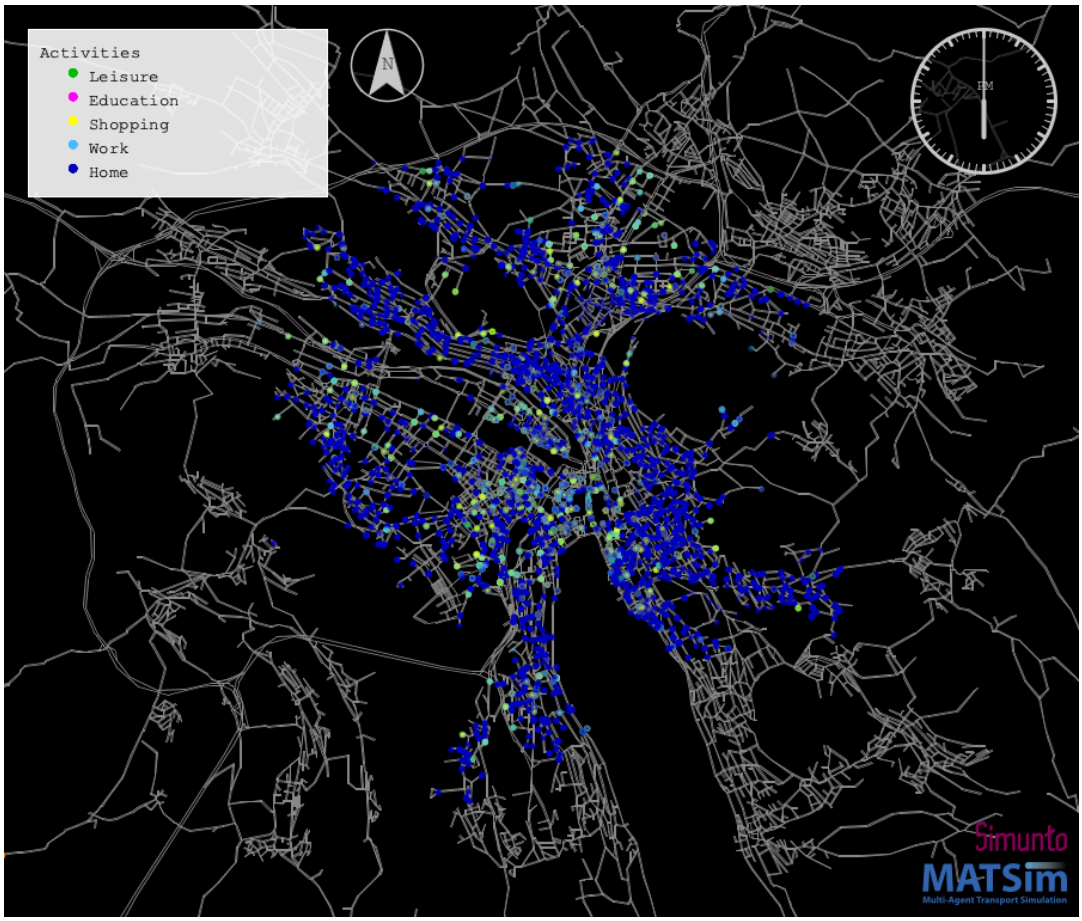


matsim-toy: a framework for generating minimal MATSim scenarios for arbitrary cities using Open Data

Davi Guggisberg Bicudo

Semester project
Institute for Transport Planning and Systems **May 2019**

Institut für Verkehrsplanung und Transportsysteme
Institute for Transport Planning and Systems

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Table of contents

Semester project

matsim-toy: a framework for generating minimal MATSim scenarios for arbitrary cities using Open Data

Davi Guggisberg Bicudo
IVT
ETH Zurich

CH-8093 Zurich

E-Mail: davig@ethz.ch

May 2019

Abstract

This report describes a framework for creating basic MATSim scenarios exclusively from open data. MATSim is an agent-based microsimulation framework designed for large-scale transport models. MATSim models typically require extensive data, its demand modelling in particular relies on data highly specific to the model's geography. For this reason, no tools automating tasks in demand modelling for MATSim are available. The proposed framework addresses this problem by relying on data available at a global scale. OpenStreetMap data is used as the basis while high level open data APIs of global scope provide demographic data at the national data. The developed tool is simple to use and requires no additional input besides the city and country names. Modelling complex microsimulations with coarse data can result only in very basic models, while the process heavily relies on strong assumptions and arbitrary parameters. Nevertheless, the framework is able to successfully generate models with all the basic elements required by MATSim and the resulting simulation produces feasible scenarios.

Keywords

MATSim; OpenStreetMap; Transport Modelling

Preferred citation style

Guggisberg Bicudo, Davi (2019) *matsim-toy*: a framework for generating minimal MATSim scenarios for arbitrary cities using Open Data, *Semester Project*, IVT, ETH Zurich, Zurich.

Introduction

MATSim (Multi-Agent Transport Simulation) is a transport modelling framework that allows microsimulation of travel behavior for large scenarios. MATSim has been used for developing both very large and highly realistic transport models (Horni, Nagel, & Axhausen, 2016). The level of detail modelled and simulated with MATSim usually requires large datasets and high time investments on the developed of its models.

Researchers and planners often have to work for a long time before they are able to see any results. This initial overhead can be seen as unavoidable due to the nature of transport models, but it may also be detrimental for small scale applications and limiting the access to newcomers willing to learn and experiment with MATSim.

With the goal of reducing this overhead, the present work proposes the creation of a methodology and software framework for the creation of toy MATSim models, which allows planners and researchers to create basic MATSim models for arbitrary cities of the world without having to provide any input data. This scenario can then be used as an initial baseline that can be incrementally improved and tested upon.

1.1 MATSim models

MATSim reaches equilibrium when each of the persons in its population, modelled as utility-maximizing agents, no longer are able to unilaterally improve their utilities. Their possibilities for improvement include switching routes to less congested roads, switching modes and changing departure times (Horni et al., 2016).

The MATSim engine simulates how well a transport supply, composed of the road network and the public transport services, is able to serve a transport demand, composed of a synthetic population of agents.

Transport models developed with MATSim usually involve large efforts of data collection and processing, while automation is limited since data formats and requirements are specific to each model (Hörl, 2017; Kickhöfer, Hosse, Turner, & Tirachini, 2016; Ziemke & Nagel, 2017).

1.1 Data requirements

Transport models usually require socio-demographic data at some level of disaggregation over the model's area, which can be at the zonal, household or at the individual level. This data usually comes from travel surveys and censuses, which are conducted typically every 5 or 10 years. When available, especially with high disaggregation and coverage, this type data provides the best information sources for modelling the travel demand.

Even though in many cases such data is publicly available, their structure and conceptual framework is custom to each location's particular needs and characteristics, making it impossible to create a fully automated tool for generating detailed synthetic populations for arbitrary cities. This leads to the large effort required to process the data and model the required transport demand.

For modelling the transport supply, data from the road network and public transport (PT) service is required. This data, when available, can be converted to formats usable by MATSim in a more direct way. For road networks it is common to use road networks extracted from OpenStreetMap (OSM) whereas for PT services the use of General Transit Feed Specification (GTFS) data is common.

1.2 MATSim modelling tools

The challenges imposed by the complexity and data requirements of microsimulation transport models have motivated the creation of tools to ease the modelling process. For transport supply, due to its more general character and common input data format, tools developed for this purpose were more successful. For instance, the pt2matsim framework (Poletti, 2016), which can be used to generate a MATSim network from OSM data, generate the schedule from GTFS or HAFAS and map-match both with good accuracy. Another example is the JOSM MATSim plugin which allows effective visualization of the network and PT service as well as manually editing it (Kühnel & Zilske, 2019).

For transport demand on the other hand there hasn't been any substantial development, and the main reason is the absolute diversity of each model's requirements and input data.

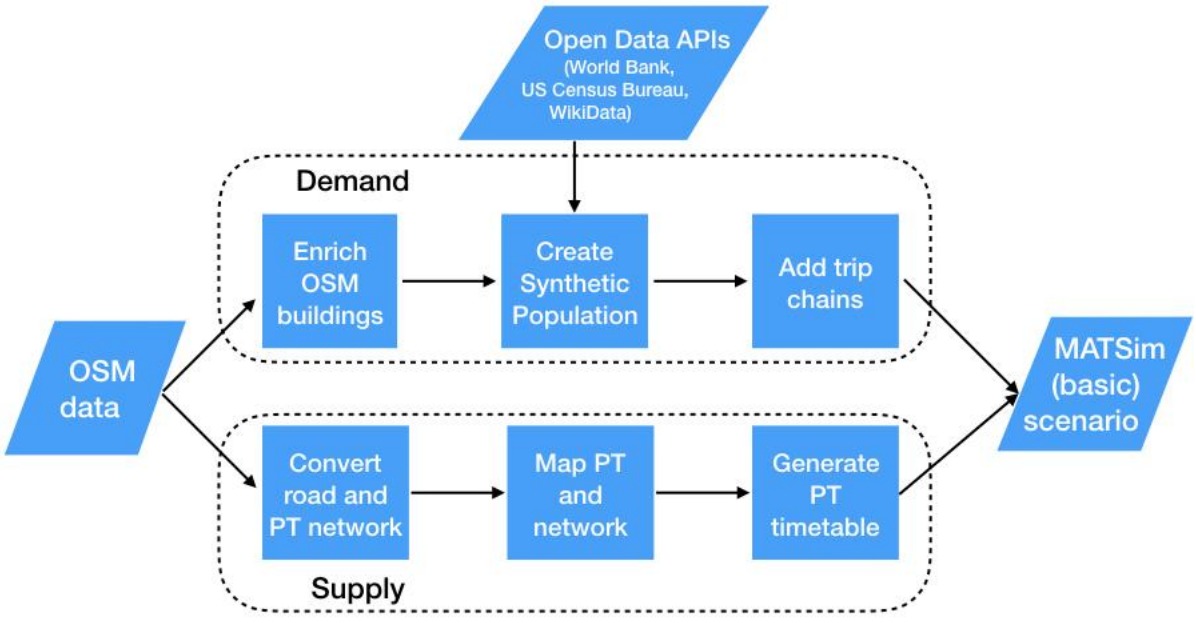
2 Methodology

The development of a MATSim model, in its simplest terms, requires a population, made of synthetic agents with a trip chain, and a road network. Very often a public transport schedule is also implemented. The population represents the transport demand whereas the road network and PT schedule the transport supply.

In order to build a MATSim toy model for an arbitrary city, these 3 elements have to be produced with data openly available and of easy access. This constrains the sources to global open data APIs. Global APIs offer only scarce data at the city level, so several assumptions and simplifications will be needed to build the models. An exception is OpenStreetMap (OSM), which offer quite detailed geospatial vector data at the level of buildings and roads, although with varying data completeness and accuracy.

Figure 1 provides an overview of the proposed methodology, while the data and steps to create the models are described in the subsequent sections.

Figure 1: Proposed framework



2.1 OpenStreetMap

In OSM, features are modelled via nodes and ways, with metadata included in the form of key-value tags. A node could be for instance a bus stop and the street in front of it a way. Closed ways are used to represent objects whose shape are polygons, such as buildings, forests and political borders. Besides nodes and ways, OSM supports also relations, which are collections of OSM objects such as nodes, ways and other relations, and are used to represent complex objects such as public transport routes and turn restrictions.

Originally, transport models modelled their needed transport graphs from data provided by official authorities. This type of data usually comes for each location in a different format, thus requiring custom conversion for each new transport model. Creating large-scale models can also be an issue when datasets from different authorities have to be combined. For these reasons, data from OSM can be used as an interesting alternative. For instance, Zilske (2011) and Dallmeyer, Lattner, & Timm (2014) describes the usage of OSM road networks in the creation of different approaches to transport modelling. OSM can also be used to partially model public transport supply, as described in the method proposed by Poletti (2016).

Zilske (2011) further discusses the potential uses of other OSM features in transport modelling, particularly regarding the assignment of travelling agents to OSM buildings instead of randomly spreading in a survey zone's polygon. Part of their suggestions are practically implemented in this work.

2.2 Demographic data

MATSim requires a very detailed definition of its travelling agents and their daily travel patterns. Typical data for setting up MATSim synthetic populations include census, travel surveys, business cadasters and stated preference surveys, all as spatially accurate as possible (i.e. every entity or activity ideally has a coordinate attached).

On the other hand, provision of open global demographic data at the city scale is limited for several reasons. Diversity of methodologies, lack of resources and privacy concerns are some of them. Mostly what is available comes at the national level, such as the databases of the World Bank and of the United Nations. Transport and travel behavior data is not available at all.

In order to create models for arbitrary cities, the data has to come from global, public, open data APIs. Since there is a limited amount of this type of services and their offerings are very limited, the best options will be used and the rest of the information needed will have to be estimated or assumed and the modelling simplified.

The absolute population of a city is not available in any official data source, but fortunately Wikipedia pages for cities usually has this information in a context box which can then be queried via the Wikimedia API. Since Wikipedia is an informal source, no coverage or precision claims can be made.

Demographic distribution of cities aren't available. At the national level, databases such as the ones provided by the UN or the United States Census Bureau can be used. For this work, the latter was preferred for its ease of use.

Finally, to estimate the employment status of agents, the unemployment rates at the country level provided by the World Bank API can be used.

2.3 Modelling of travel demand

As discussed before, the data available is very scarce, so several assumptions and simplifications will be made, explicitly and sometimes also implicitly due to the clear limitations of the approach.

OSM is the only data source providing a higher level of detail, so it will be used as the base for the methodology applied. Particularly useful are buildings, available in many cities and modelled as polygons representing their floor plans. Buildings are often tagged with additional information such as its use and height.

The method proposed for modelling the travel demand is composed of three stages:

1. Enrichment of OSM building data
2. Generate the basic synthetic population
3. Generate simple trip chains to each person

In the first stage, missing heights are estimated in order to obtain built volumes and buildings are classified in different usage categories. The second stage uses the building volumes, public transport stations and basic demographic data to estimate the number of dwellers and

workplaces in each building. Finally, in the third stage, agents are assigned a simple trip chain according to their demographic attributes and home locations.

These stages will be described in detail in the following sections.

2.3.1 Enrichment of OSM building data

This stage can be further divided into three steps:

1. Pre-processing of OSM raw input data
2. Estimation of building heights
3. Classification of building usage

The first step is straightforward and consists of the tasks of converting from OSM's raw XML format to a more usable data frame format, filtering buildings to the area of interest and filtering relevant tags. This first step may also include down sampling of the amount of buildings which reduces total computational time.

The other two steps are described in the following subsections.

Height estimation

Often only a small share of the buildings contains height information. Some contain instead the number of floors, which can then be multiplied by a floor height factor to obtain an estimate of the building height.

For the remaining buildings, a K-nearest neighbors (k-NN) algorithm is applied to obtain a regression of the heights. In this case, the k-NN regression obtains the height of a building by averaging the heights of its K nearest neighbors. If the dimensions provided to the algorithm would be the building's x and y coordinates, the estimation would simply return the average height of the K (physically) closest buildings that contained height information. It is possible though, to improve the regression by including additional dimensions such as building area and perimeter.

Building usage classification

This step attempts to classify buildings into 4 broad categories:

a. Exclusive residential usage ("residential")
b. Exclusive workplace usage ("workplace")
c. Mixed workplace and residential usage ("mixed")
d. Idle buildings (e.g. lighthouse or water reservoir) ("idle")

Workplace usage is understood as locations where people work, where their specific uses can be commercial, industrial, government, institutional, etc.

The classification applied is a simplified version based on the method proposed by Kunze & Hecht (2015).

Buildings are classified based on their areas and by the OSM tags of the buildings themselves and from eventual overlapping point of interest (POI) nodes or land-use polygons. In this work the tags identified by Kunze & Hecht (2015) are used (available as a table at the Appendix of their work).

The classification of the buildings follow the order of definability: *workplace*, *mixed*, *idle* and *residential*.

Buildings are classified as *workplace* when they fall into any of the cases:

- They have a building, amenity or shop tag defined as strictly non-residential, e.g. *library*, *fire_station*, *camp_site*, etc.
- The building overlaps a land-use polygon tagged as one of *industrial, farmland, farmyard, forest, grass, meadow*.
- The building's area is above a certain threshold.

The remaining buildings can then be classified as *mixed* if they fit into any one of the following:

- They have an intersecting POI, which are defined as nodes containing any relevant usage tag (from Kunze & Hecht (2015)). For the intersection, a 2m buffer was used as recommended by Kunze (2013).
- They have a *shop* or *amenity* tag.
- Their *building* tag does not have any of the following values: *yes, residential, house, detached.*

After this, the final classification is based on a simple rule, buildings below a minimum usable area are considered *idle* and the rest is classified as *residential*.

Finally, the buildings receive workplace and residential built volume attributes according to their usage classification and an arbitrary factor for mixed use. The factor is used to determining the share of the built volume of *mixed* buildings should be assigned to *residential* usage and the built volume is simply the floor area multiplied by the building's height.

2.3.2 Creation of the synthetic population

In this stage, the demographic data detailed in section 2.2. will be applied to the buildings from the previous stage, in three steps:

1. Accessing and pre-processing demographic data
2. Estimate number of residents and workplaces per building
3. Generation of synthetic population

The first step is straightforward and consists of accessing the different API systems, pre-processing their outputs and combining them to obtain the city's total population, employed population, rate of unemployment and demographic distribution.

The other two steps are detailed in the following subsections.

Residents and workplaces estimation

In this step, the total population and total employed population values will be distributed among the buildings. The underlying assumption is that the number of workplaces in the city is exactly the same as the employed population, which means there are no cross-border commuters. The total employed population will further on be mentioned as total workplaces.

The distribution uses a rasterized, areal interpolation approach, where a grid is laid over the city and the number of residents and workplaces are estimated *per cell*, and finally assigned back to the buildings according to their volumes.

Cells' weights are defined by a simple PT accessibility measure and built volume.

The approach used in this stage is based on Bakillah, Liang, Mobasheri, & Arsanjani (2014) and can be partly defined as areal interpolation with a dasymmetric approach using point (PT stations) and polygonal (buildings) vectors as ancillary data.

To calculate the cell weights, a raster grid representing normalized PT accessibility in the city is summed with a normalized grid representing built volume. The resulting grid is once again normalized so that all cells sum up to 1, and each cell multiplied by the total population or total workplaces.

The accessibility value of each cell in the first raster is based on the catchment area of stops as well as walking distance to it. The catchment area and basic accessibility value provided by a stop depends on the PT mode (e.g. a train station provides accessibility to *more* and *higher* accessibility than a bus stop). This value is then multiplied by the inverse normalized distance between the cell and the station, to account for decreasing accessibility at longer walking distances.

The building volume raster is simply the normalized sum of the volumes of buildings within a cell (the building's centroid decides to which cell a building belongs). The workplaces estimation considers workplace volume whereas population estimation consider residential volume.

To assign the cell values back to the buildings, a weighted distribution, based on the building's volumes, is applied.

Creation of the synthetic population

In this second step, a table of persons is generated, such as the one displayed in Table 1, where 3 example persons are detailed.

Table 1: Example agents generated for synthetic population

Age	Sex	Employed	Car Availability	Has License	Studying	Home ID	Home_X	Home_Y
6	m	FALSE	never	FALSE	TRUE	145678	2673756	1220875
53	f	TRUE	always	TRUE	FALSE	125328	2673238	1220298
27	f	FALSE	never	FALSE	FALSE	175179	2673239	1220425

The choice of attributes for the population follows common MATSim requirements and were defined as follows:

- Employed: unemployment rate is applied to everyone between an arbitrary minimal and maximal working age. Everyone under or above these thresholds are considered either full-time students or retired.
- Studying: everyone above a minimal student age and below the minimal working age. Working and studying is thus mutually exclusive.
- Car availability: an arbitrary car availability rate is applied to everyone between also arbitrary thresholds of minimal and maximal driver ages. In MATSim, values may be *never, always* or *sometimes*, but here only *never* and *always* are used.
- Has license: *false* if car availability is *never*, *true* otherwise.
- Age and sex: direct representation of the country's demographic distribution.
- Home (ID and coordinates): sampled from *buildings* according to the value estimated in previous section.

2.3.3 Generation of trip chains

For the generation of the trip chains, in practice one would generally use travel surveys, which are datasets containing several inter correlated variables across different dimensions. Since this kind of data is not present for this case, only very simplistic trip chains can be assigned to the persons. The travel behavior of the resulting population is finally composed of mostly entirely arbitrary dimensions with limited variability.

The trip chains generated contain at most two activities besides home, where students and workers may have a secondary activity (e.g. leisure or shop) and unemployed or retired receive two secondary activities.

Under the previous assumptions, this stage's methodology can be described in 3 steps:

1. Calculating the travel time matrix between all possible origins and destinations
2. Assign activity locations to each person
3. Calculate a possible travel diary for each person

The first step consists of calculating the free-flow car travel times from all buildings with residential use to all buildings with workplace use. The graph used is the OSM street network and the weights used for the shortest path calculation are the typical MATSim road speeds.

In the second step, agents receive an activity location for their primary and secondary activities. For primary activities, students are assigned to the closest building with one of the respective OSM tags marking it as a school and workers have their workplace weighted-sampled from a random building with workplace usage, where the weights correspond to the number of workplaces in each building. Secondary activity locations are sampled also from buildings with workplace usage, but their weights are adjusted by the inverse distance to their home locations.

In the third step, agents receive a travel diary defined as follows:

- All students start and finish school at the same time.
- Workers' have their start time and duration sampled from normal distributions with arbitrary parameters.
- Everyone that has a secondary activity performs it after their primary activity, where the durations are also sampled from normal distributions.
- The mode of transport depends on the person's car availability, age and free flow car travel time to their primary activity.
- The free flow car travel time and a factor accounting for their mode's relative speed is used to estimate the time agents leave home and are expected to return.

2.4 Modelling of the transport supply

The transport supply, composed of the road network and of the public transport schedule, is modelled based on the data available in OSM and some arbitrary factors.

Direct conversion of OSM's street network to MATSim is now common practice and easily achieved with the pt2matsim tool (Poletti, 2016). The same tool is also used to generate PT schedules from common schedule data formats (i.e. GTFS and HAFAS) and to map them to the previously generated road network. Depending on the quality of the input data the resulting mapped schedule approximate very well the actual PT service (Poletti, 2016).

Fortunately, OSM supports mapping of PT services through special relations connecting stations to form PT routes and lines. From this data, the pt2matsim tool can generate schedules, although without timetable information which is missing in OSM.

Thus, in order to create the PT timetables to the proposed models, strong assumptions and simplifications have to be drawn.

In simple terms, a timetable is composed of departure times at each stop and the needed travel times between stops. If one assumes a constant service frequency throughout the day, calculating the travel times between stops and applying this frequency creates a theoretically viable transit schedule. It is clear that, in reality transit schedules have numerous additional physical constraints, such as crew and vehicle scheduling, but since these aren't implemented in MATSim, the schedule generated in this manner is a viable one.

If in this process some issue with the routing between stops occur (due to e.g. input data problems), the Euclidean distance is used combined with a beeline factor to calculate the travel time.

3 Framework usage

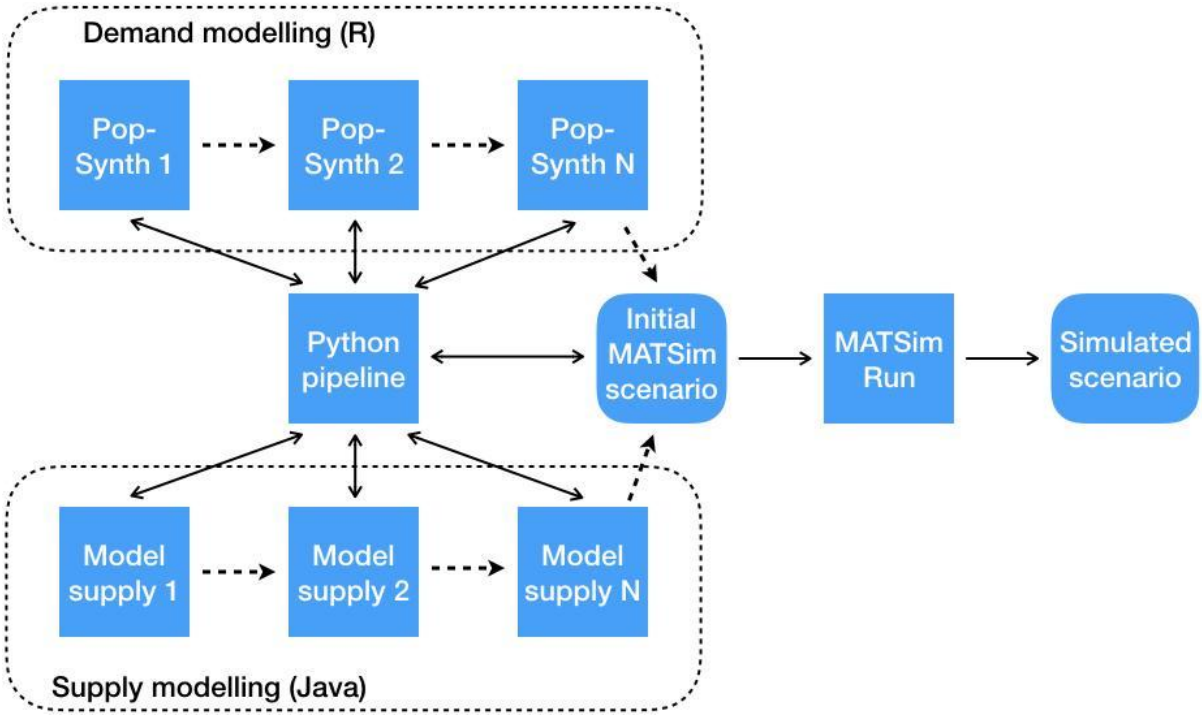
3.1 System design

The developed framework depends on several software components, responsible for different tasks in the modelling.

All methods described in the previous chapter, with exception of the last section, were implemented using the R language for statistical computing. Transport supply modelling, as described in the remaining of the previous chapter, was implemented mostly with the *pt2matsim* framework, which is in Java (the language MATSim is also developed in). In order to chain all the steps described in the proposed methodology in an automated and reliable way, an existing framework currently in use at the Institute for Transport Planning and Systems (IVT) was used, which is implemented in Python.

The integration of the different programming languages also requires integrating different dependency management tools for installing all the required libraries with their own requirements. Figure 2 displays the design of the system.

Figure 2: *matsim-toy* system design



The Python pipeline works by chaining the steps for the scenario development and caching the outputs of each step. This way it is not needed to re-run the entire process if one step of it must be changed. For instance, requesting only the last step to be run will automatically trigger all previous steps that don't have any stored cache.

3.2 Setup and requirements

Running the proposed framework requires Java (version 1.8), Maven (version 3), R (version 3.5) and Python (version 3.6), besides a number of R libraries and a few Python libraries.

A setup script is provided that handles the installation of all needed packages in a separate environment via Miniconda3. This script makes sure everything is in place before actually starting the modelling. To use it, simply run:

source setup/setup.sh <target directory> <software to be installed>

Where the first argument is the folder where the needed files will be downloaded and installed and the second argument is a comma-separated list of the software required (e.g.

java,maven,conda for all options, without spaces). R, Python and all required libraries are installed with help of Miniconda3 in a separate environment (the target directory) which can later be deleted for a clean uninstall.

Alternatively, a manual installation is also possible. If the required software are available, it is only needed to install a few Python libraries (tqdm, pyyaml, pandas and requests) and pacman, a package manager for R which will handle later the installation of the remaining R libraries on-the-run.

3.3 Quick start

1) Download (or clone) the GitHub repository.
2) Run *source setup/setup.sh env_dir java,maven,conda*
3) Run *source setup/activate.sh env_dir* (tests and activates the conda environment).
4) Run *python run.py city_name country_name*

This will produce a MATSim toy model for city_name (part of country_name) and run it for 100 iterations, all with default configuration. Default model creation configuration can be seen in the file *config.yml* and the default MATSim simulation configuration can be seen in *matsim/config.xml*.

If none of the required software or libraries were available, installing can take several minutes, depending on the speed of the internet connection.

Running the framework takes also some time, depending on the size of the scenario. The example provided is a 10% scenario for Luzern (Switzerland), a relatively small city with ~90'000 inhabitants, which takes about 11 minutes to build and 23 minutes to run 100 iterations in MATSim in a 16GB macOS i7.

3.4 Configuration

The configuration for running the different steps is stored in a *yml* file. The example *config.yml* contains the configuration for the initial example and contains minimal configuration and only the final step, which means all other steps are run with default arguments. The *config_full.yml* on the other hand shows all steps available with their respective arguments. The argument values are the default so running the framework with *config.yml* and *config_full.yml* produces

the same results, although running a second time would only trigger the last step in the former while re-running the entire pipeline in the latter.

3.5 Alternative uses

The proposed framework's intended use is mainly to create and run a toy MATSim scenario for any given city, but the steps implemented may be used in different ways and adapted to other needs. Most importantly, the scripts may be edited or replaced entirely in order to accommodate additional input data, still taking advantage of the other steps implemented and the pipeline's chaining and caching capabilities.

4 Results

This chapter present the results of the proposed modelling framework, after applying the methodology described in Chapter 2 and running the framework as described in Chapter 3.

A 10% scenario for the city of Zurich, Switzerland is presented, where the intention is to show that a feasible model was created but without making any claims to its degree of realism, since it is a very simple toy model.

4.1 Demand modelling

Figure 3 displays intermediate results of the demand modelling, showing buildings in Zurich after the estimation of residents and workplaces. Only 10% of the buildings are present due to the 10% sampling. The highlighted building is one of the city's universities and was classified as *workplace*, with 44 workplaces for the chosen sample size (would have been estimated as 441 in the 100% scenario).

The generated population has ~44 thousand agents. Figure 4 displays the age histogram for the synthetic population (in 5-year bins) whereas Table 2 displays its key statistics.

Figure 3: Results after OSM building data enrichment

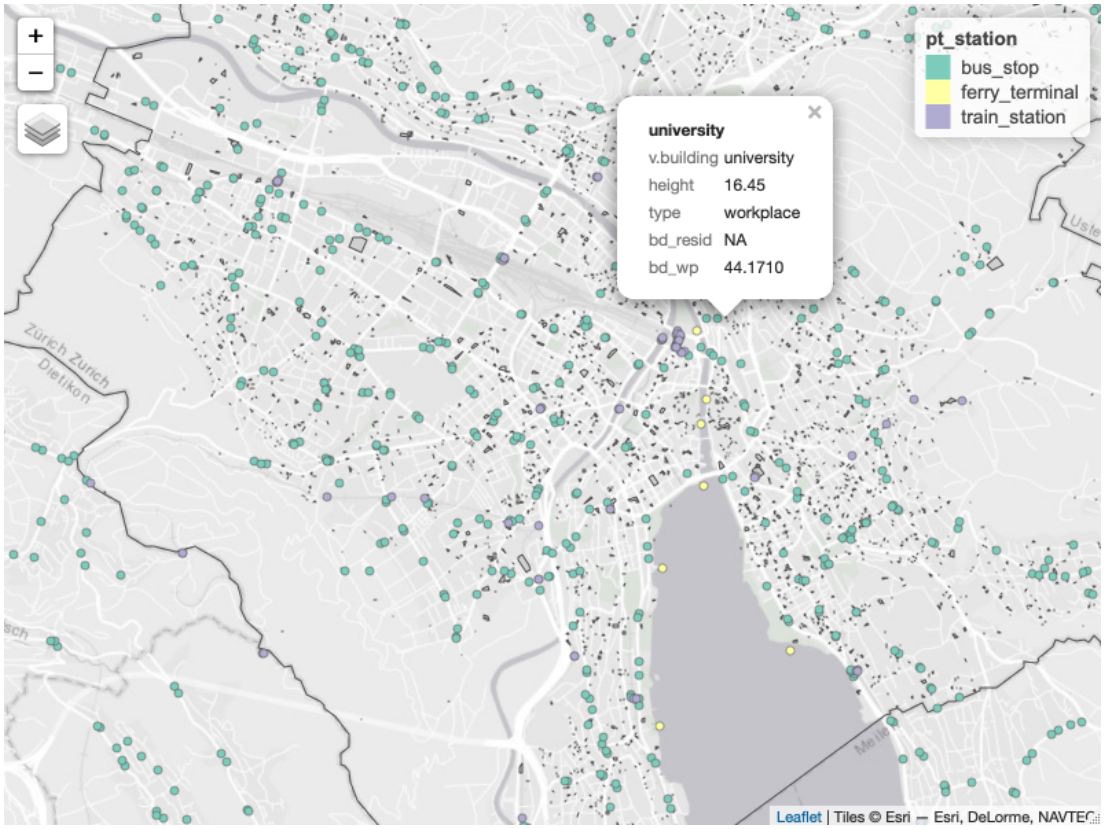


Figure 4: Age histogram for the synthetic population

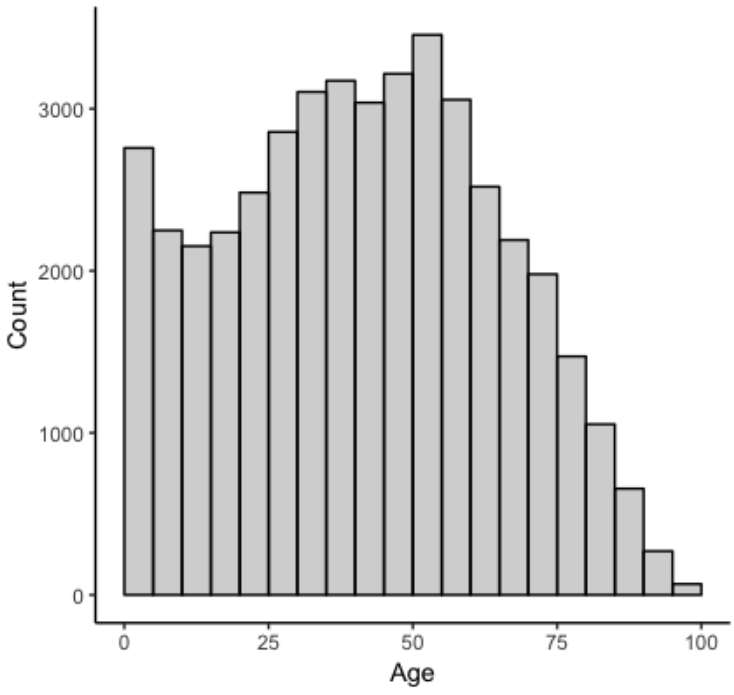


Table 2: Key statistics of the generated synthetic population

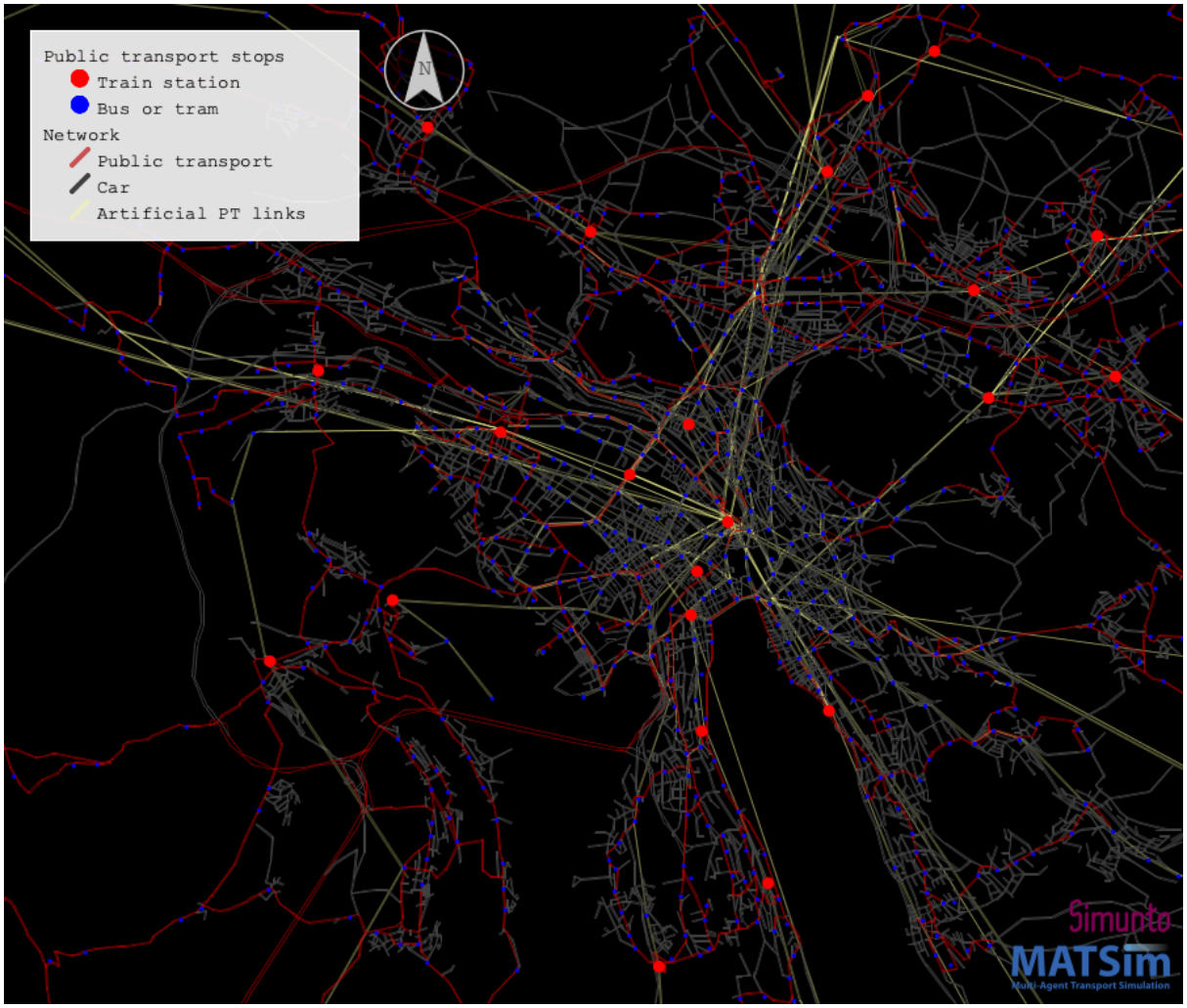
Attribute	Values
Employment status	Employed - 60.2% Student - 18.2% Retired - 17.5% Unemployed - 4.1%
Car availability	Always - 51.3% Never - 28.2% Other (too young or old) - 20.5%
Secondary activity	None - 45.3% Shopping - 43.9% Leisure - 10.8%
Main mode	Car - 35.7% Public transport - 31.7% Walk - 23.2% Bike - 9.4%

It is important to note that the values in Table 2 are generated based on the country demographics and country unemployment value, besides arbitrary parameters from the methodology, thus any city in Switzerland would have the same shares and differ simply in total population.

4.2 Transport supply

Figure 5 displays the generated transport network, with links used by PT highlighted. The links marked in light yellow are generated by pt2matsim whenever a suitable route for a PT connection is not found.

Figure 5: Transport supply for Zurich



4.3 MATSim Simulation

The simulation was run mostly with default MATSim parameters. A set of default parameters for running MATSim is defined in a matsim/config_template.xml file, which can be manually changed. For the agent's plan innovation strategies, in 80% of the iterations a simple logit-based plan selection is performed, 10% of the times they reroute and in the remaining 10% a subtour mode choice is applied.

Figure 6 shows the visualization of the simulation results at 8:30, where it can be seen the usual traffic congestion in the center but the highways actually quite free due to lacking cross-border demand.

Figure 6: Visualization of traffic at 8:30, after 100 iterations

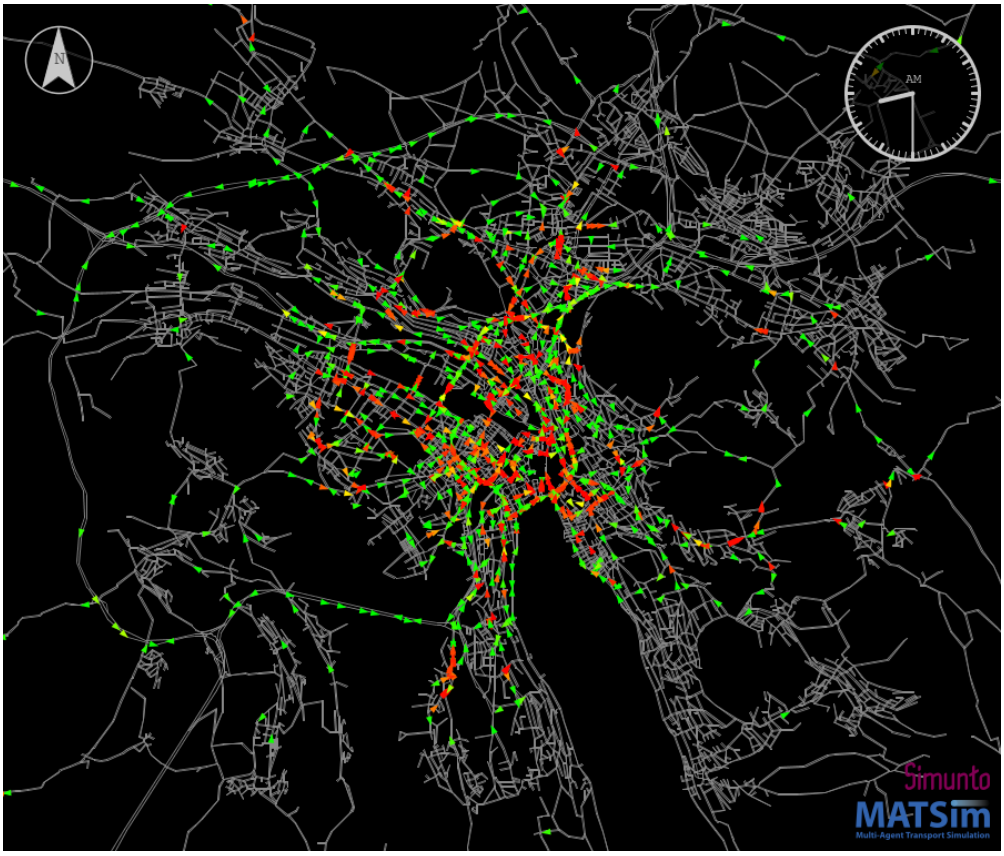
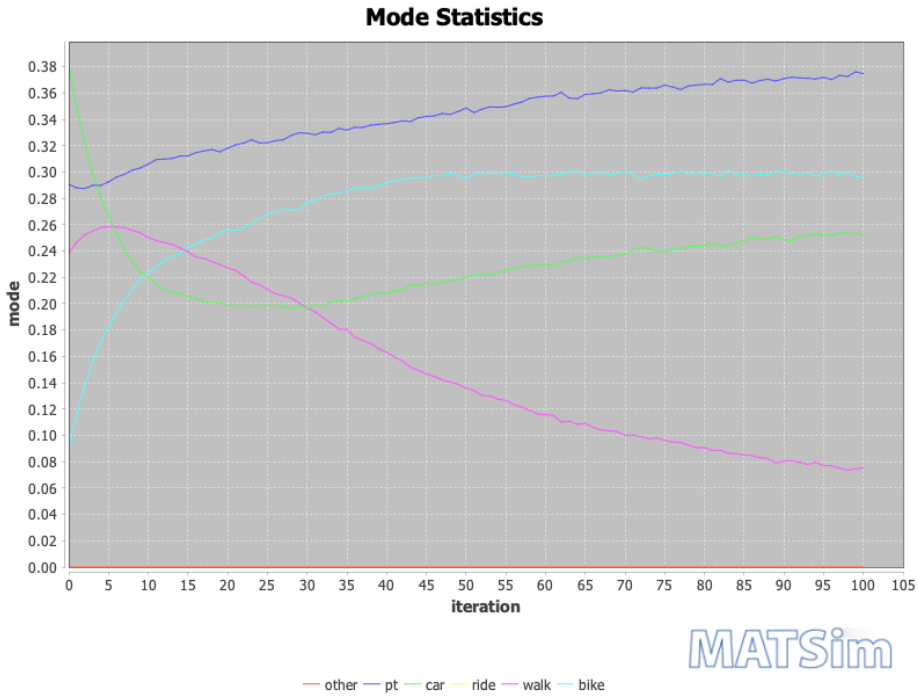


Figure 7: Development of the mode share through 100 iterations



The initial state evolves in terms of routes and modes throughout the iterations, and although the aggregate measure of the plan scores reach an equilibrium around iteration 50, the modal split does not, as shown in Figure 7.

The spatial distribution of modal choices also shows expected results, as displayed in Figure 8 where trips originating at the borders of the model are more often done by car than in the center.

Figure 8: Spatial distribution of car trips share in simulated scenario

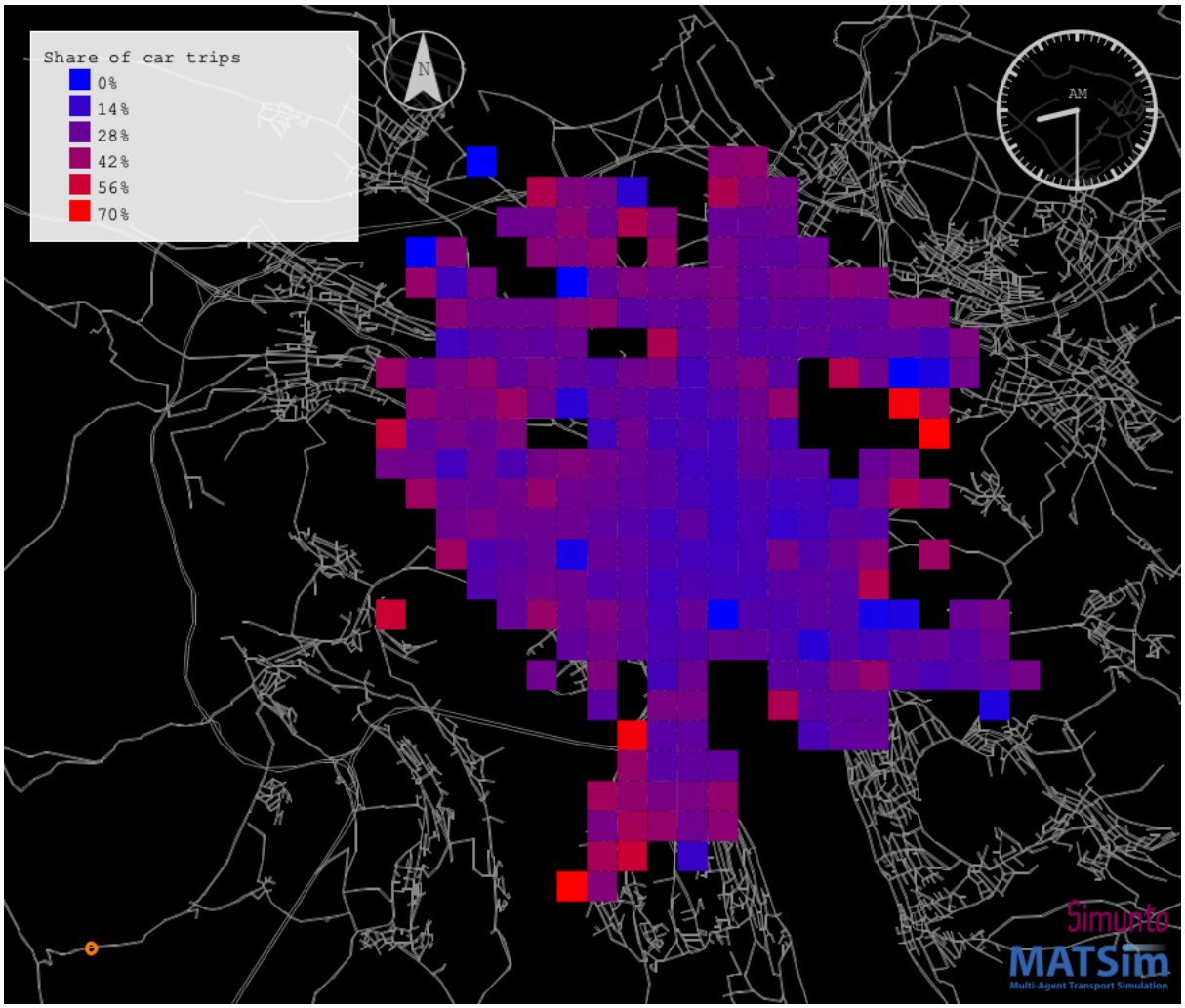
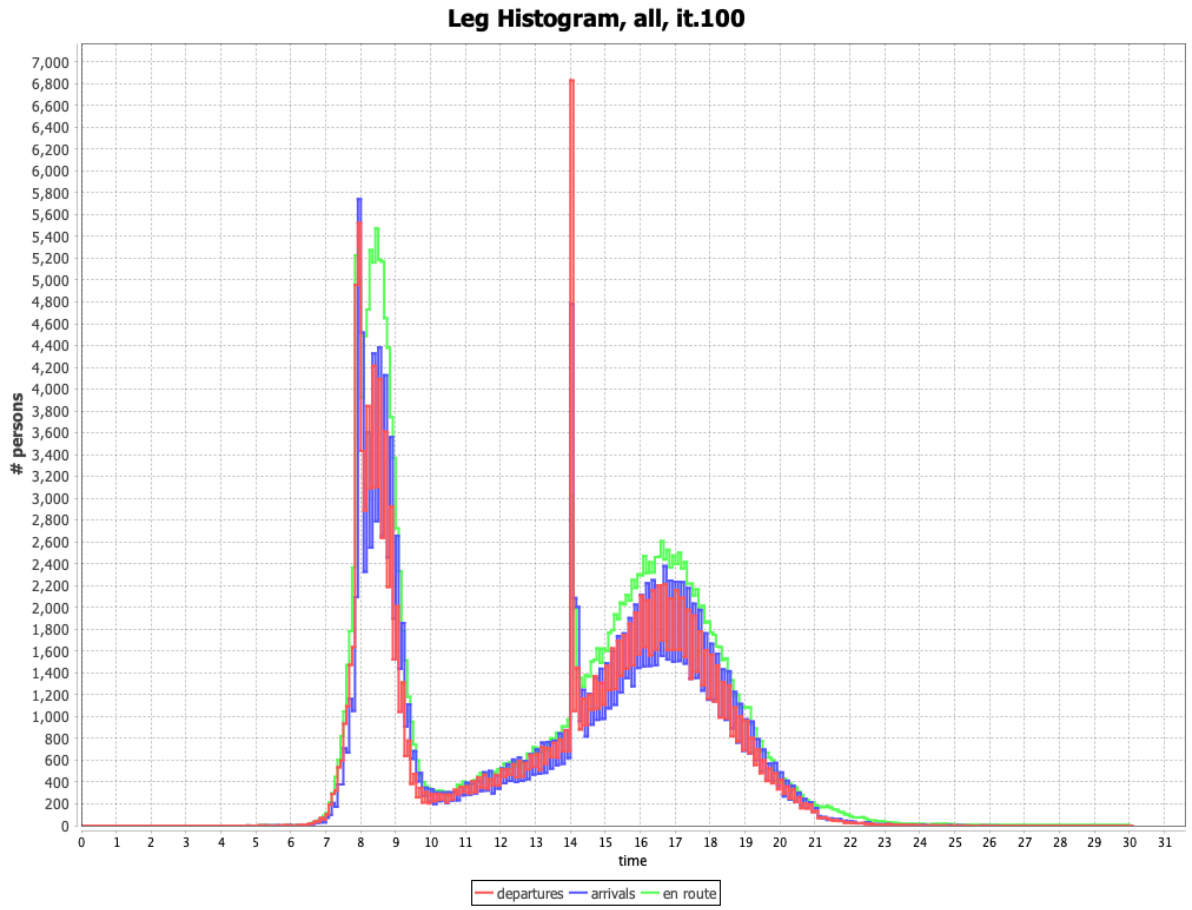


Figure 9 displays the leg histogram of all modes, showing clearly the departure peaks for students at 8:00 and 16:00, while the remaining trips are more distributed throughout the day. The afternoon peak is noticeably much smoother that the morning peak, which is due to the accumulated effects of the gaussian distributions.

Figure 9: Leg histogram of the simulated scenario



5 Discussion

The development of the proposed framework lead to some interesting reflections, which are summarized in this chapter.

Most important of all, it became clear that no matter how sophisticated methods may be available, data will always be crucial for transport modelling, especially when one considers transport as a complex system. In particular, when considering the relationship of transport with the social and geographical system where they are embedded, the input data gains special relevance. Unfortunately, such data remains, in their format and conceptual framework, very specialized to local needs, which prevents the development of more sophisticated tools for automating transport demand modelling. In the future, new data collection and processing

methods might be able to reduce this difficulty, although privacy concerns may actually make it even more challenging.

Also noteworthy is the fact that, even when the goal is to create minimal models with very few data, MATSim modelling still requires a substantial amount of work. The task of replicating the complexity of microsimulation models from coarse data can be quite intricate, especially when it comes to generating the travel diaries. The large number of arbitrary parameters used while still generating very limited travel diaries indicate it.

In terms of the input data used, OSM is traditionally used for supply modelling, but in this work, it was possible to see that it can also be effectively used in demand modelling. In cases where fine-grained spatial distribution of population and activities isn't available, OSM can be valuable.

6 Conclusion

The present work proposed a framework for creating toy MATSim models for arbitrary cities with minimal input data, data which is actually collected from public APIs thus requiring no user input besides the name of the city itself.

The generated models contain all the basic elements of a MATSim scenario and are in themselves technically complete and feasible, although no claims are made on their degree of realism.

The goal of providing a methodology and a framework for creating toy MATSim models was achieved, even though several clear limitations can be pointed. Nevertheless, planners and researchers may use the proposed framework to produce initial toy models which can help not only in getting used to MATSim models but also to provide a minimal baseline for their planned scenarios.

In the future, the framework can be extended by increasing its configurability and improved estimation methods that could provide more realism to the scenarios. Also promising is the increasing availability of open data APIs, which may in the future be included to produce more sophisticated scenarios. Re-implementing the R scripts in Python or Java would also be a practical improvement to reduce the framework’s complexity.

Acknowledgments

I would like to thank Prof. Axhausen and Sebastian Hörl for the ideas, support and counseling provided throughout the execution of this semester project. I would also like to thank Dr. Marcel Rieser for kindly providing a license for Simunto Via, used for some of the visualizations in Chapter 4.

7 References

Bakillah, M., Liang, S., Mobasheri, A., & Arsanjani, J. J. (2014), Fine-resolution population mapping using OpenStreetMap points-of-interest, *International Journal of Geographical Information Science*, *28*(9), 1940–1963.

Dallmeyer, J., Lattner, A. D. & Timm, I. J. (2014), GIS-based traffic simulation using OSM, *In: Data Mining for Geoinformatics*, Springer, pp. 65–82.

Hörl, S. (2017) A MATSim scenario for Autonomous Vehicles in La Défense and Île-de-France, *Working paper Institute for Transport Planning and Systems*,10XX, Institute for Transport Planning and Systems (IVT), ETH Zurich, Zurich.

Horni, A., Nagel, K., & Axhausen, K. W. (2016), The multi-agent transport simulation MATSim, Ubiquity Press, London.

Kickhöfer, B., Hosse, D., Turner, K., & Tirachini, A. (2016), Creating an open MATSim scenario from open data: The case of Santiago de Chile, *VSP Working Paper*, 1–22, TU Berlin, Transport Systems Planning and Transport Telematics, Berlin.

Kühnel, N., & Zilske, M. (2019), JOSM MATSim plugin, *GitHub Repository*, GitHub.

Kunze, C. (2013), Nutzung semantischer Informationen aus OSM zur Beschreibung des Nichtwohnnutzungsanteils in Gebäudebeständen, Diplomarbeit, Technische Universität Dresden, Dresden.

Kunze, C., & Hecht, R. (2015), Computers, Environment and Urban Systems Semantic enrichment of building data with volunteered geographic information to improve mappings of dwelling units and population, *Computers, Environment and Urban Systems*, *53*, 4–18.

Poletti, F. (2016) Public Transit Mapping on Multi-Modal Networks in MATSim, Master thesis, Institute for Transport Planning and Systems, ETH Zurich, Zurich.

Ziemke, D., & Nagel, K. (2017), Development of a fully synthetic and open scenario for agent-based transport simulations – The MATSim Open Berlin Scenario, *VSP Working Paper* 17-12, TU Berlin, Transport Systems Planning and Transport Telematics, Berlin.

Zilske, M., Neumann, A. & Nagel, K. (2011), OpenStreetMap for traffic simulation, *In: Proceedings of the 1st European state of the map*, 126–134, Wien.