DATA61 | CSIRO

INTERNSHIP REPORT

Multi-Agent Simulation for Connected and Autonomous Vehicles

Author:
Mingyou MA

Supervisor:
Simona Adriana Mihăiță
Chen Cai

A report submitted in fulfillment of the requirements for the 2018/2019 CSIRO Undergraduate Vacation Scholarship Program

in the

Advanced Data Analytics in Transport

February 22, 2019

DATA61 | CSIRO

Abstract

Advanced Data Analytics in Transport

Internship Report

Multi-Agent Simulation for Connected and Autonomous Vehicles

by Mingyou MA

Addressing congestion in large cities is a major concern for traffic management centres, especially when new transportation modes emerge rapidly. Lately, connected and autonomous vehicles have gain a lot of popularity due to the promise of providing a new flexible transportation mode, which could be shared, real-time, electric and without the need of using any parking slots due to the continuous movement of cars. Future predictions foresee a considerable increase of the effective road capacity through intelligent and inter-connected vehicles. On the flip-side, more vehicles might be observed on the road if the autonomous vehicles get so comfortable, cheap and available that aggregated transit forms such as buses or trains become obsolete. But evaluating the impact of injecting a new fleet of autonomous vehicles in the existing road network is quite challenging as many unknown factors need to be taken into consideration, especially if they will run in a taxi-mode environment. Multi-agent simulation seems like an adapted tool to be able to evaluate the behaviour of such vehicles, the variation of traffic demand, the network response to incidents, etc. This report reviews the state-of-the-art multi-agent transport simulation frameworks along with presenting the methods, results and limitations of simulating a real-world urban dynamics using Victoria Road, Sydney as the case study.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

The history of self-driving cars can be traced back to the 1920s, with the promising trails took place in the 1950s. Experiments on autonomous vehicles (AVs) have been continuously conducted for a century. With the advancement of technology, modern vehicles are developed significantly, demonstrating different degrees of automation. Higher level of automations are still under development and testing, and will finally achieve fully automated vehicles. Potentially, AVs could benefit the society economically and environmentally by reducing crashes, cost of congestion, energy consumption and pollution. AVs could also change the vehicle ownership and alter the pattern of future land use (Panagiotopoulos and Dimitrakopoulos, 2018). Apart from that, they are able to transform the existing transport network when vehicle are provided with connectivity and interactions with pedestrians, other vehicles and even infrastructure.

Automated vehicles can be classified into 5 levels (scaled from 1 to 5) based on the degree of driving automation defined by SAE International (2016). Conventional cars are level 0 vehicles which do not exhibit any driving automation features, meaning that drivers have to perform the entire dynamic Driving Task (DDT) at all the times. While, level 1 automated vehicles have the capability to execute either the *longitudinal* **or** the *lateral* vehicle motion control subtask. Both *lateral* **and** *longitudinal* vehicle motion controls enable level 2 automated car to perform partial driving automation. From level 3 of automation, the Automated Driving System (ADS) performs the entire DDT when AS is engaged. With level 3 automations, vehicles monitor the Operation Design Domain (ODD) to determine whether ODD limits are about to exceed. If so, the level 3 vehicles will issue a timely request to intervene to the DDT fallback-ready user. Level 4 is built on level 3, with smoother vehicle-driver transitions to minimise the risk. The vehicle safety is significantly enhanced. In level 5, the fully driving automation will be finally established. Drivers are not required to perform the DDT or DDT fallback module. And drivers are fully considered as passenger when ADS is engaged. Currently, the most advanced commercialised models of AVs could only be considered as level 2.

Yet, the proliferation of AVs is still far from guaranteed. The high production cost of AVs hinder the large-scale implementation and mas consumer availability (Cars, 2012; Fagnant and Kockelman, 2015). Various questions associate with AVs, relating to legal, liability, privacy, licensing, security and regulations of insurance, making the implementation of AVs more challenging. Nevertheless, AVs are increasingly grabbing attentions from transport policymakers in various countries. And the AVs related research projects have become more favoured in the most recent years. Thus, investigating the impacts of AVs on the urban area is an critical in order to boost the realisation of future autonomy traffic.

1.2 Public's Perception of Connected and Autonomous Vehicles

Public's perception will play a crucial part of determining the role of the driverless cars in the future. The current studies using stated preference methods on autonomous vehicles (AVs) heavily focus on level 5 automation. Public acceptance of AVs is mainly governed by four aspects namely, *awareness*, *cost*, *safety* and *demographic differences*.

Starting from awareness, Schoettle and Sivak (2014) conducted a survey in the USA, Australia and the UK, and concluded that on average two thirds of people had hard of driverless cars. This number rose significantly , with 80% of 347 respondents in Austin, Texas, USA were aware of AVs in 2016 due to the continuous testing of self-driving cars in Austin (Bansal, Kockelman, and Singh, 2016). However, the percentage can be as conservative as 63%, which indicates that most of the public are aware of AVs but there is still a considerable proportion that are not. But this number is expected to be increased on a yearly basis due to more media coverage of this new mode of transport.

Cost is one of the important factors affecting travelling behaviour. A study carrying a series of analysis of online comments regarding the public's perception of driverless cars demonstrated that cost is a biggest concern (Fraedrich and Lenz, 2014). They also found that the majority of respondents were not prepared to pay any extra for this new transport mode. And the expected average additional cost brought by AVs were around US$2,000. The National automobile Dealers Association of the USA forecast that the average additional cost of AVs would be US$7,000-10,000. Currently, the cost of technology required for a semi-driverless car tremendously exceeds the average cost that people are prepared to pay. For instance, the additional cost for Tesla's enhanced autopilot with autonomy level 3 is US$49,000.

Safety concern is an inevitable component when a new mode of transport is introduced. A study performed by Jardim, Quartulli, and Casley (2013) requested respondents to rank safety, cost and legal issues that associate with the self-driving car. The result showed that 80% ranked safety as the primary concern. Dai and Howard (2013) mentioned that the public will become more interested in AVs if they could actively exhibit more safety benefits. A similar finding stated by Kyriakidis, Happee, and Winter (2015) also indicated that the safety was the number one concern and the most attractive quality of a self-driving vehicle.

Finally, distinct demographic groups displayed attitudes towards self-driving cars differently. On average, men thought AVs would be safer, while considering more about cost. On the other hand women thought AVs to be less safe and valued safety more than men. Interestingly, older women did not believe that the safety of AVs was promising. Additionally, younger generations are more interested in purchasing a private self-driving cars rather than using shared autonomous vehicles (Jardim, Quartulli, and Casley, 2013).

1.3 Private and Shared Autonomous Vehicles

The future AV-era will inevitably transform urban transport environment. Two major visions of adopting AVs as new modes of transport are private AVs and shared AVs (SAVs) with on-demand services. There are several SAVs modes which are able to be developed based on the existing mobility services, such as carsharing (e.g.

GoGet), ridesourcing (e.g. Uber), and ridesharing (e.g. UberPool), which can be classified as below (Nazari, Noruzoliaee, and Mohammadian, 2018):

- AV-rental - An autonomous carsharing system, whereby a traveller can rent an AV for a specific amount of time (e.g. on a daily or hourly basis) regardless of the number of trips made in that period.
- AV-taxi - An autonomous taxi system or autonomous ridesourcing, whereby a traveller can make a single trip from a specific location to another point, thereby making it a point-to-point. It is a trip-based service. AV-taxi will either have a backup driver or no driver. AV-taxi will be ideally operated as the on-demand service, with either *station-based* mode or *free-flow* mode.
- AV-access/egress - AVs could be a part of the multimodal public transport trips that consist of three stages: access, main part and egress. (Yap, Correia, and Arem, 2015)
- AV-carpool - AVs could serve for the carpooling services, which is more environmentally friendly and sustainable.

Generally, AVs could facilitate personal independence and mobility, thus further increasing the demand for automobile travel. In the autonomy traffic, however, the already-congested traffic patterns and roadway infrastructure would be negatively impacted due to the increased trip-making. SAVs could potentially become an optimal solution to enhance the mobility while alleviating traffic congestion. Fagnant and Kockelman (2015) demonstrated that each SAV could replace approximately 10 privately-owned vehicles. Therefore, share and automated mobility services would direct the future transport system towards positive directions.

1.4 Impacts of Autonomous Vehicles

It is viable to program AVs to not break traffic laws, with shortened reaction times and capability to smooth traffic flows. The improvement of fuel economy and the reduction of emissions are also promising features that AVs could potentially benefit the society.

Recently, researchers are developing ways for AV technology to reduce congestion and fuel consumption. For instance, AVs are capable to sense and possibly anticipate lead vehicles' braking and acceleration decisions. Such technology allows for smooth braking and find speed adjustments of following vehicles, leading to fuel savings, less brake wear, and reductions in traffic-destabilising shockwave propagation. Regarding intersection management, AVs could potentially create a new paradigms for signal control. Such advanced system could nearly eliminate the intersection delay. However, Dresner and Stone (2008) estimated that at least 95% AV-market penetration may be required to achieve such intelligent interesction deployment.

Apparently, numerous benefits would be difficult to realise until high AV shares are present. With 10% of all vehiles on a given freeway segment are AVs, there will likely be an AV in every lane at regular spacing during congested time, which could smooth traffic for all travellers. While AVs have a potential to increase roadway capacity with higher market penetration, the induced demand resulting from more automobile use might require additional capacity.

The safety and congestion-reducing impacts of AVs have potential to create significant changes in travel behaviour. In the future AVs traffic environment, it may bring higher automobile use, resulting in increased emissions, greater petrol consumption, etc. Nevertheless, some fuel-saving strategies could be developed as AVs are programmable. For example, AVs' smart parking decisions could help avoid cruising for parking. Liu (2018) modelled the joint equilibrium of departure time and parking location choices when commuter travel with autonomous vehicles, using bottleneck-constrained highway model to capture the essentials of the traffic dynamics and commuters' departure time choices. Liu proved that since walking time was replaced by additional AV self-driving time, there were substantial differences in commuting equilibrium proposed by Arnott, De Palma, and Lindsey (1991). With the consideration of social cost, travel cost for parking, number of parking space and the distance between parking spaces and city center, Liu also compared the total system cost for automated AV parking under two distinct flow patterns i.e. user equilibrium and system optimum time-dependent which accounts the toll and parking cost. The result reveals that if the social cost of unit parking supply does not vary too much over space, there should be more parking spaces in the city center to save self-driving of AVs; and the parking should be planned further away from city center if the social cost of parking in city center is much more expensive than that for locations further away. This concludes that the future autonomous traffic environment has the higher dependency on the existing infrastructures and facilities. Therefore, both urban transport environment and travellers' mobility pattern will be significantly impacted with the introduction of self-driving vehicles.

1.5 Simulation and Autonomy Traffic

As mentioned above, the current technologies like car-sharing and ride-sharing could be transformed into shared autonomous vehicles (SAVs) in the near future. The evolutionary future autonomous modes of transport would substantially alter travellers' mobility patterns. However, to examine such potential impacts brought by SAVs is challenging as currently due to various barriers to AVs implementation, AVs have not been officially tested in urban transportation system in a large scale, so most of the research works utilise simulation method to assess the developed models for AVs. For example, a simplified symmetric grid-based urban area with all-AVs traffic served as a platform for agent-based model operation in order to estimate environmental benefits brought by SAVs (Fagnant and Kockelman, 2015). A similar research work applying Central Area, Singapore as a case study used taxi trajectory data to model SAVs. Also, the multi-agent simulation was adopted to validate online and off-line vehicle-rebalancing models; and dynamic vehicle-passenger assignment (Lima Azevedo et al., 2016).

1.5.1 Conventional Simulation Models

Modelling and simulation of the real-world traffic is an effective approach to get more insights of traffic congestion; and to answer the problems of improving the traffic conditions. Conventionally, the two major traffic simulation models are static models and dynamic models. A static model represents an unchanged study area and a fixed time period without considering the fluctuation of demand of trips over the period. Also, the interactions between different time and distance steps are not taken into account. A typical static model is the four-step model. By contrast, a dynamic model is driven by the variability of transport demand over the study period. It is more suitable to describe the physical flow of road traffic. There are three approaches in dynamic modelling i.e. microscopic, mesoscopic and macroscopic models (Haman et al., 2017).

Even these simulation models are sophisticated and widely used in transport planning, some limitations still exist. Starting from the static models, although it has capability to model for large spatial scales, it utilises the travel demand that does not depend on time, which makes it impossible to capture the actual congestion. And to fulfil the aim of dynamic modelling for wide spatial scales, mesoscopic models are more applicable. However, it is less likely for mesoscopic models to replicate the local effects of traffic due to their coarse modelling properties. Microscopic models is able to maximise the detail of simulation of traffic dynamics, but it has higher computational cost and is less flexible.

1.5.2 Agent-based Simulation Model

Multi-agent system (MAS) is a tool which is well suited to the simulation of urban dynamics. It is more flexible than macroscopic models based on differential equations to simulate the scalable phenomena. MAS creates an artificial universe in which the experiments can be conducted by representing the individuals, their behaviours and their interactions. MAS is also regarded as a microscopic simulation. Four fundamental aspects of an MAS are: *Agent Behaviours*, *Environment*, *Scheduling* and *Interaction*. To explain in detail, (i) *Agent Behaviour* is intended to model the deliberative process agents. It is the approach to formulate agents' minds. (ii) *Environment* characterises the various physical objects in the simulated world. It is the process of shaping the situated environment, physical body of the agents and the endogenous dynamics of the environment. (iii) *Scheduling* handles the modelling of the passage of time and the definition of scheduling policies adopted to run the behaviours of agents. And finally, (iv) *Interaction* targets on outcomes of actions and interactions between agents at a given moment.

Comparing to conventional micro-, meso-, and macroscopic simulations, MAS offers more flexibility, with relatively lower computational cost. The flexibility of MAS can be defined from multiple dimensions. For instance, the complexity of agent can be easily modified with accordance to the actual urban environment. Changing the agent's behaviour, ability to learn and rules of interactions is considered as the primary dimension of flexibility. The secondary flexibility is the ability to adjust levels of description and aggregation, which means that the MAS can manipulate the group of agents, single agents or subgroups of agents, with the heterogeneous levels of description coexisting in a specific model.

Additionally, the emergent phenomena could be explicitly observed by MAS. Emergent phenomena result from the interactions of individual entities. Considering a traffic jam as an example, it results from the behaviour of and interactions between individual vehicle drivers. It is challenging to adopte the conventional modelling approaches to understand and predict such emergent phenomena. However, MAS has the nature of modelling the emergent phenomena by using canonical approaches (Bazghandi, 2012).

Chapter 2

State-of-the-art Multi-agent Simulation Framework

Multi-agent simulation is system and computational methods to study a collaborative and reactive transportation system by modelling autonomous decision-making by a collection of subsystem entities called *agents*. This computational method is developed to simulate the more complex adaptive system (CAS) or a distributed artificial intelligence (DAI) system. Compared to the traditional discrete event simulation, agent-based simulation defines the local behaviour rules of each entity from a bottom-up perspective. The main roots of agent-based simulation are in modelling human social and organisational behaviour and individual decision making. Hence, agent-based modelling tools can be used to test how changes in individual behaviours will affect the system's emerging overall behaviour.

Most of the agent-based frameworks are built by individual-based models. Note that individual-based models have roots in the activity-based travel demand models. Basically, all the agent-based frameworks exhibit a similar architecture, in which combines two transportation components – travel activities and network loading – into an integrated simulation platform.

Although dynamic traffic assignment (DTA) and agent-based simulations share some similarities (For example, both systems use simulation as a network loading method to measure travel time. And travellers' route choices are revised based on the measured travel time in previous iteration.), there are some significant dissimilarities between DTA approach and agent-based simulation which can be regarded as follows:

- Simulation-based DTA targets on changing travellers' route choice decisions only. Whereas, an agent-based model provides the feedback of travel time to a multi-dimensional decision domain, including not only travellers' route choice decisions but also a series of activity decisions, such as activity location, schedule and change of participation agenda.
- Since the decision-making for an agent is more complex than simulation-based DTA, an agent-based system usually adopts heuristic rules in feedbacks to achieve approximate convergence and consistency. Hence, less computational resources for agent-based simulation is required in order to achieve equilibrium between the network loading an assignment result.

The existing agent-based transportation systems follow a structural design as follows:

a) An agent represents an individual person or traveller and is associated with the individual demographic and travel characteristics.

b) The system generates demand, or an activity travel plan, for each individual agent based on demographic characteristics.

c) Activity plans are revised, enhanced and finalised such that all plans meet spatial (system environment) and temporal (schedule) constraints.

d) The activity plans are fed into a simulation module to produce the network-wide transportation results.

e) Network performance is a source of feedback to both activity plan and route choice decisions. Agents revise activity travel plans and route choice decisions, such that both decisions are optimised simultaneously.

This section aims to present a brief introduction of each well-known agent-based transportation simulation frameworks, including their features, drawbacks, models, framework structure, etc. Simulation platforms are illustrated in each distinguish sub-section.

2.1 TRANSIMS

TRANSIM is an open-source activity-based simulation platform. Travellers are modelled as agents and synthesised by census and survey data. The traffic simulation component in TRANSIMS is microscopic, consisting of car-following and lane-changing models.The framework structure of TRANSIMS is as follows:

1) Creation of population based on demographic data.

2) Generation of activities and activity locations.

3) Mode and routing choice simulation.

4) Transportation microsimulation based on choices.

5) Partial revision of individual activities and route choices based on the updated overall traffic performance.

Some drawbacks of TRANSIMS are:

- TRANSIMS cannot initially predict the fastest route as it does not provide the congestion information. Instead, an iterative relaxation approach is adopted by running the initial routes generation and replanning a fraction of the trips.
- TRANSIMS adopts a very detailed microsimulation, which leads a high computational cost.

Overall, TRANSIMS seeks to model traffic flow dynamic characteristics more, with larger demand of computation.

2.2 MATSim

MATSim is also an open-source activity-based simulation platform. The system is capable to run millions of agents simultaneously in the metropolitan area. It consists 2 layers i.e. physical layer and mental layer. Physical layer simulates the actual physical world where the agents move (traffic network). Whereas, mental layer generates strategies including route, mode choice and daily activity plans, and builds agents mind. The overall architecture of MATSim is considerably similar to TRANSIMS, except that the details of implementations.

The structure framework of MATSim is presented below:

1) Generation of a set of initial plans.

2) Each agent is assigned a created plan for execution.

3) Running the simulation to execute the plans.

4) Re-score the plans based on a produced new travel time for each trip.

5) A subset of the agents is chosen for plan adjustment or injection of new plans by external strategy modules.

6) Assigning those subset agents with a new or revised travel plans by running strategy modules.

7) Generation of updated route and mode choices for each agent.

8) Re-running the simulation until the stop criterion is satisfied.

Additionally, MATSim utilises spatial-queue model to measure traffic dynamics and queue spillovers rather than using point-queue model), with less time consumption. Nevertheless, the spatial-queue model may cause the unrealistic simulation of traffic dynamic. Especially for the case of congestion, not only losing appropriate flow dynamics propagated along the space and time dimensions but possibly resulting in an overestimation of travel time. But generally, MATSim is computationally fast.

2.3 OpenAMOS

An open-source activity-based simulation framework, OpenAMOS applies zonal socio-economic data and household travel survey data to synthesise agents. Agents' entire daily activity-travel pattern is segmented into various components as follows:

- Activity type choice models;
- Activity duration models;
- Activity location choice models;
- Mode choice and mode transition models;
- Initial departure timing models;
- Initial location models.

Basically, multinomial logit model which is used to formulate the activity-level patterns dominates the entire simulation process The structure of OpenAMOS is presented below:

1) Activity type choices are estimated by multinomial logit model.

2) These estimations finalise the blocked attributes.

3) The open attributes of an individual's activity-travel pattern are estimated by microsimulator in OpenAMOS.

4) OpenAMOS inputs the generated trips to a dynamic traffic assignment simulation platform called MALTA.

5) The dynamics in network traffic flow along the temporal dimension is replicated in MALTA module.

6) The performance producedby MALTA feeds back into OpenAMOS.

7) Traveller's activity-travel pattern decisions are updated based on the MALTA performance feedbacks.

In conclusion, OpenAMOS focuses more on individual-based model with the extensive application of socio-economic and household travel survey data.

2.4 SACSIM

SACSIM is a regional travel forecasting model, adopting multinomial logit and nested logit models for daily activity generations. It is an integrated activity-based disaggregate econometric model. Agents are synthesised by households drawn from the region's U.S. Census Public Use Microdata Sample.

A detailed description of framework structure is as follows:

1) Generation of a 1-day activity and travel schedule for each person in the population (including a list of agents' tours and trips on each tour).

2) The created activities are loaded to the network in order to determine the congestion and network performance.

3) Then the network performance is re-measured to model and update a person's activity and travel patterns.

4) The updated travel patterns are injected to the network again.

5) The model achieves equilibrium and stops when the network performance used as input matches the network performance which is obtained from the simulation result.

2.5 ILUTE

ILUTE is a comprehensive urban transportation agent-based simulation platform. The key feature of this system is its ability to capture interactions between urban land use, travel demand, the transportation system, even the environmental impacts. Also it is capable to address high-resolution policy analysis in a variety of transportation and urban planning contexts. ILUTE is developed based on the urban land use. Land use and transportation are fundamentally connected, in which the land use pattern directly determines travel needs, activity agenda type and participation, and viability of alternative travel modes.

ILUTE is a disaggregated, behavioural approach to simulate activities of agents. Various entities such as persons, families, houses, buildings, firms could be the agents in ILUTE. It is worthwhile to mention that ILUTE aims to simulate the emergent land-use / transportation interactions in the long term. In this study, ILUTE is less relevant. Thus, it will not be explained in detail.

2.6 SimAGENT

SimAGENT is a system that features PopGen, CEMSELTS (Comprehensive Econometric Microsimulator for Socioeconomics), CEMDAP (Comprehensive Econometric Microsimulator for Daily Activity travel Patterns) and traffic assignment. The detailed explanation of structure is as follows:

1) PopGen generates a synthetic population of individuals and households.

2) CEMSELTS simulates and generates long-term demographic attributes for persons and households.

3) CEMDAP generates the daily activity travel plan for each individual. It simulates activity-travel patterns of all individuals in the region for a 24-hour period. There are two separate steps in activity plan generation: (1) generation of mandatory activity and corresponding schedules; (2) generation of full daily activity agenda and schedule.

4) Traffic-assignment module determines a traveller's route choice decision and measures the performance of the overall network. The time-dependent trip interchange matrices of O-D among the traffic analysis zones which are generated from CEMDAP are served as fed into the traffic assignment component to determine routes and the overall network performance.

2.7 SimMobility

SimMobility is a high-level architecture simulation platform, which is composed of three main modules differentiated by the time frame (short-term, mid-term and long-term), in which the behaviour of an urban system is modelled. It is also an agent-based simulator. Every agent exists and is recognised by all three levels; and information is used according to each level's needs. The long-term time frame is similar to ILUTE, which predicts the evolution of land use, determining the life-cycle decisions of agents. As mentioned above, it beyond the scope of this study. Therefore, the long-term framework is not considered.

Regarding the mid-term time frame, it handles transport demand and supply at the day level. It simulates daily activities and travel at the individual level by using the behavioural models, which exhibits the similar architecture as OpenAMOS and MATSim because the mid-term is also an activity-based simulator. Activity sequence, trip origin / destination, and departure times are predicted based on the econometric activity schedule approach. The framework structure is illustrated as follows:

1) The preday model generates the activity plans and trip decisions for each agent based on the availability of modes, utility specifications, and activity types (either mandatory or non-mandatory activities).

2) The output from preday model will be served as an input in the within-day model. The within-day model makes the predictions of routes for planned trips, transforming activity plans and trip decisions into actual trips.

3) Depending on the traffic conditions and effective travel times, the agents could reschedule the rest of the day, cancel an activity, reroute while travelling.

4) Then the supply simulator follows the dynamic traffic assignment. It runs for private and public transport modes. Unlike other simulation platforms, this framework also accurately estimates the effect of bus operations on the road traffic.

5) The updated network performance measures were then transferred back to preday model as a learning mechanism for individual choices re-estimation.

6) Though an iterative process, consistency can be achieved between the demand and supply models of the mid-term simulator.

Note that, there is a connection between mid-term and short-term simulators. The within-day component provides trip chains to the short-term simulator, uses as input demand to simulate specific network regions in mode detail. (Figure 2.1)

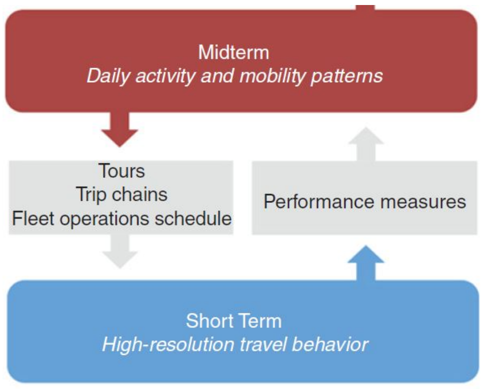

Midterm
Daily activity and mobility patterns
Tours
Trip chains
Fleet operations schedule
Performance measures
Short Term
High-resolution travel behavior

FIGURE 2.1: Connection between short-term and mid-term simulators in SimMobility Lima Azevedo et al., 2016

SimMobility short-term framework contributes to simulations of drivers' route choice decisions and driving behaviour parameters, and randomly-assigned vehicle characteristics. This framework is developed based on the open-source traffic simulation application MITSIM. It allows to precisely capture drivers' responses to neighbouring conditions by implementing the acceleration model. Also, the lane-changing model integrates mandatory and discretionary lane changes in a single structure. Additionally, merging behaviour through courtesy and yielding as well as drivers' response to traffic signals, information, speed limits and incidents. Other simulation platforms mentioned in the previous subsections are less advanced than SimMobility short-term simulator, with the coarser modelling on flow dynamics characteristics (MATSim).

In Lima Azevedo et al., 2016, the authors integrate the SimMobility short-term simulator with a detachable autonomous mobility on-demand (AMOD) controller. AMOD controller consists of a fleet management which controls the vehicle-rebalancing, and a vehicle-tracking component. To fully apply the scenario where autonomous vehicles are penetrated in the traffic, the authors adjust and modify the acceleration and lane-changing models in the vehicle flow model of SimMobility short-term simulator, achieving the behaviour homogenisation of autonomous vehicles. For example. the safety margins in regard to gap acceptance, safety headway, and reaction time were reduced to 1.0, 1.0 and 0.5s respectively.

2.8 Agentpolis

Agentpolis is another open-source large-scale multi-agent simulation platform. Agents are created in the simulation environment that consists road network composed nodes connected by road segments (powered by OpenStreetMap). A case study in Fiedler, Čáp, and Čertickỳ, 2017 defines on-demand vehicles and passengers as agents. The authors aim to utilise AgentPolis to investigate the impact of large-scale mobility-on-demand system deployment on total traffic intensity and formation of congestion. Unlike other introduced multi-agent simulation platforms, Agentpolis does not include activity plans generator. Hence, the agents' trip decisions are not re-scheduled. Although this simulation platform exhibit easy-to-use features, comparing with other simulations platform, it is limited for the purpose of this study.

In this project, MATSim is adopted. More details of MATSim will be introduced in the following chapter.

Chapter 3

MATSim Framework

3.1 MATSim Introduction

MATSim[1] is an activity-based, extend-able, multi-agent simulation framework written in Java. The framework is designed for large-scale scenarios, meaning that all models' feature are stripped down to efficiently handle the targeted functionality. Some previous studies in investigating autonomous vehicles used MATSim as a simulation platform. For the network loading simulation, a queue-based model is implemented. Comparing with SimMobility Short-Tem, it omits complex and computationally expensive car-following behaviour. At the current stage, MATSim is designed to model a single day.

Agents in MATSim are travellers who generate daily activities. The ultimate goal of agents is repeatedly optimising its daily activity schedule while in competition for spatial-temporal slots with all other agents on the transportation infrastructure. The mechanism is similar to the route assignment iterative cycle, but goes beyond route assignment by incorporating other choice dimensions like time choice, mode choice, or destination choice into the iterative loop. Figure 3.1 demonstrates the simulation loop and corresponding functionality in MATSim. In every iteration, prior to the simulation of the network loading with the MATSim *mobsim* i.e. mobility simulation, each agent selects a plan from its memory. This selection is dependent on the plan scores, which are computed after each *mobsim* run, based on the executed plans' performances. On the next iteration, a certain share of the agents (typically 10%) are allowed to clone the selected plan and modify this clone i.e. rescheduling. At the end of each iteration, the actual performance of the plan in the synthetic environment is taken to compute each executed plan's score. The iterative process is repeated until the average population score stabilises.

3.1.1 Traffic Flow Model in MATSim

The large-scale modelling with lower computational cost is the key feature in MATSim. It utilises the computationally efficient queue-based approach to model the traffic dynamic (Figure 3.2). A vehicle is a added to the tail of the waiting queue when it enters a new road segment (i.e link) from an intersection. As it is the queue-based model, it remains on that link until 1) the time for travelling on the link with free flow has passed; and 2) until the vehicle has become the head of the waiting queue; and 3) until the next link has an additional slot for the vehicle to enter. However, the car-following effects are unable to be captured as this model over simplifies the actual traffic dynamic.

[1]This entire chapter cites the book *The Multi-Agent Transport Simulation MATSim* (Horni, Nagel, and Axhausen, 2016)

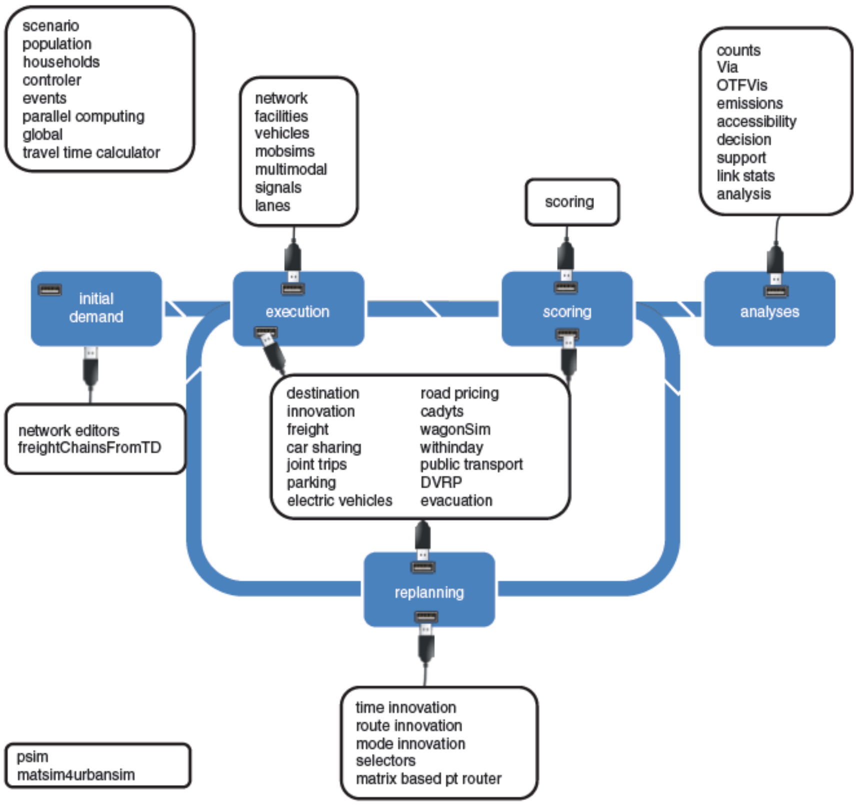


FIGURE 3.1: MATSim Functionality

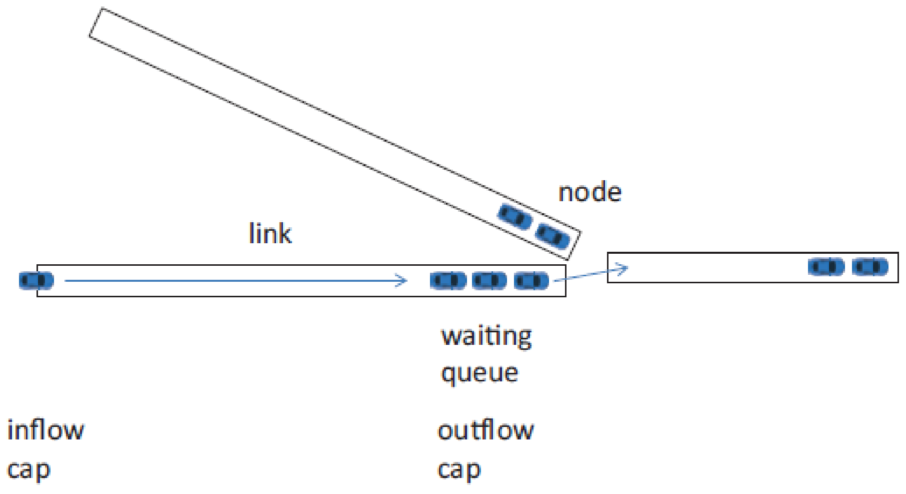


FIGURE 3.2: MATSim's Traffic Flow Model

3.1.2 Co-Evolutionary Algorithm and Agent's Plan

To achieve the equilibrium condition, the co-evolutionary algorithm is adopted as illustrated in Figure 3.3. Different species subject are able to interact with each other through operating such co-evolutionary algorithm. In MATSim, individuals are represented by their daily activity plans, where a person represents a species. The advantage of using the co-evolutionary algorithm is that the optimisation can be performed with respect to agents' daily activity plans and travel plans. Apart from the achievement of standard traffic flow equilibria, which ignores travellers' activities, in the co-evolutionary algorithm, the agents cannot further improve their plan unilaterally.

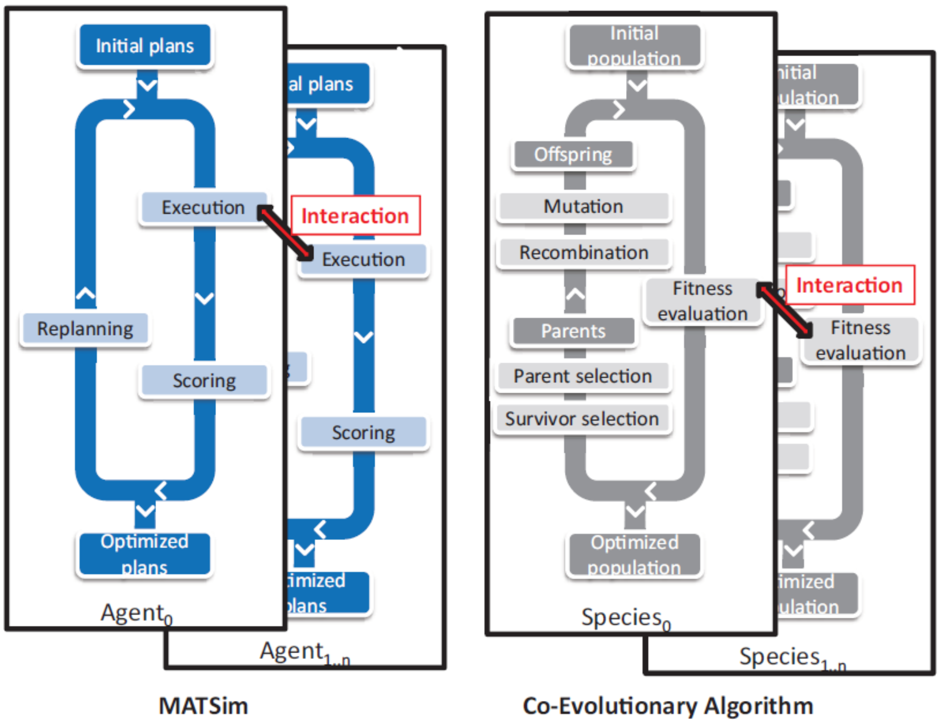


FIGURE 3.3: The co-evolutionary algorithm in MATSim

3.2 Scoring and Plan Evaluation

As detailed in Section 3.1.2 and by Figures 3.3, MATSim operates based on a co-evolutionary algorithm, in which each agent learns by manipulating multiple plan. And all these plans are scored by executing and evaluating them in *mobsim*. And to run for the next iteration, those plans with higher scores will be chosen or modified. Here are the key elements in the iterative process:

- **mobsim** - The mobility simulation dumps a selected plan per each agent and executes it in the simulated environment.
- **scoring** - The performance of the plan in the simulated environment in each iteration is taken to calculate executed plan's score.
- **replanning** - There are three steps in the replanning module:

1) If an agent has more plans than the maximum number of plans that is defined in the MATSim configuration by the user (for more details about the MATSim configuration, please refer to the Appendix), the exceed plans will be removed according to the plan sector.
2) Some agents' plans may be subjected to duplication, modification, and finally selection for the next iteration. This is defined as choice extension or innovation.
3) All other agents select between their existing plans.

To explain these three elements in detail, an agent's generated plans in each iteration could be considered as the agent's **choice bank** with respect to that iteration. Basically, step 1 and 2 re-plan or modify the **choice bank**, while step 3 implements the selected plan. Normally, the plan with higher score will be more likely to be chosen.

Obviously, the MATSim agent-based model is scoring-oriented. Only those plans with relatively high scores will be assigned to the agent and survive the plans removal step. Therefore, the scoring function has to be precise and be able to capture the heterogeneity of travellers' preferences. For instance, some may favour congested traffic over smooth one, others may prefer a crowded but more budget trips by public transport, while others could choose cycling or walking to reach his or her destination regardless the weather condition.

It is noteworthy that MATSim is modelled based on complete day plans, which means that the partial daily-modelling is not straightforward. However, modelling the whole-day activity using utility function is limited due the absence in the literature, MATSim adopts Charypar-Nagel scoring instead. In the following subsections, the Charypar-Nagel Modelling Approach will be introduced.

3.3 Charypar-Nagel Utility Function

Charypar and Nagel (2005) formulated the score function based on road congestion model proposed by Vickrey (1969). In MATSim, apart from the time-of-travel modelling, which is the original interest of Charypar-Nagel model, this model is being utilised for evaluating other choice dimensions. In this section, the mathematical expressions of Charypar-Nagel utility function are detailed and explained.

Basic Function - The basic utility score S_{plan} consists two components i.e. $S_{act,q}$ and $S_{trav,mode(q)}$, representing activity utilities and travel utilities, respectively. The sum of the $S_{act,q}$ plus the sum of $S_{trav,mode(q)}$ gives the score of the plan. which can be expressed as:

$$S_{plan} = \sum_{q=0}^{N-1} S_{act,q} + \sum_{q=0}^{N-1} S_{trav,mode(q)} \tag{3.1}$$

where N is the number of activities in the whole-day plan. Trip q is the subsequent trip following the activity q. And the last activity is combined with the first activity to balance the number of trips and activities.

Activities - The utility of an activity q is computed by:

$$S_{act,q} = S_{dur,q} + S_{wait,q} + S_{late.ar,q} + S_{early.dp,q} + S_{short.dur,q} \tag{3.2}$$

where the individual components are defined as follows:

- $S_{dur,q}$
$$S_{dur,q} = \beta_{dur} \cdot t_{typ,q} \cdot \ln(t_{dur,q}/t_{0,q}) \tag{3.3}$$
is the utility function of the activity q, where opening the opening hours of the locations of activities are included. t_{dur} is the performed activity duration, β_{dur} is related to the marginal utility of activity duration and $t_{0,q}$ is the duration when utility starts to be positive.

- $S_{wait,q}$
$$S_{wait,q} = \beta_{wait} \cdot t_{wait,q} \tag{3.4}$$
represent the waiting time spent. β_{wait} is the direct marginal utility of time spent waiting; and $t_{wait,q}$ is the waiting time.

- $late.ar, q$
$$later.ar, q = \begin{cases} \beta_{late.ar} \cdot (t_{start,q} - t_{latest.ar,q}) & \text{if } t_{start,q} > t_{latest.ar,q} \\ 0 & \text{else} \end{cases} \tag{3.5}$$
quantifies the late arrival penalty, where $t_{start,q}$ is the activity starting time q and $t_{latest.ar}$ is the latest possible penalty-free activity starting time.

- $S_{early.dp}$
$$S_{early}.dp = \begin{cases} \beta_{early.dp} \cdot (t_{end,q} - t_{earliest.dp,q}) & \text{if } t_{end,q} > t_{earliest.dp,q} \\ 0 & \text{else} \end{cases} \tag{3.6}$$
defines the penalty for not staying long enough, where $t_{end,q}$ is the activity ending time and $t_{earliest.dp.,q}$ is the earliest possible activity end time q. Normally, the value of $\beta_{early.dp}$ is set to be zero, with the expectation of good-performance data about this effect.

- $S_{short.dur,q}$
$$S_{short.dur,q} = \begin{cases} \beta_{short.dur} \cdot (t_{short.dur,q} - t_{dur,q}) & \text{if } t_{dur,q} < t_{short.dur,q} \\ 0 & \text{else} \end{cases} \tag{3.7}$$
is the penalty for 'too short' activity, where $t_{short.dur}$ is the shortest possible activity duration. Similarly, $\beta_{short.dp}$ is recommended to be zero.

Figure 3.4 illustrates the sample of activity starting and ending time in the MATSim `configuration.xml` file.

Travel - Travel disutility for a leg q is defined as:

$$S_{trav,q} = C_{mode(q)} + \beta_{trav,mode(q)} \cdot t_{trav,q} + \beta_m \cdot \Delta m_q + (\beta_{d,mode(q)} + \beta_m \cdot \gamma_{d,mode(q)}) \cdot d_{trav,q} + \beta_{transfer} \cdot x_{transfer,q} \tag{3.8}$$

where:

- $C_{mode(q)}$ is a mode-specific constant.

- $\beta trav, mode(q)$ is the direct marginal utility of the time spent travelling by mode.

- $t_{trav,q}$ is the travel time between activity locations q and $q + 1$.

```
<module name="planCalcScore">
    <param name="learningRate" value="1.0" />
    <param name="BrainExpBeta" value="2.0" />

    <param name="lateArrival" value="-18" />
    <param name="earlyDeparture" value="-0" />
    <param name="performing" value="+6" />
    <param name="traveling" value="-6" />
    <param name="waiting" value="-0" />

    <param name="activityType_0"            value="h" /> <!-- home -->
    <param name="activityPriority_0"        value="1" />
    <param name="activityTypicalDuration_0" value="12:00:00" />
    <param name="activityMinimalDuration_0" value="08:00:00" />

    <param name="activityType_1"            value="w" /> <!-- work -->
    <param name="activityPriority_1"        value="1" />
    <param name="activityTypicalDuration_1" value="08:00:00" />
    <param name="activityMinimalDuration_1" value="06:00:00" />
    <param name="activityOpeningTime_1"     value="07:00:00" />
    <param name="activityLatestStartTime_1" value="09:00:00" />
    <param name="activityEarliestEndTime_1" value="" />
    <param name="activityClosingTime_1"     value="18:00:00" />
</module>
```

FIGURE 3.4: Timing in the MATSim `configuration.xml` file

- β_m is the marginal utility of money (normally positive).
- Δm_q is the change in monetary budget caused by fares, or tolls for the complete leg (normally negative or zero).
- $\beta_{d,mode(q)}$ is the marginal utility of distance (normally negative or zero).
- $\gamma_{d,mode(q)}$ is the mode-specific monetary distance rate (normally negative or zero).
- $d_{trav,q}$ is the distance travelled between activity locations q and $q+1$.
- $\beta_{transfer}$ are public transport transfer penalties (normally) negative.
- $x_{transfer,q}$ is a $0/1$ variable signaling whether a transfer occurred between the previous and current leg.

Generally, MATSim is executed based on the co-evolutionary algorithm which reviews, updates, discards and selects the generated plans from *mobsim*. The utility functions are the key for evaluating the performance of each plan and running the co-evolutionary algorithm program. In the next chapter, a MATSim case study will be detailed.

Chapter 4

Case Study: Victoria Road, Sydney

To use agent-based model to investigate the autonomous vehicles, firstly, the basic multi-agent framework for simulating the conventional traffic environment has to be conducted. This chapter introduces the original datasets adopted in this project for the traffic simulation purpose.

4.1 Background of Victoria Road

Victoria Road is a major transport corridor in Sydney, New South Wales, Australia, connecting Parramatta with the western end of Anzac Bridge. It is currently one of the longest roads in Sydney. The total length is 21 km, with three lanes in each direction between Rozelle and Gladeville. The road was named as one of the most congested road in Sydney with an average travel speed of 24 kilometres per hour during the morning peak hour and 31 kilometres per hour in the afternoon peak (Haynes, 2013), which makes this road a good case study for traffic analysis and modelling.

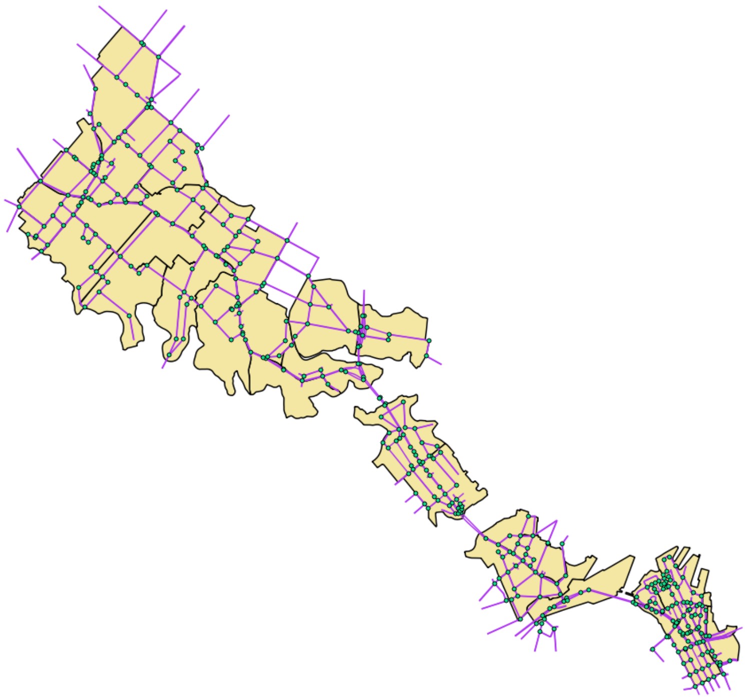


FIGURE 4.1: Victoria Road, Sydney

4.2 Data Description

If multi-agent simulation is a well-designed skyscraper, then data is columns and beams. Data is the critical component in the agent-based model in order to replicate the real world urban dynamics. Original datasets utilised in this project are detailed in this section.

4.2.1 Traffic Network Data

Traffic network data describes the configuration of the road, including capacity, free flow speed, number of lanes, etc. There are two components in a traffic network i.e. nodes and links. A link (or edge) of a network is the connections elements between the nodes (or vertices) of the network. In the traffic network data of Victoria Road, all the links have a direction, meaning they point from one node to the next. The attributes of the traffic network data[1] are as below:

- **node coordinate** - The coordinate is an essential data representing the basic geographical features. In MATSim the standard coordinate system is **EPSG:32608 – WGS84**.
- **link starting node and ending node** - Each link has a pair of nodes with corresponding IDs (represented by numerical values). The link starting node (referred as from-node) and link ending node (referred as to-node) indicate the link direction.
- **capacity** - The capacity of a link indicates the maximum number of vehicles could be fitted on the road under congested traffic condition.
- **speed** - The free flow speed of the link (in km/h).
- **number of lanes** - The number of lanes on each traffic links.

4.2.2 Origin-Destination Matrices

An origin-destination matrix (O-D matrix) describes travellers' movement in a certain urban area. O-D matrix is an important data in the second step of classical urban transportation planning system model — *trip distribution*. Such matrices indicates the number of trips generated / attracted by each region. The O-D matrices can be processed and used to synthesise the virtual population, which will be explained in the following chapter. O-D matrices are results of the microscopic simulation performed by traffic modelling software — **AimsunNext**. It is noteworthy that the trips are generated and attracted by 'centroids' in **AimsunNext**. Additionally, the number of trips are not necessarily integer.

In this project, 9 O-D matrices are provided with different time range, describing the number of trips generated from one centroid and attracted by another from 06.45 am to 09.00 am.

4.2.3 Centroids

Centroids are the artificially-defined points in the traffic network by the **Aimsun Next**. Vehicles are 'pumped' or 'absorbed' by centroids in the AimsunNext. A centroid or a couple of centroids are being allocated in each traffic analysis zone

[1]Note that the node data and link data are in two separated `.GeoJSON` files

(TAZ), meaning that in a TAZ unit, all the generated trips are being aggregated and represented by centroids. Each centroid has a coordinate, which is an essential information to understand the travellers' travel pattern.

The datasets mentioned in this chapter have to be processed to comply with the requirement of MATSim input file. In the next chapter — **Methodology**, the methods and assumptions associated with data processing are presented.

Chapter 5

Methodology

The input datasets required by different simulation framework could vary significantly. For instance, in SimMobility, the simulation-related datasets are programmed and managed by SQL, precisely postgreSQL. While in MATSim, the essential data of traffic network and agents' plan are stored in the `.xml` file with its unique format. This section details the assumptions, methods and algorithms used to synthesise the virtual urban traffic environment for the Victoria Road, Sydney along with the creation of population.

5.1 Network Clearance

The network in MATSim is node-to-node based, which means that the links have to be terminated by two nodes. In the original datasets, some links do not have the information about the ending nodes. This causes those links to be open-ended. Typical locations of 'open-ended' links are the boundary of the network and roundabouts (Figure 5.1).

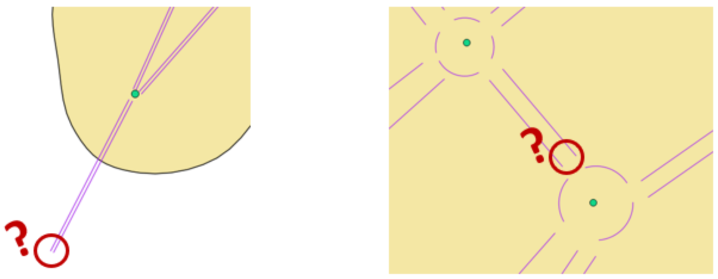


FIGURE 5.1: Open-ended links

Luckily, the QGIS-extracted `link.geoJSON` file exhibits additional details of all the links on Victoria Road. Thus, the link terminating coordinates are yielded, consequently filling the missing data. This process can be regarded as the 'creation of fake nodes', as the yielded coordinates do not actual exist on the original datasets. To replicate a complete network of Victoria Road, artificially-added nodes have to be taken into account.

Overall, there are 602 nodes and 1199 links, with 212 'fake nodes' which accounts for 35%. In addition, the fundamental features of links such as free flow speed, number of lanes, etc. are also included, forming the final MATSim input file —

`network.xml`[1]. The final product is visualised by **Via**, the MATSim visualiser (Figure 5.2).

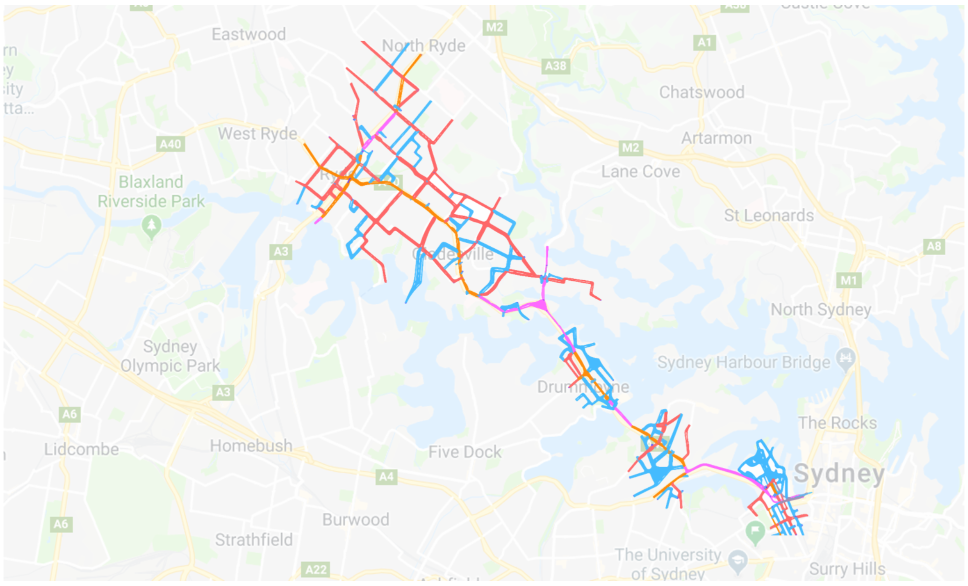


FIGURE 5.2: Visualisation of Victoria Road by Via, the MATSim visualiser

5.2 Population Creation

Environment, agents and interactions are the three main components in the multi-agent simulation. In the previous section, the traffic environment formulation has been introduced. To replicate the real world travelling behaviours, agents' daily activities have to be calibrated by the data reflecting the actual urban dynamics. In this section, the method of data manipulation and processing are introduced.

5.2.1 Creation of Residence and Workplace

Agents can be regarded as travellers in MATSim. Each of them has a specific plans of daily activity. To generate the population, some basic properties have to be created such as residences , schools or workplaces. In this case study, due to data limitation (only 9 O-D matrices were provided), only the traffic in the morning peak hours was simulated. Therefore, all the simulated agents are considered as morning commuters, meaning that only residences and workplaces (or schools) are the possible locations where agents could perform their activities.

Hence, the residences and workplaces are written by processing the original network data. Due to lack of the facility data describing the actual locations of the buildings on the Victoria Road, some artificially-generated facilities (residences and workplaces) are placed, surrounding the nodes (Figure 5.3).

5.2.2 Trip Distribution

Once the location of facilities has been defined, the trips generated and attracted by each of them have to be quantified. In this project, each simulated trip extracted

[1]**For the details of network.xml; please refer to the Appendix.**

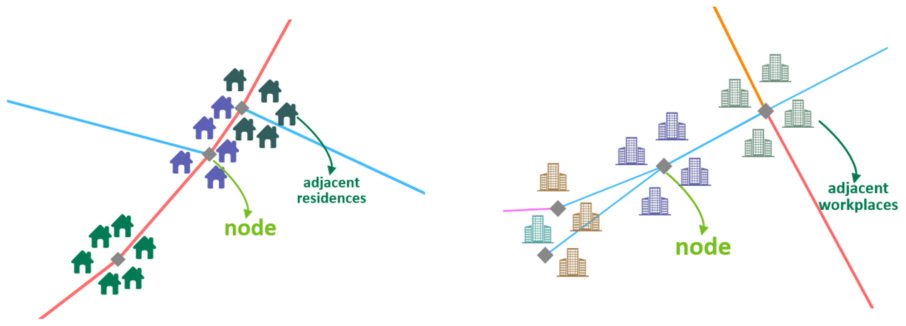


FIGURE 5.3: Creation of facilities

from the AimsunNext software is regarded as the trip generated by a single agent. This means that the total number of trips in the O-D matrices represents the size of the agent population in MATSim agent-based model.

Unlike the AimsunNext microscopic simulation, in MATSim, all the trips are node-to-node-based, rather than centroid-to-centroid-based. Hence, the trips have to be redistributed from centroids to nodes. The procedures of redistributing trips are as below:

1. **Allocation of nodes to the nearest centroids** - The first step of converting centroid-to-centroid-based trips to node-to-node-based trips is assigning the nodes to the nearest centroids where the vehicles are pumped or absorbed. In this project, the allocation of nodes are based on the one traffic analysis zone (TAZ) unit. All the nodes that stay in the same TAZ as the centroids do will be regarded as the centroid-adjacent nodes. Note that the nodes on the boundary of the TAZ will be also considered as the centroid-adjacent nodes with respect to that TAZ. (Figure 5.5)

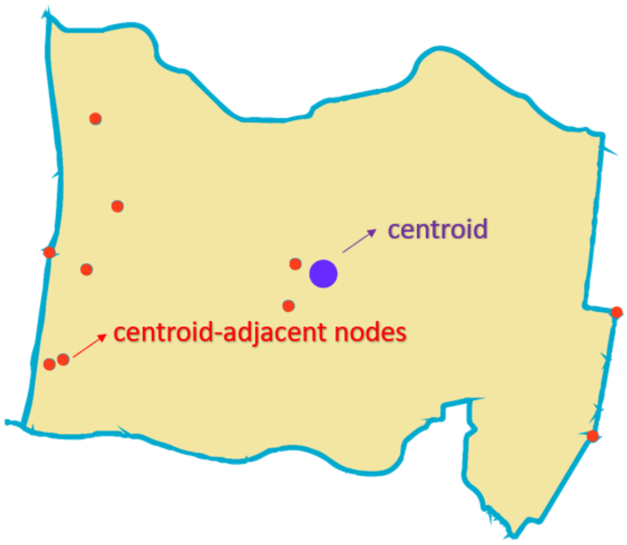


FIGURE 5.4: Centroid-node allocation

2. **Trip redistribution** - Trips generated or attracted by centroids are aggregated in each TAZ unit. It is challenging and almost impossible to know the attractiveness of each centroid-adjacent nodes. Hence, in this project, all the trips generated or attracted by centroids are evenly distributed to the centroid-adjacent nodes, which can be expressed by:

$$t_{\alpha_{|C_{J_i}}} = \frac{T_{|C_J}}{n_{|C_J}} \tag{5.1}$$

where

- $T_{|C_J}$ is the total number of trips generated or attracted by centroid C_J;
- $n_{|C_J}$ is the total number of centroid-adjacent nodes with respect to centroid C_J;
- $t_{\alpha_{|C_{J_i}}}$ is the total number of trip that are received by each centroid-adjacent nodes $\alpha_{|C_{J_i}}$ with respect to centroid C_J.

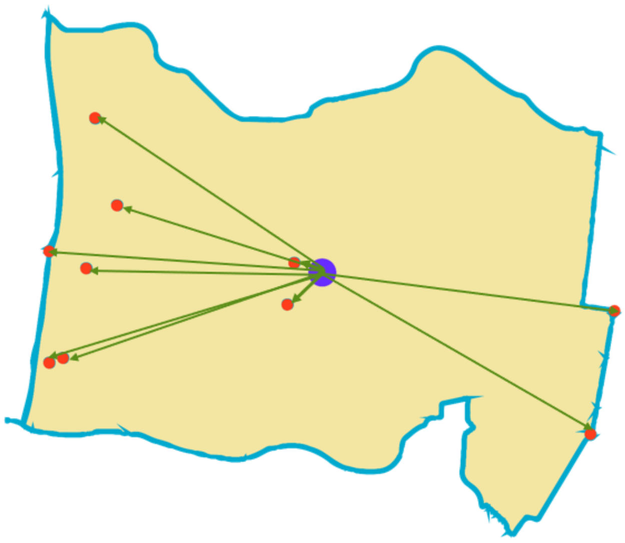


FIGURE 5.5: Centroid-node allocation

3. **Centroid-based to node-based trips conversion** - To define a proper trip, its origin and destination have to be cleared. In the original O-D matrices, the trips start from one centroid and end at the other. But in MATSim, the edges of a trip are nodes. The mathematical expression of the conversion of centroid-based to node-based trips can be regarded as:

$$t_{(\alpha_{|C_{I_i}}, \alpha_{|C_{J_j}})} = \frac{T_{|(C_I, C_J)}}{n_{|C_I} \cdot n_{|C_J}} \tag{5.2}$$

where

- $t_{(\alpha_{|C_{I_i}}, \alpha_{|C_{J_j}})}$ is the total number of trips generated by the node $\alpha_{|C_{I_i}}$, which belongs to the origin centroid C_I; and attracted by node $\alpha_{|C_{J_j}}$, which belongs to the destination centroid C_J. This can be regarded as the node-to-node-based trips;

- $T_{|(C_I,C_J)}$ is the total number of aggregated centroid-to-centroid-based trips.

5.2.3 Route Assignment

Route assignment or route choice concerns the selection of paths between origins and destinations in a transportation network. To build an agent's mind, initially, the paths have to be determined and be written in the agents' fundamental plan which is served as the MATSim input i.e. `AgentPlan.xml`. However, there are thousands of solutions of the routing problem. In this project, to simplify this step, all the agents are assumed to be 'greedy', meaning that they would only choose the path with the lowest cost. This assumption leads the application of **Dijkstra's shortest path algorithm** which is used to find the shortest paths between nodes in a graph. Here, the cost of travelling on the links is the link travel time, which is derived by using link length divided by free flow speed.

The Dijkstra's algorithm is applied and formulated in Python as shown below:

```
def dijsktra(graph, initial, end):
    # shortest paths is a dict of nodes
    # whose value is a tuple of (previous node, weight)
    shortest_paths = {initial: (None, 0)}
    current_node = initial
    visited = set()

    while current_node != end:
        visited.add(current_node)
        destinations = graph.edges[current_node]
        weight_to_current_node = shortest_paths[current_node][1]

        for next_node in destinations:
            weight = graph.weights[(current_node, next_node)] + weight_to_current_node
            if next_node not in shortest_paths:
                shortest_paths[next_node] = (current_node, weight)
            else:
                current_shortest_weight = shortest_paths[next_node][1]
                if current_shortest_weight > weight:
                    shortest_paths[next_node] = (current_node, weight)

        next_destinations = {node: shortest_paths[node] for node in shortest_paths if node not in visited}
        if not next_destinations:
            return "Route Not Possible"
        # next node is the destination with the lowest weight
        current_node = min(next_destinations, key=lambda k: next_destinations[k][1])

    # Work back through destinations in shortest path
    path = []
    while current_node is not None:
        path.append(current_node)
        next_node = shortest_paths[current_node][0]
        current_node = next_node
    # Reverse path
    path = path[::-1]
    return path
```

5.2.4 Time of Travel

An agent-based simulation in the application of transport modelling can be regarded as the activity-based model. Time is the one last dimension required in MATSim. Without the definition of time, agents will not perform the daily activity. Thus, time of travel has to be allocated to each agent. However, in reality, the actual departure time is extremely difficult to capture. In this project, the departure time for each agent is randomly assigned to an agent based on the time range corresponding to O-D matrices. For instance, for the O-D matrix describing the number of trips from 06.45 am to 07.00 am, an agent can be depart at any time in between 06.45 am to 07.00 am such as 06.50 am.

With the creation of agents' residences, workplaces, origins, destinations, departure time and routes, total 103, 700 agents were formulated. The associated agent plans was written in a single `AgentPlan.xml`. A sample agent's plan is shown below:

```
<person id="12">
   <plan>
      <act type="h" x="333284.09" y="6250641.41" link="33910" end_time="08:03" />
      <leg mode="car">
         <route>105087 117570 1082342 117573</route>
      </leg>
      <act type="w" x="333362.42" y="6250353.08" link="1082341" end_time="12:30" />
   </plan>
</person>
```

For the generation of full agents' plan, please refer to the Appendix.

Chapter 6

Result

As mentioned in the previous section, in the agent-based model, there are three essential components, namely environment, agents and interactions between agents and environment. MATSim requires the initial loading of the environment and population, then conducting the simulation and returning the outputs describing the variability of agents' revised plans in each iteration. Also, the interactions are able to be visualised by the visualiser. In this chapter, the simulation results[1] are detailed along with the result visualisation[2].

6.1 Interactions

Interactions are the third fundamental element in an agent-based simulation model. Agents have the capability to interact with other. Also, they are able to response, react and re-schedule (3Rs) based on the environmental changes. In MATSim, the interactions are not explicit. The most direct way is to use the MATSim extensions such as **Simunto Via**, **senozon**, etc. to visualise the result and to yield more insights of interactions.

6.1.1 Variation of Travel Speed

Figure 6.1 demonstrates the visualisation of the morning peak hour (06.45 am to 09.00 am) traffic on the Victoria Road, Sydney, using the output from MATSim simulation. This visualisation was achieved by **Simunto Via**, the MATSim visualiser.

It is obvious to see that each agent's[3] colour varies substantially. This is due to the colour settings in the visualiser. Each colour has a different meaning. For instance, the agents with green colour indicate that their travel speeds in the environment were the same speed as the allowable link travel speed i.e. the free flow speed. While, the orange arrows means that those agents were travelling with a quarter of the maximum link travel speed.

The variation of colours directly reflects the changes in agents' travel speed. It is a typical example of the agent-environment interaction, as agents are able to adjust their travel behaviours with accordance to the overall traffic condition. The mechanism behind such feature is the queuing model introduced in Section 3.1.1 with the aid of Figure 3.2.

[1]The results section only illustrates the findings of interactions — the third key element in the agent-based model

[2]Please visit `https://youtu.be/QTOMq_3Tvg4` to watch the video of output visualisation.

[3]An agent is represented by an arrow in the visualiser.

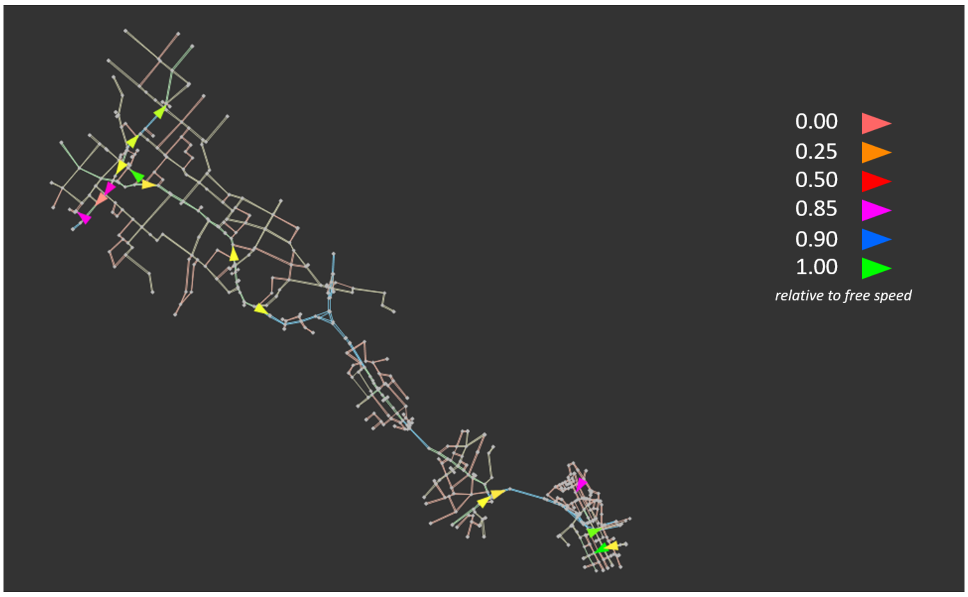


FIGURE 6.1: Variations of agents' travel speed

6.1.2 Re-routing

As discussed in Section 3.2, the co-evolutionary algorithm assists the replanning module to upgrade or re-write agents' plan, and selecting the plans with higher scores as the input for the next iteration. Re-routing is one of the replanning approaches. Figure 6.2 presents the difference between route assignment at the initial iteration and iteration 10. It is clear to observe that the agents made minor alternations in route choices. It is noteworthy that the revised paths are not necessarily to be physically shorter or with lower cost of travel. Instead, those updated routes always have higher scores with respect to the previous iterations (Figure 6.3).

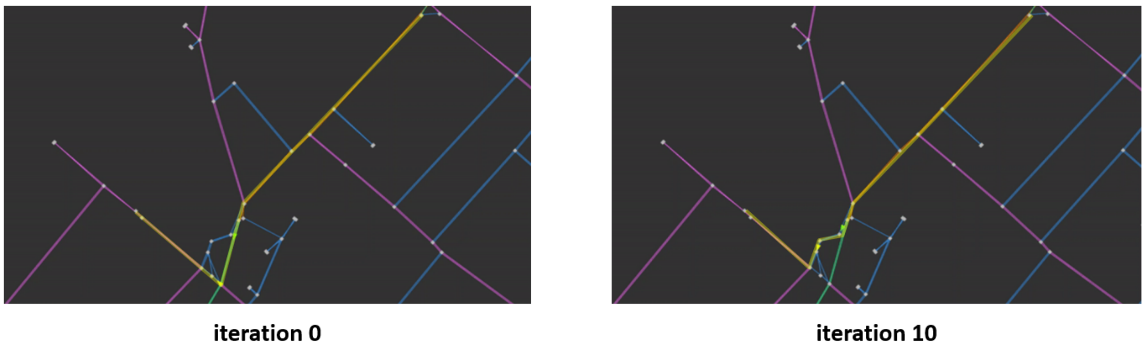


FIGURE 6.2: Agents' Re-routing

6.1.3 Re-timing

Time of travel is the other degree of freedom which could be adjusted in order to elevate the score of a plan. Figure 6.4 and 6.5 compares the different departure time selected by agents at the initial iteration and iteration 10. It can be found that at iteration 10, agents have 'learnt' to leave their 'residences' (origins) either earlier or later (i.e. before 06.45 am and after 09.00 am).

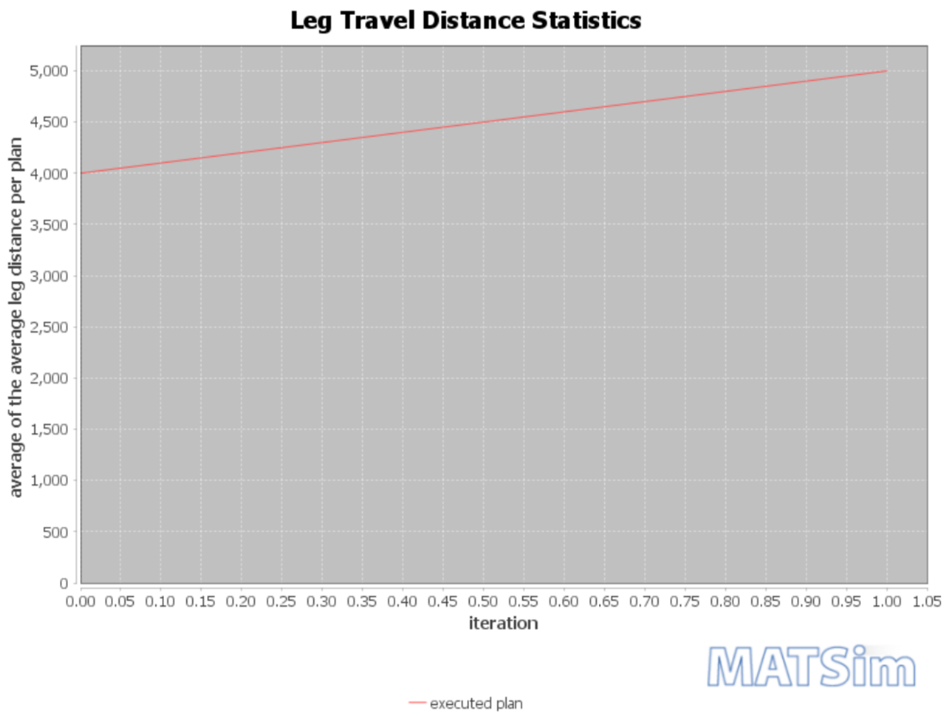


FIGURE 6.3: An escalation of total travel distances from iteration 0 to 1.

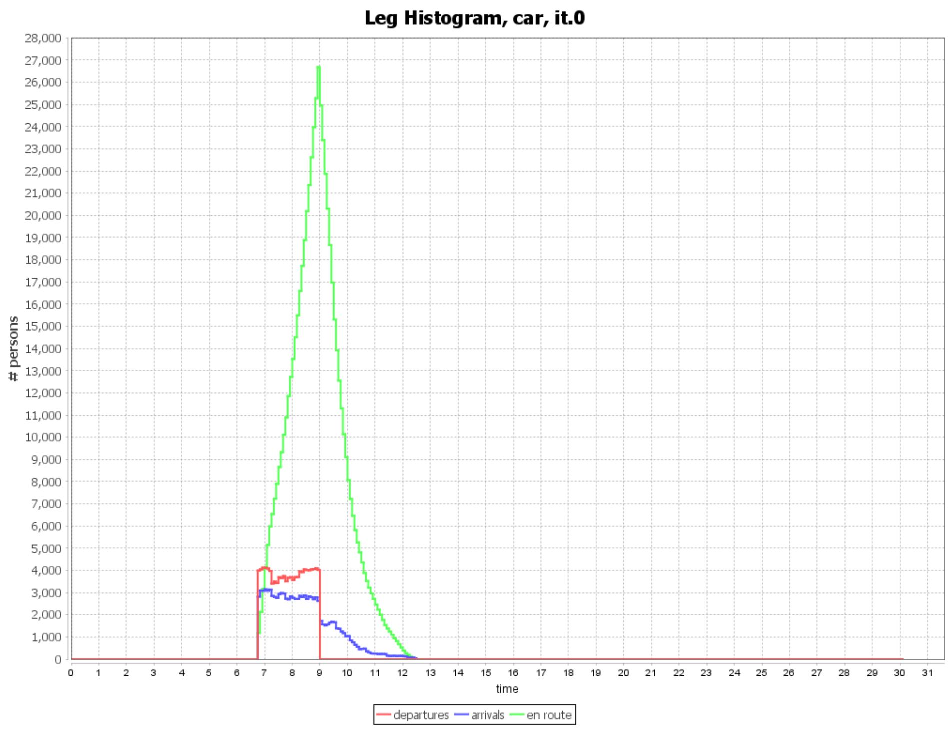


FIGURE 6.4: Time-related statistics at iteration 0.

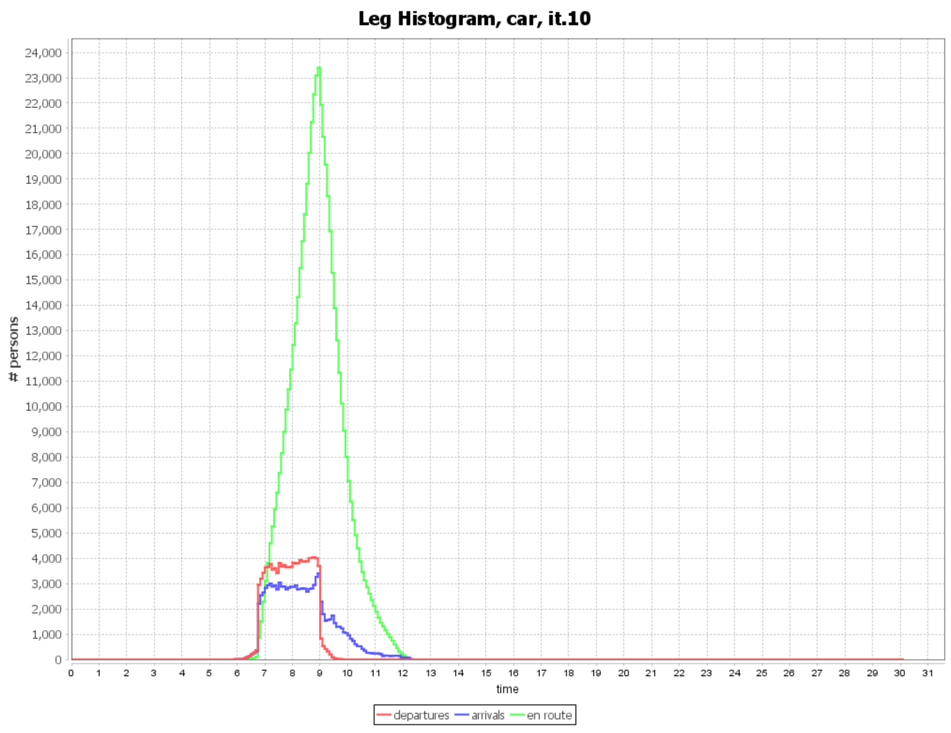


FIGURE 6.5: Time-related statistics at iteration 10.

6.2 Score Statistics

Figure 6.6 shows the changes of plans' score from iteration 0 to 10. Clearly, in this case, the equilibrium condition achieved at iteration 5 as the scores were converged at about 52.

6.3 MATSim Computational Cost Statistics

Figure 6.7 presents the MATSim's computational cost to perform the morning traffic simulation on the Victoria Road. Iteration 0 has the longest runtime, which indicates that MATSim takes more time to process the initial agents' plans. Once the convergence achieved, the runtime dropped significantly and varied slightly (iteration 5 – 9).

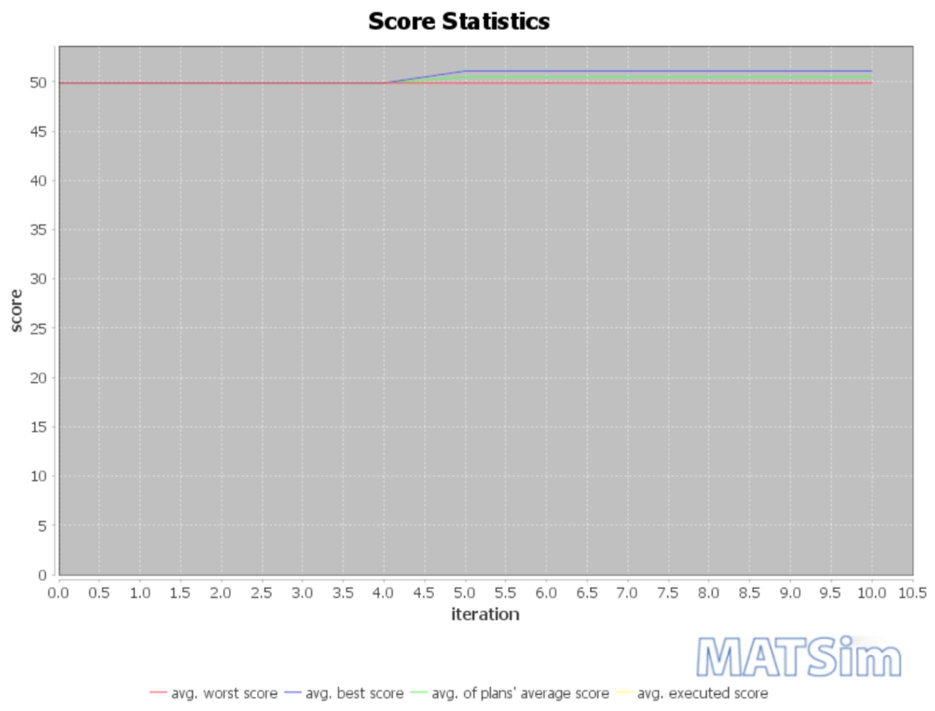


FIGURE 6.6: Score statistics for simulating Victoria Road, Sydney

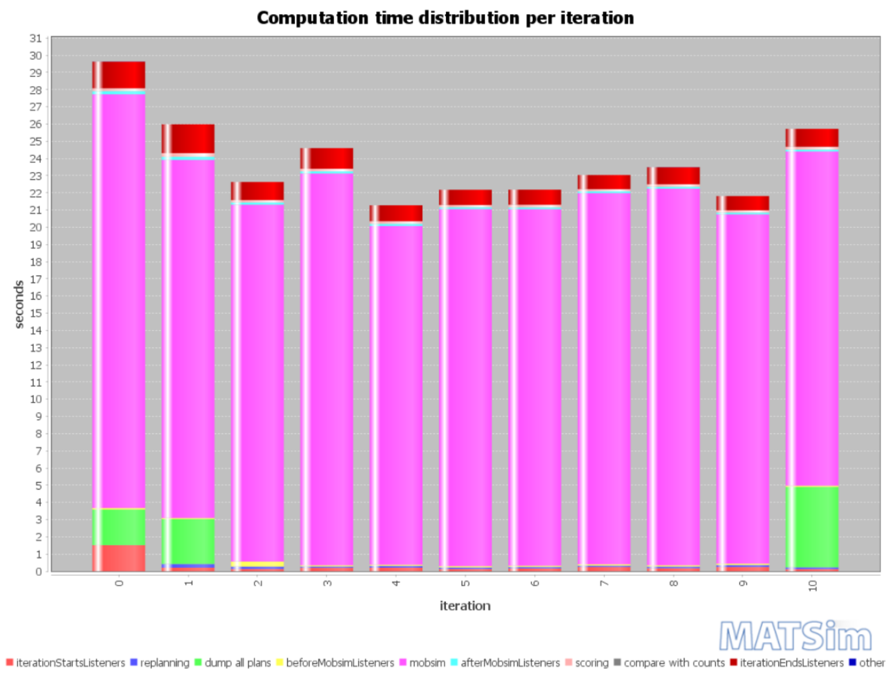


FIGURE 6.7: Computational cost statistics

Chapter 7

Limitation and Future Study

7.1 Data Limitation and Future Data Support

This project aims to initiate the multi-agent transport simulation, which lays the foundation for future agent-based modelling and simulating for connected and autonomous vehicles. The mono-mode urban transport environment (i.e. private vehicles only) is adopted for the case study. To fully address the challenges of urban dynamics, the next step would be building both agents and traffic environment with higher complexity, which means that agents could have additional features in their daily activity such as secondary trips; more facilities could be added in; and multiple means of transport like taxi, buses, trains, etc. could be involved in the existing model.

The datasets utilised in this study are basic and limited as:

1. The O-D matrices only capture the travel patterns of private vehicles, leading the mono-mode traffic simulation;
2. The socio-demographic data does not give any information about the work start time, which impacts the plan-replanning module. Consequently, the agents travel behaviours are failed to be simulated with accordance to the real-world scenario;
3. There is no data for the facilities such as schools, shops, workplaces and residence, on the Victoria Road. The artificially-created node-surrounding facilities further degraded the simulated traffic environment.
4. Centroid-to-centroid-based trips are at the aggregate level. The manual trip redistribution could end up with trip over- or under-estimation.

To enhance quality of the replication of agents and traffic environment for performing the multi-agent transport simulation, the suggested raw datasets are listed as below:

- General Transit Feed Specifications (GTFS) that describes schedules and associated geographic information of the public transportation services operating on the Victoria Road, Sydney;
- Taxi trajectory data for autonomous vehicles modelling;
- Land use data detailing the locations and characteristics of residential buildings, firms and schools on Victoria Road for the creation of the authentic urban environment;
- Smart card data presenting the actual usage of the public transport for building up the high-level agents' plans.

7.2 Holonic Multi-agent System

Simulation is an appropriate approach to establish abstractions of the system in order to study the complex systems which are inaccessible through direct observation and measurement. Simulating large number of entities requires great computing resources. A solution to tackle this problem is to apply macroscopic models. Nevertheless, macroscopic models are unreliable as they are failed to allow the observation at the individual levels. However, the Holonic Multi-Agent System (HMAS) seems like a suitable tool to deal with this high complex traffic system. Structure of a Holonic Multi-Agent System is built upon the Multi-Agent System.

A Holonic Multi-Agent System provides the promising approach for constructing the complex domains which is characterised by a hierarchical structure. A holon is defined as simultaneously a whole and a part of a whole. A holon can be composed by other holons, with the fulfilment of three conditions: being stable, having a capacity for autonomy and being able to cooperate. The hierarchical structure composed of holons is called a holarchy (Figure 7.1). Tchappi et al. (2018) proposed that the multilevel mechanism is subjected to be developed for the entire simulation (for agent behaviours and environment) while remaining generic. They intend to extend the four perspectives of an MAS to integrate the multilevel aspects. The ultimate goal is to make an agent's behaviours and interactions dynamically adapt the external execution constraints.

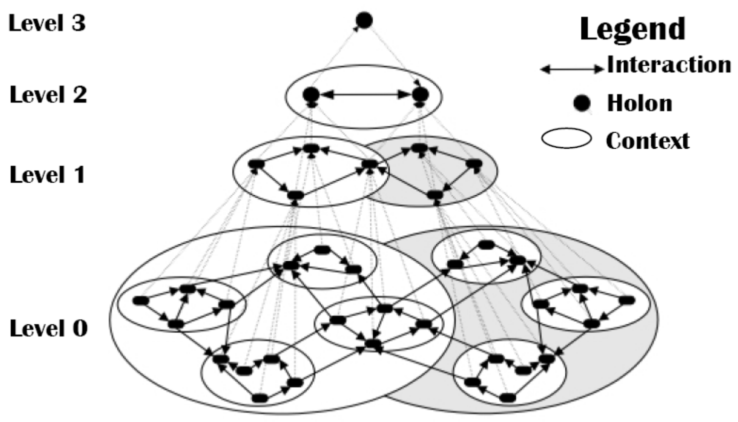


FIGURE 7.1: Structure of Holonic Organisational Multilevel Model

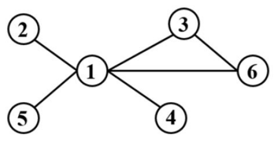
To introduce the holonification method, the previous research work presented by Esmaeili, Mozayani, and Motlagh (2014) is reviewed.

Esmaeili, Mozayani, and Motlagh mentioned that in a holonic multi-agent system, the interaction only happens in between holons that are either from their relative upper, lower or same level. Note that in such systems, any individual holon is regarded as semi-autonomous entities, in which the their actions and behaviours are dominated by their own local knowledge under the control of their super-holon.

With the recent advancement in multi-agent systems, agents have become more organisation-oriented in which each agent interacts only with its neighbours. Hence, regardless of the complexity of the system, only some local information about the entire network is sufficient for agents. Finally, a more efficient system can be achieved.

A multi-agent network can be represented by $MAN =< A, I >$, where A is the set of agents of MAS and I refers to the set of interaction relationship between the agents. An adjacency matrix Λ, in which the rows and the columns are indexes of agents and the items in Λ are defined as:

$$\Lambda(a_i, a_j) = \begin{cases} 1 & \textit{if } (a_i, a_j) \in I \\ 0 & \textit{otherwise} \end{cases} \quad ; \; a_i, a_j \in A \tag{7.1}$$



$A = \{1,2,3,4,5,6\}$

$I = \{(1,2),(1,3),(1,4),(1,5),(1,6),(3,6)\}$

$$\mathbf{\Lambda} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

And the holonification problem in each level of a holarchy is finding a set, H, of subsets h_i, such that:

$$\{h_1, h_2, ..., h_k\} \,|\, h_i \subset A; \; \bigcup_i h_i = A \tag{7.2}$$

where, k is the total number of holons an h_i are the holons themselves. Esmaeili, Mozayani, and Motlagh assumed that that in **each level** of a holarchy, the holons are disjoint, which is different from the general definition of general holonic multi-agent system. Thus,

$$\forall h_i, h_j \in H; l(h_i) = l(h_j) \longrightarrow h_i \cap h_j = \emptyset \tag{7.3}$$

Well-defined equations i.e. Eq.(7.1) – (7.3), allows to implement of holonification process. Esmaeili, Mozayani, and Motlagh adopted the **bottom-up** approach to establish the holarchy. This process is defined as *Holonification*. The very first step of holonification is locating the agents which are eligible to play the role of the head. A basic holon is composed by these head agents at the lowest level in the holarchy. Then, the other agents of the network attempt to join these holons according to their corresponding position in the network and the eligibility of their neighbours. This process is repeated in order to construct the upper levels of holarchy until a single holon, which can be regarded as a root, is generated, containing all the other holons in the holarchy.

The concept of centrality in network theory allows to select the important agents as the heads in the network. Esmaeili, Mozayani, and Motlagh used eigenvector centrality to define the eligibility of the agent a_i to become a head as follows:

$$elig(a_i) = \lambda_{\max}^{-1} \sum_j \Lambda(a_i, a_j) \cdot elig(a_j) \tag{7.4}$$

where, the λ_i are the eigenvalues of the adjacency matrix Λ, and λ_{max} is the largest of them.

Consequently, by comparing eligibility values of each agent, the head within each level of holarchy can be determined. That is:

$$Head = \left\{ a_i \in A | elig(a_i) > \frac{\sum_j elig(a_j)}{|A|} \right\} \tag{7.5}$$

Then a basic holon for each the agents that are selected as head is created. Each head is the only members of the corresponding created holon. And for each of the non-head agents, they are required to find the neighbour with higher eligibility value:

$$BN(a_i) = \underset{a_j \in Neighbour(a_i)}{\arg\max} (elig(a_j)) \tag{7.6}$$

where, $BN(a_i)$ is the function to find the most eligible neighbour of agent a_i, and $Neighbour(a_i)$ is the set of all neighbours of a_i in the network.

$$Holon(a_i) = \begin{cases} h_{a_j} & a_j \in Head \ (a_i, a_j) \\ Holon(a_j) & otherwise \end{cases} \quad ; a_j = BN(a_i) \tag{7.7}$$

where, $Holon(a_i)$ specifies the holon that a_i should join, and h_{a_j} is the holon that a_j is a head in.

The algorithm is repeated from Eq. (7.4) – (7.7) until there is only a single holon created in the current level i.e. the root holon. Finally, the holonic and organisational multi-agent network is formed.

TABLE 7.1: Traffic network configuration

Property	Value
Number of intersections	25
Number of links	31
Length of each link	436.72m
Maximum speed	60km/h
Number of lanes per link	2
Arrival distribution	uniform
Simulation duration	5 hours
Traffic demand	30,000 veh/hour

To evaluate the proposed "bottom-up" method is evaluated by implementing the algorithm into the problem of controlling traffic signals. Here, intersections are considered as agents, with the roads connecting these intersections which are regarded as interaction edges (Figure 7.2).

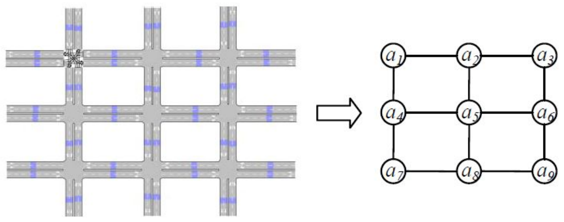


FIGURE 7.2: Multi-agent network model of a urban traffic network

The tested urban traffic network consists 25 intersections i.e. agents and 31 connecting roads (interaction edges). And framework utilised to conduct the experiment is **AIMSUN**. Additional parameters are illustrated in Table 7.1.

The result of the holonic multi-agent simulation indicates that the holarchy structure is effective, especially for the large scaled and complex system.

To integrate the concept of such **bottom-up** holonification method with the traffic simulation for Victoria Road, it could be applied to the classification of agents that are regarded as travellers. Agents with similar travel patterns or daily activity plans, potentially, are able to be clustered together to lower the computational cost, boosting the efficiency of analysis.

Chapter 8

Conclusion

This report introduces the how the multi-agent simulation can be applied to the transport modelling. With the detailed comparison of the state-of-the-art agent-based simulation frameworks, MATSim is adopted in this project as MATSim requires fundamental transport-related dataset to perform the simulation. Also, MATSim is developed based on the activity-based model, having the co-evolutionary scored-based modules to comprehensively review, edit or even discard the agents' behaviours. It is noteworthy that MATSim has lower computational cost comparing with other multi-agent platforms such as SimMobility, due to its queuing-based model. However, this model is not capable to replicate the traffic dynamics. Vehicles behaviours in a microscopic level cannot be evaluated.

To understand how the agent-based model works, in this project, a real-world case study — Victoria Road, Sydney is applied. Multiple datasets such as dynamic origin-destination matrices, network shape files, centroid configurations extracted by AimSun are utilised, processed and integrated in order to build two of three essential elements i.e. agents and environment in the multi-agent simulation. Due to data limitation, agents are assigned the privately-owned vehicles as the only mode of morning commuting (06.45 am to 09.00 am) transport. Artificially-created facilities like residences and workplaces which are served as trip origins and destinations, respectively, are synthesised with the network. Additionally, all the centroid-to-centroid-based trips that are simulated by AimSun have been converted to node-to-node-based trips in order to comply with MATSim's requirement. Finally, with the aid of Dijkstra's shortest path algorithm and randomly timing generation algorithm, 103,700 agents have been created and simulated in MATSim.

The simulation results reveal the key characteristics of the agent-based model. Firstly, agents are able to respond to the external environment. For instance, agents' travel speeds reduce significantly when the traffic volume rises. Secondly, agents learn how to re-route without changing their trip origins and destinations. Furthermore, earlier or later departure is the other approach of optimising the travel plans, with the enhancement of scores of agents' plans, until the convergence has achieved.

To investigate autonomous vehicles (AVs) using multi-agent simulation as a tool, future data supports have to be readily available. Typical required datasets are taxi trajectory data, GTFS data, smart card data and land use data. Apart from that, the holonic multi-agent system (HMAS) could be developed based on the current agent-based model to enhance the efficiency of simulation. For on-demand autonomous vehicles, HMAS is potentially capable to cluster the AVs with accordance to their stations or regions.

Bibliography

1. Arnott, Richard, Andre De Palma, and Robin Lindsey (1991). “A temporal and spa- tial equilibrium analysis of commuter parking”. In: *Journal of public economics* 45.3, pp. 301–335.
2. Bansal, Prateek, Kara M Kockelman, and Amit Singh (2016). “Assessing public opin- ions of and interest in new vehicle technologies: An Austin perspective”. In: *Transportation Research Part C: Emerging Technologies* 67, pp. 1–14.
3. Bazghandi, Ali (2012). “Techniques, advantages and problems of agent based mod- eling for traffic simulation”. In: *International Journal of Computer Science Issues (IJCSI)* 9.1, p. 115.
4. Cars, Self-Driving (2012). “The next revolution”. In: *Ann Arbor, MI., Richard Wallace and Gary Silberg, KPMG and CAR*.
5. Charypar, David and Kai Nagel (2005). “Generating complete all-day activity plans with genetic algorithms”. In: *Transportation* 32.4, pp. 369–397.
6. Dai, D and D Howard (2013). “Public Perceptions of Self-driving Cars: The Case of Berkeley, California”. In: *Prepared for the 93rd Annual Meeting of the Transportation Research Board, Revised Submission Date: November*. Vol. 15, p. 2013.
7. Dresner, Kurt and Peter Stone (2008). “A multiagent approach to autonomous inter- section management”. In: *Journal of artificial intelligence research* 31, pp. 591–656.
8. Esmaeili, Ahmad, Nasser Mozayani, and Mohammad Reza Jahed Motlagh (2014). “Multi-level holonification of multi-agent networks”. In: *Intelligent systems (ICIS), 2014 iranian conference on*. IEEE, pp. 1–5.
9. Fagnant, Daniel J and Kara Kockelman (2015). “Preparing a nation for autonomous
10. vehicles: opportunities, barriers and policy recommendations”. In: *Transportation Research Part A: Policy and Practice* 77, pp. 167–181.
11. Fiedler, David, Michal Čáp, and Michal Čertický (2017). “Impact of mobility-on-
12. demand on traffic congestion: Simulation-based study”. In: *Intelligent Transporta- tion Systems (ITSC), 2017 IEEE 20th International Conference on*. IEEE, pp. 1–6.
13. Fraedrich, Eva and Barbara Lenz (2014). “AUTOMATED DRIVING– INDIVIDUAL AND SOCIETAL ASPECTS 1 ENTERING THE DEBATE 2”. In:
14. Haman, Igor Tchappi et al. (2017). “Towards an multilevel agent-based model for traffic simulation”. In: *Procedia Computer Science* 109, pp. 887–892.
15. Haynes, Rhys (2013). *Doing 24km/hr? You must be on Victoria Rd*. https://www.dailytelegraph.com.au/news/nsw/doing-24kmhr-you-must-be-on-victoria-rd/news-story/3c8514f9a09ad870f26b8890e393f5f1 .
16. Horni, Andreas, Kai Nagel, and Kay W Axhausen (2016). *The multi-agent transport simulation MATSim*. Ubiquity Press London.
17. Jardim, Adam Sebastian, Alex Michael Quartulli, and Sean Vincent Casley (2013). “A Study of Public Acceptance of Autonomous Cars”. In:
18. Kyriakidis, Miltos, Riender Happee, and Joost CF de Winter (2015). “Public opin- ion on automated driving: Results of an international questionnaire among 5000 respondents”. In: *Transportation research part F: traffic psychology and behaviour* 32, pp. 127–140.

19. Lima Azevedo, Carlos et al. (2016). *Microsimulation of Demand and Supply of Autonomous Mobility On Demand*. Tech. rep.

20. Liu, Wei (2018). “An equilibrium analysis of commuter parking in the era of autonomous vehicles”. In: *Transportation Research Part C: Emerging Technologies* 92, pp. 191–207.

21. Nazari, Fatemeh, Mohamadhossein Noruzoliaee, and Abolfazl Kouros Mohammadian (2018). “Shared versus private mobility: Modeling public interest in autonomous vehicles accounting for latent attitudes”. In: *Transportation Research Part C: Emerging Technologies* 97, pp. 456–477.

22. Panagiotopoulos, Ilias and George Dimitrakopoulos (2018). “An empirical investigation on consumers’ intentions towards autonomous driving”. In: *Transportation Research Part C: Emerging Technologies* 95, pp. 773–784. SAE International (2016). *Taxonomy and definitions for terms related to driving automa- tion systems for on-road motor vehicles*.

23. Schoettle, Brandon and Michael Sivak (2014). “A survey of public opinion about autonomous and self-driving vehicles in the US, the UK, and Australia”. In: Tchappi, Igor Haman et al. (2018). “A brief review of holonic multi-agent models for traffic and transportation systems”. In: Procedia computer science 134, pp. 137–144. Vickrey, William S (1969). “Congestion theory and transport investment”. In: The American Economic Review 59.2, pp. 251–260.

24. Yap, MD, Goncalo Correia, and Bart van Arem (2015). “Valuation of travel attributes for using automated vehicles as egress transport of multimodal train trips”. In: Transportation Research Procedia 10, pp. 462–471.

25. Mihaita A. S., Dupont L., Cherry O., Camargo M., Cai C., Evaluating air quality inside an eco-neighbourhood by combining stationary and smart mobile pollution, Journal of Cleaner Production (JCLP), Vol. 221, pp 398-418, ISSN = 0959-6526, https://doi.org/10.1016/j.jclepro.2019.02.179, (IF = 7.24, H5 = 154, place 42/100 top rated venues in Google scholar).

26. Mihaita A. S., Benavides, M., Camargo, C. Cai, Predicting air quality by integrating a mesoscopic traffic simulation model and air pollutant estimation models, International Journal of Intelligent Transportation System Research (IJITSR), DOI: 10.1007/s13177-018-0160-z, Online ISSN1868-8659, Online scheduled for 15 May 2019, Volume 17, Issue 2, pp 125–141.

27. Mihaita A. S., Dupont L., Camargo M., Multi-objective traffic signal optimization using 3D mesoscopic simulation and evolutionary algorithms, Simulation Modelling Practice and Theory (SIMPAT), https://doi.org/10.1016/j.simpat.2018.05.005, Volume 86, August 2018, Pages 120-138, (IF = 2.063, H5=49).

28. Wen T, Mihăiţă A-S, Nguyen H, Cai C, Chen F. Integrated Incident Decision-Support using Traffic Simulation and Data-Driven Models. Transportation Research Record. 2018;2672(42):247-256. doi:10.1177/0361198118782270, (IF = 0.695, H5 = 48)

29. Mihaita A.S., Mocanu S., Lhoste, P., "Probabilistic analysis of a class of continuous-time stochastic switching systems with event-driven control", European Journal of Automation (JESA), July 2016.

30. Monticolo, D., Mihaita, A.S., Darwich, H., Hilaire, V., "An Agent Based System to build project memories during engineering projects", Knowledge Based Systems Journal (KBS), January 2014

31. Monticolo, D. Mihaita A.S. "A multi Agent System to Manage Ideas during Collaborative Creativity Workshops", International Journal of Future Computer and Communication (IJFCC), vol 3., nr 1, February 2014, P66-71, (extended version of the paper presented in ICFCC 2013).

32. Mihaita A.S., Mocanu S., "Simulation en temps continu pour la commande orientée

événements des systèmes stochastiques à commutation", European Journal of Automation (JESA), 45 1-3 (157-172), Oct 2011.

33. Mihaita A. S., Dupont L., Cherry O., Camargo M., Cai C., Air quality monitoring using stationary versus mobile sensing units: a case study from Lorraine, France, 25th ITS World Congress (ITSWC 2018), Copenhagen, Denmark, 17-21st of September 2018.
34. Wen Tao, Mihaita A.S., Nguyen Hoang, Cai Chen, Integrated Incident decision support using traffic simulation and data-driven models. Transportation Research Board 97th Annual Meeting (TRB 2018), Washington D.C., January 7-11, 2018.
35. Mihaita A. S., Tyler Paul, Wall John, Vecovsky Vanessa, Cai Chen, Positioning and collision alert investigation for DSRC-equipped light vehicles through a case study in CITI, 24th World Congress on Intelligent Transportation Systems (ITSWC 2017), Montreal, Canada, 29 October - 2 November 2017.
36. Mihaita A. S, Cai Chen, Chen Fang, Event-triggered control for improving the positioning accuracy of connected vehicles equipped with DSRC, International Federation of Automatic Control World Congress (IFAC WC 2017), 9-14 July 2017, Toulouse, France.
37. Mihaita A. S, Tyler Paul, Menon Aditya, Wen Tao, Ou Yuming, Cai Chen, Chen Fang, "An investigation of positioning accuracy transmitted by connected heavy vehicles using DSRC", Transportation Research Board 96th Annual Meeting (TRB 2017), Washington D.C., January 8-12, Paper number 17-03863, 2017, https://pubsindex.trb.org/view/2017/C/1438533.
38. Mihaita A. S., Benavides, M., Camargo, M., "Integrating a mesoscopic traffic simulation model and a simplified NO2 estimation model", 23rd World Congress on Intelligent Transportation Systems (ITSWC 2016), Melbourne, Australia, 10-14 October 2016.
39. Mihaita A.S., Camargo, M., Lhoste, P. , " Evaluating the impact of the traffic reconfiguration of a complex urban intersection ", 10th International Conference on Modelling, Optimization and Simulation (MOSIM 2014), Nancy, France, 5-7 November 2014 (accepted on 18th of July 2014).
40. Mihaita A.S., Camargo, M., Lhoste, P. "Optimization of a complex urban intersection using discrete-event simulation and evolutionary algorithms", International Federation of Automatic Control World Congress (IFAC WC 2014), Cape Town, Africa, 24-29 August 2014.
41. Issa, F., Monticolo, D., Gabriel, A. , Mihaita, A.S., "An Intelligent System based on Natural Language Processing to support the brain purge in the creativity process", IAENG International Conference on Artificial Intelligence and Applications (ICAIA 2014), Hong Kong, 12-14 March, 2014.
42. Monticolo, D., Mihaita A.S., "A Multi Agent System to manage ideas during Collaborative Creativity Workshops", 5th International Conference on Future Computer and Communication (ICFCC 2013), Phuket, Thailand, 26 May 2013.
43. Mihaita A. S., Mocanu S., "Un nouveau modéle de l'énergie de commande des systèmes stochastiques à commutation", Septième Conférence Internationale Francophone d'Automatique (CIFA 2012) Grenoble, France, 4-7th of July, 2012.
44. Mihaita A. S., Mocanu S., "An Energy Model for the Event-Based Control of a Switched Integrator", International Federation of Automatic Control World Congress (IFAC WC 2011), Milano, September 2011.
45. A-S. Mihaita, H. Li, Z. He and M. -A. Rizoiu, "Motorway Traffic Flow Prediction using Advanced Deep Learning," 2019 IEEE Intelligent Transportation Systems Conference (ITSC), 2019, pp. 1683-1690, doi: 10.1109/ITSC.2019.8916852.
46. Mihaita, A.S., Liu, Z., Cai, C., Rizoiu, M.A "Arterial incident duration prediction using a bi-level framework of extreme gradient-tree boosting", ITS World Congress 2019, Singapore, 21-25 Oct 2019
47. Mao, T., Mihaita, A.S., Cai, C., Traffic Signal Control Optimisation under Severe Incident Conditions using Genetic Algorithm, ITS World Congress 2019, Singapore,

21-25 Oct 2019.

48. Shaffiei, S. Mihaita, A.S., Cai, C., Demand Estimation and Prediction for Short-term Traffic Forecasting in Existence of Non-recurrent Incidents, ITS World Congress 2019, Singapore, 21-25 Oct 2019.

49. Rizoiu, M. A., & Velcin, J. (2011). Topic extraction for ontology learning. In W. Wong, W. Liu, & M. Bennamoun (Eds.), Ontology Learning and Knowledge Discovery Using the Web: Challenges and Recent Advances (pp. 38–60). IGI Global. https://doi.org/10.4018/978-1-60960-625-1.ch003

50. Rizoiu, M. A., Xie, L., Caetano, T., & Cebrian, M. (2016). Evolution of privacy loss in wikipedia. In WSDM 2016 - Proceedings of the 9th ACM International Conference on Web Search and Data Mining (pp. 215–224). New York, New York, USA: ACM Press. https://doi.org/10.1145/2835776.2835798

51. Rizoiu, M.-A., & Xie, L. (2017). Online Popularity under Promotion: Viral Potential, Forecasting, and the Economics of Time. In International AAAI Conference on Web and Social Media (ICWSM '17) (pp. 182–191). Montréal, Québec, Canada. Retrieved from https://aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15553

52. Mishra, S., Rizoiu, M.-A., & Xie, L. (2018). Modeling Popularity in Asynchronous Social Media Streams with Recurrent Neural Networks. In International AAAI Conference on Web and Social Media (ICWSM '18) (pp. 1–10). Stanford, CA, USA. Retrieved from https://arxiv.org/pdf/1804.02101.pdf

53. Kong, Q., Rizoiu, M.-A., Wu, S., & Xie, L. (2018). Will This Video Go Viral: Explaining and Predicting the Popularity of Youtube Videos. In The Web Conference 2018 - Companion of the World Wide Web Conference, WWW 2018 (pp. 175–178). Lyon, France: ACM Press. https://doi.org/10.1145/3184558.3186972

Appendix A

Multi-Agent Simulation Using MATSim - The Step-by-step Instruction

The appendix is a documentation explaining how to conduct the multiagent simulation for conventional traffic using MATSim. It is the part of the project — Multi-agent Simulation for Connected and Autonomous Vehicles in the Data61 vacation scholarship program. Note that the page numbers are reset (starting from page 1) in the appended document.

Multi-Agent Simulation Using MATSim – The Step-by-step Instruction

Mingyou Ma
mingyou.ma@student.unsw.edu.au

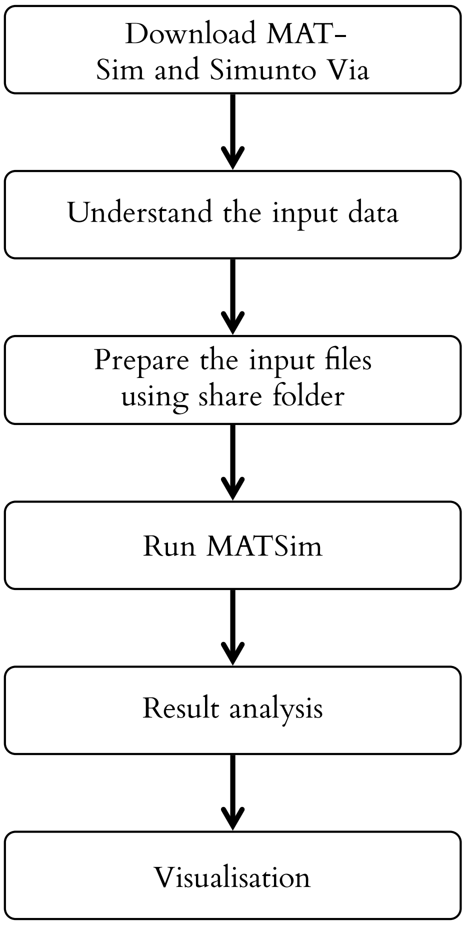
February 15, 2019

Introduction

This is a documentation explaining how to conduct the multi-agent simulation for conventional traffic using MATSim. It is the part of the project – *Multi-agent Simulation for Connected and Autonomous Vehicles* in the Data61 vacation scholarship program.

Contents

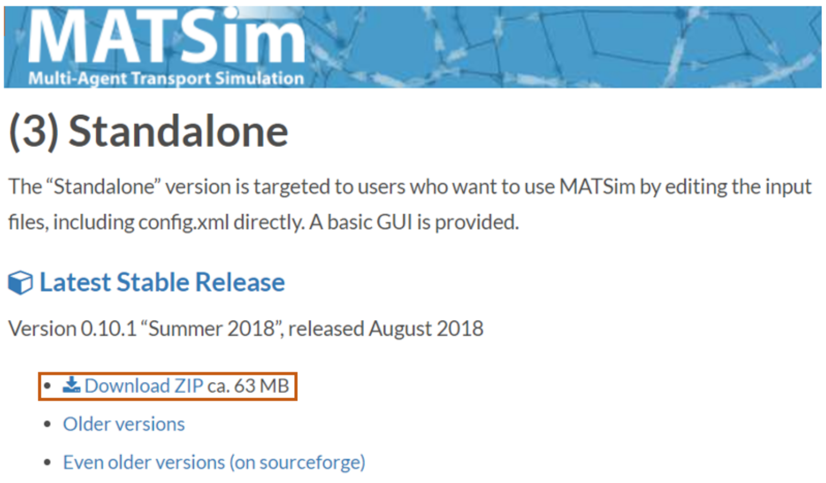
Quick Guide to Workflow



Download MAT-Sim and Simunto Via
Understand the input data
Prepare the input files using share folder
Run MATSim
Result analysis
Visualisation

1 Download and Set up MATSim

1. Visit `https://matsim.org/downloads/`.
2. Download the latest stable release (`.zip`).



(3) Standalone

The "Standalone" version is targeted to users who want to use MATSim by editing the input files, including config.xml directly. A basic GUI is provided.

Latest Stable Release

Version 0.10.1 "Summer 2018", released August 2018

- Download ZIP ca. 63 MB
- Older versions
- Even older versions (on sourceforge)

Figure 1: MATSim Download Page

3. Extract the downloaded MATSim file (`.zip`).
4. Download Eclipse IDE for JAVA developers.
   `https://www.eclipse.org/downloads/packages/release/2018-12/r/eclipse-ide-java-developers`
5. Launch Eclipse IDE.

> You should have JAVA JDK in your system to launch Eclipse IDE.
> If not, please check the tutorials for JAVA JDK installation below:
> *Tutorial for Windows installation:* `https://youtu.be/j-X-JNTlN78`
> *Tutorial for Mac OS installation:* `https://youtu.be/y6szNJ4rMZ0`
> *Tutorial for Linux installation:* `https://youtu.be/VvMYTQ_q-6o`

6. Create a new JAVA project in Eclipse IDE.
   Deselect `Use default location` and click Browse... to load the **extracted** MATSim folder.

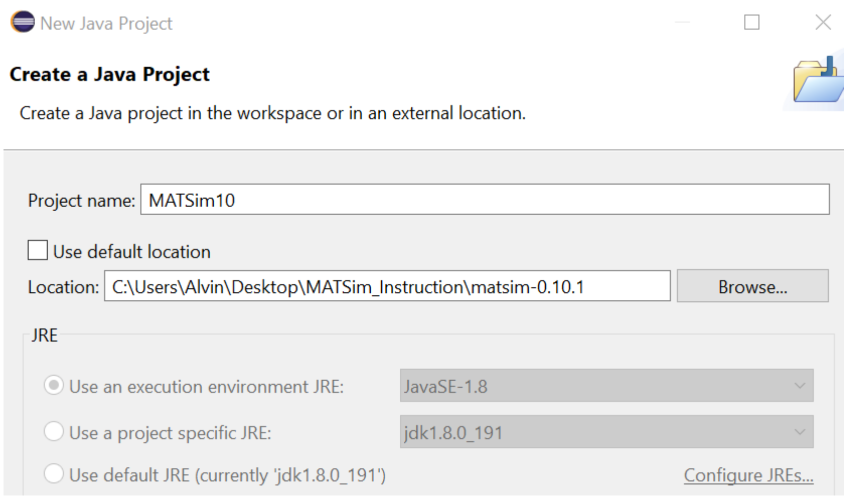

New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: MATSim10
☐ Use default location
Location: C:\Users\Alvin\Desktop\MATSim_Instruction\matsim-0.10.1 Browse...
JRE
Use an execution environment JRE: JavaSE-1.8
Use a project specific JRE: jdk1.8.0_191
Use default JRE (currently 'jdk1.8.0_191') Configure JREs...

Figure 2: Create a MATSim project in Eclipse IDE

7. Click Next then click Libraries.
   Remove `matsim-0.10.1-source.jar`.

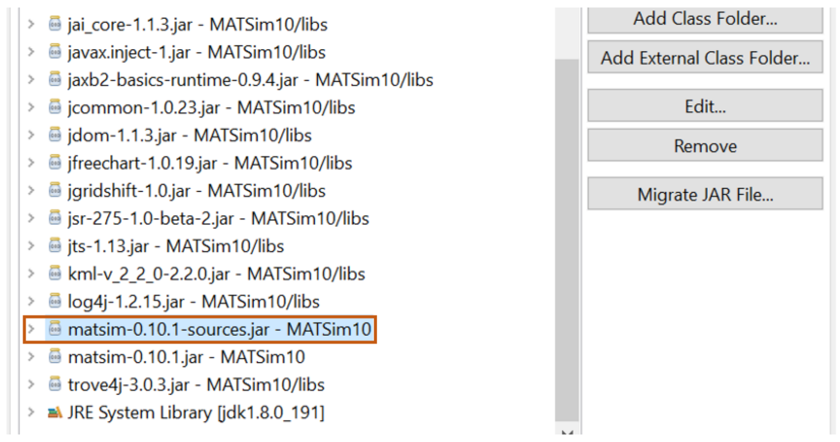

jai_core-1.1.3.jar - MATSim10/libs
javax.inject-1.jar - MATSim10/libs
jaxb2-basics-runtime-0.9.4.jar - MATSim10/libs
jcommon-1.0.23.jar - MATSim10/libs
jdom-1.1.3.jar - MATSim10/libs
jfreechart-1.0.19.jar - MATSim10/libs
jgridshift-1.0.jar - MATSim10/libs
jsr-275-1.0-beta-2.jar - MATSim10/libs
jts-1.13.jar - MATSim10/libs
kml-v_2_2_0-2.2.0.jar - MATSim10/libs
log4j-1.2.15.jar - MATSim10/libs
matsim-0.10.1-sources.jar - MATSim10
matsim-0.10.1.jar - MATSim10
trove4j-3.0.3.jar - MATSim10/libs
JRE System Library [jdk1.8.0_191]

Figure 3: Remove `matsim-0.10.1-source.jar`

8. Expand `matsim-0.10.1.jar`
   Click Source attachment:(None) » Edit... .
   In the `Source Attachment Configuration` window, check `Workspace location`. Then click Browse... .
   Select `matsim-0.10.1-source.jar`. Then click OK » OK » Finish .

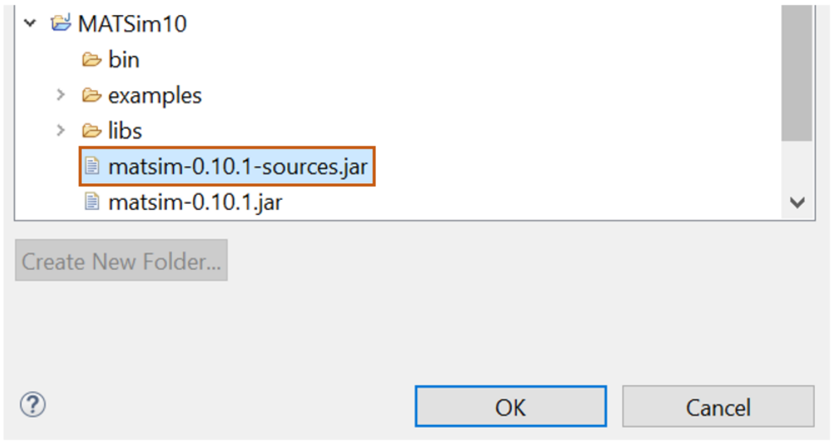
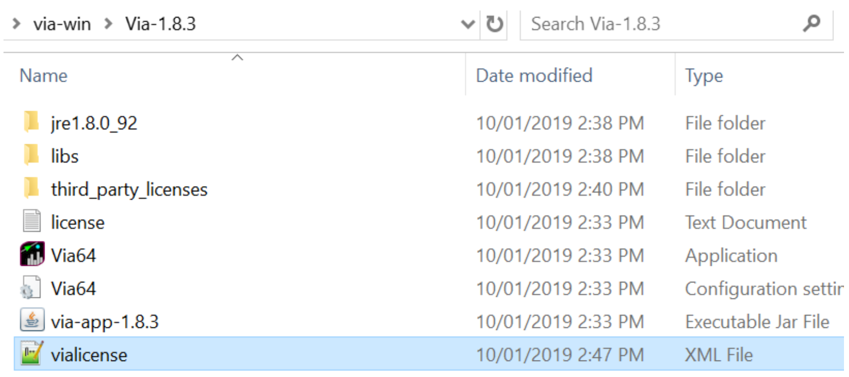


Figure 4: Source Location Selection

9. Now MATSim has been successfully installed and set up.

2 Download MATSim Visualiser – Via

1. Create a license `https://simunto.com/via/licenses/free`.
2. Check the email sent by Simunto Via and download the license (`.xml`).
3. Visit `https://simunto.com/via/download`.
4. Place the license (`.xml`) in the `Via-1.8.3` folder.



› via-win › Via-1.8.3 Search Via-1.8.3

Name	Date modified	Type
jre1.8.0_92	10/01/2019 2:38 PM	File folder
libs	10/01/2019 2:38 PM	File folder
third_party_licenses	10/01/2019 2:40 PM	File folder
license	10/01/2019 2:33 PM	Text Document
Via64	10/01/2019 2:33 PM	Application
Via64	10/01/2019 2:33 PM	Configuration settir
via-app-1.8.3	10/01/2019 2:33 PM	Executable Jar File
vialicense	10/01/2019 2:47 PM	XML File

Figure 5: Via License Loading

5. Now Via has been successfully set up.

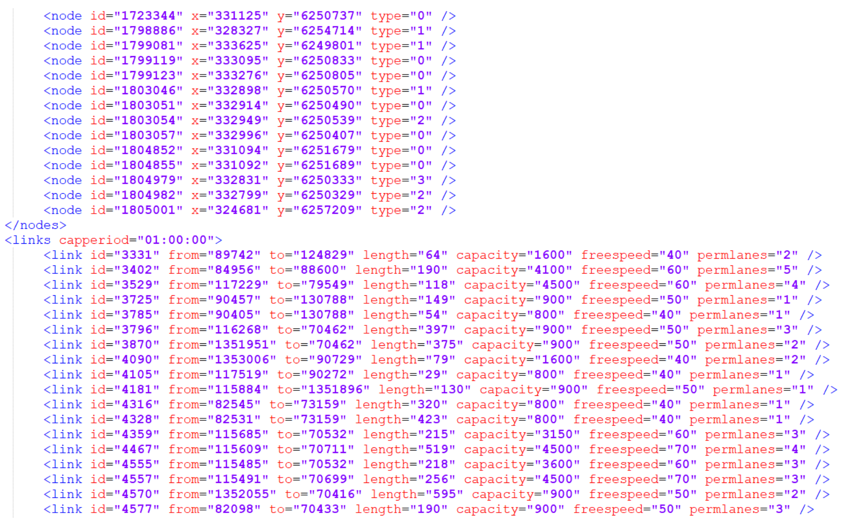
3 Input Data

3.1 `Network.xml`

In a network, there are two components, which are nodes and links. The first input file is `Network.xml`, which is used to create the traffic environment. In MATSim, links should be node-to-node-based. However, the original shape file of Victoria Road contains some open-ended links. Therefore, some fake nodes were created. Each node must be provided a coordinate. Note that, the standard coordinate system in MATSim is *EPSG:32608 – WGS84 / UTM zone 8N*. Thus, the original coordinates in Victoria Road shape file has to be converted.

MATSim requires some fundamental information about links. Typical properties of a link are capacity, length, free speed and number of lanes. Additionally, the link starting node and ending node should be also indicated.

All the network related data is stored in the `Network.xml` under the folder of *./MATSim_Victoria_Road/Victoria_Road_Input*



```
    <node id="1723344" x="331125" y="6250737" type="0" />
    <node id="1798886" x="328327" y="6254714" type="1" />
    <node id="1799081" x="333625" y="6249801" type="1" />
    <node id="1799119" x="333095" y="6250833" type="0" />
    <node id="1799123" x="333276" y="6250805" type="0" />
    <node id="1803046" x="332898" y="6250570" type="1" />
    <node id="1803051" x="332914" y="6250490" type="0" />
    <node id="1803054" x="332949" y="6250539" type="2" />
    <node id="1803057" x="332996" y="6250407" type="0" />
    <node id="1804852" x="331094" y="6251679" type="0" />
    <node id="1804855" x="331092" y="6251689" type="0" />
    <node id="1804979" x="332831" y="6250333" type="3" />
    <node id="1804982" x="332799" y="6250329" type="2" />
    <node id="1805001" x="324681" y="6257209" type="2" />
</nodes>
<links capperiod="01:00:00">
    <link id="3331" from="89742" to="124829" length="64" capacity="1600" freespeed="40" permlanes="2" />
    <link id="3402" from="84956" to="88600" length="190" capacity="4100" freespeed="60" permlanes="5" />
    <link id="3529" from="117229" to="79549" length="118" capacity="4500" freespeed="60" permlanes="4" />
    <link id="3725" from="90457" to="130788" length="149" capacity="900" freespeed="50" permlanes="1" />
    <link id="3785" from="90405" to="130788" length="54" capacity="800" freespeed="40" permlanes="1" />
    <link id="3796" from="116268" to="70462" length="397" capacity="900" freespeed="50" permlanes="3" />
    <link id="3870" from="1351951" to="70462" length="375" capacity="900" freespeed="50" permlanes="2" />
    <link id="4090" from="1353006" to="90729" length="79" capacity="1600" freespeed="40" permlanes="2" />
    <link id="4105" from="117519" to="90272" length="29" capacity="800" freespeed="40" permlanes="1" />
    <link id="4181" from="115884" to="1351896" length="130" capacity="900" freespeed="50" permlanes="1" />
    <link id="4316" from="82545" to="73159" length="320" capacity="800" freespeed="40" permlanes="1" />
    <link id="4328" from="82531" to="73159" length="423" capacity="800" freespeed="40" permlanes="1" />
    <link id="4359" from="115685" to="70532" length="215" capacity="3150" freespeed="60" permlanes="3" />
    <link id="4467" from="115609" to="70711" length="519" capacity="4500" freespeed="70" permlanes="4" />
    <link id="4555" from="115485" to="70532" length="218" capacity="3600" freespeed="60" permlanes="3" />
    <link id="4557" from="115491" to="70699" length="256" capacity="4500" freespeed="70" permlanes="3" />
    <link id="4570" from="1352055" to="70416" length="595" capacity="900" freespeed="50" permlanes="2" />
    <link id="4577" from="82098" to="70433" length="190" capacity="900" freespeed="50" permlanes="3" />
```

Figure 6: Screenshort of `Network.xml`

3.2 `AgentPlan.xml`

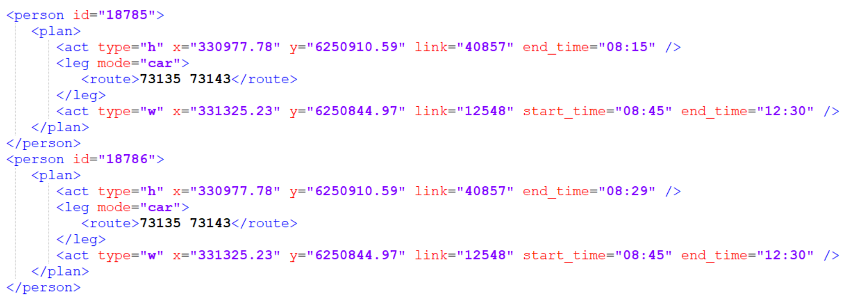
The agent's initial plan is a critical component in multi-agent simulation. An agent's plan is highly similar to a traveller's daily activity. Here are the essential components in `AgentPlan.xml` that users have to define before running MATSim.

1. `person id` - Each agent has to be assigned a unique numerical value as its ID to distinguish with other agents.
2. `act` - An act is predetermined in the MATSim `Configuration.xml`. Typically, activities could be either '`h`' for activities taking place

at home, `'w'` for activities at work, or `'edu'` for educational activities. Additional types of activities such as recreational activities could be included as well as long as it is predefined in the `Configuration.xml`.

3. `x` and `y` - The x and y values describe the coordinate of the location where an agent's activity takes place. Again, the coordinate system used in MATSim is *EPSG:32608 – WGS84 / UTM zone 8N*.
4. `link` - An agent has to be spawned on a link which is close to its activity location.

> Note that, MATSim will not execute the agent's plan without properly defining the spawning link.



```
<person id="18785">
   <plan>
      <act type="h" x="330977.78" y="6250910.59" link="40857" end_time="08:15" />
      <leg mode="car">
         <route>73135 73143</route>
      </leg>
      <act type="w" x="331325.23" y="6250844.97" link="12548" start_time="08:45" end_time="12:30" />
   </plan>
</person>
<person id="18786">
   <plan>
      <act type="h" x="330977.78" y="6250910.59" link="40857" end_time="08:29" />
      <leg mode="car">
         <route>73135 73143</route>
      </leg>
      <act type="w" x="331325.23" y="6250844.97" link="12548" start_time="08:45" end_time="12:30" />
   </plan>
</person>
```

Figure 7: Screenshort of `AgentPlan.xml`

5. `start time` - This is the time describing the activity starting time. The execution of MATSim will not be affected without this information. However, the variation of the activity starting time will influence the agent's activity-replanning scheme.
6. `end time` By contrast, an activity terminating time for each agent is compulsory in order to run MATSim properly.
7. `leg mode` - The mode of transport an agent utilised has to be mentioned in the agent's plan. To activate the `ChangeLegMode` module which is used to redefine agent's travel mode, there should be more than one mode in the agent's plan.
8. `route` MATSim requires 'node-to-node-based' trips. In the initial agent's plan, the user has to assign a route for each agent to travel from one node to the other. The route is represented by a series of node ID in a sequential order.

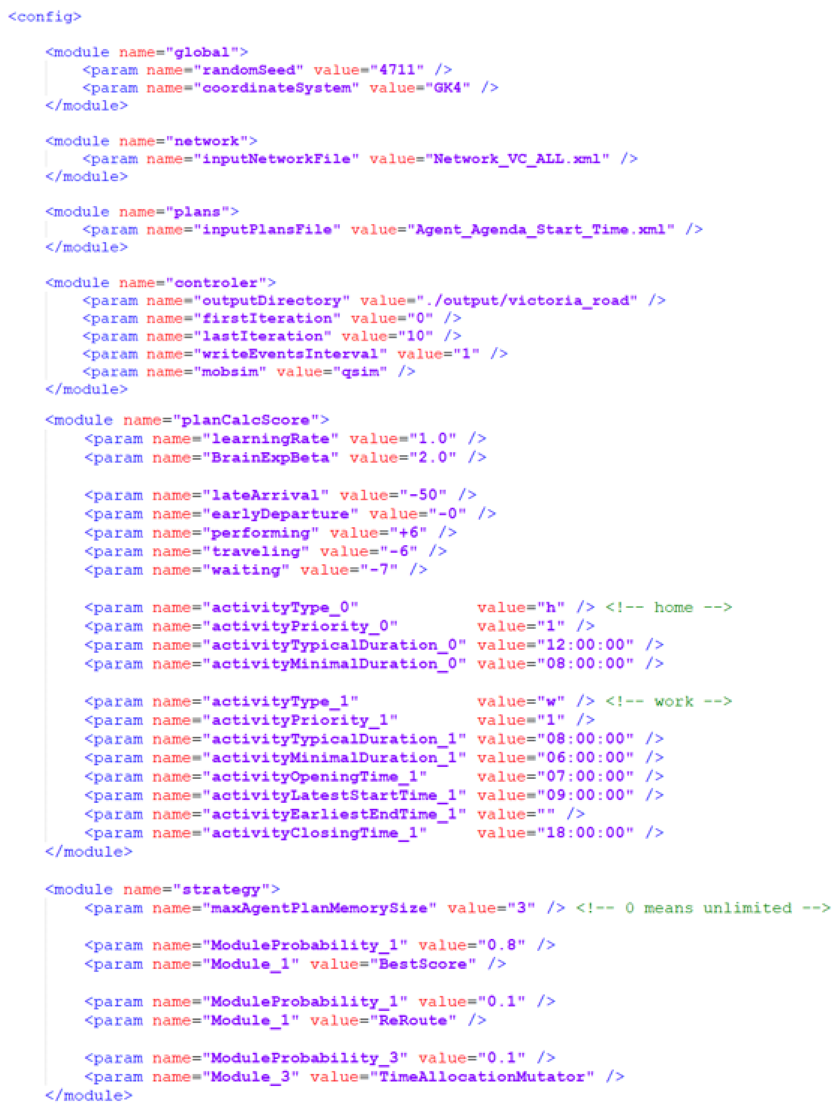
> Caution: An agent's plan has to be terminated at an `act`. And the sequence of an agent's act must follow **`act - leg - act - leg - ...- leg - act`** pattern. Otherwise, MATSim will return a series of errors.

3.3 `Configuration.xml`

MATSim is configured in the `Configuration.xml` file, building the connections between the users and MATSim and containing a settings list that influence how the simulation behaves.

All configuration parameters are simple pairs of a parameter name a parameter value. The parameters are grouped into logical groups; one group has settings related to the `Controler`, like the number of iterations.

In the Victoria Road case study, some basic MATSim settings are adopted (Figure 8).



```xml
<config>

    <module name="global">
        <param name="randomSeed" value="4711" />
        <param name="coordinateSystem" value="GK4" />
    </module>

    <module name="network">
        <param name="inputNetworkFile" value="Network_VC_ALL.xml" />
    </module>

    <module name="plans">
        <param name="inputPlansFile" value="Agent_Agenda_Start_Time.xml" />
    </module>

    <module name="controler">
        <param name="outputDirectory" value="./output/victoria_road" />
        <param name="firstIteration" value="0" />
        <param name="lastIteration" value="10" />
        <param name="writeEventsInterval" value="1" />
        <param name="mobsim" value="qsim" />
    </module>

    <module name="planCalcScore">
        <param name="learningRate" value="1.0" />
        <param name="BrainExpBeta" value="2.0" />

        <param name="lateArrival" value="-50" />
        <param name="earlyDeparture" value="-0" />
        <param name="performing" value="+6" />
        <param name="traveling" value="-6" />
        <param name="waiting" value="-7" />

        <param name="activityType_0"            value="h" /> <!-- home -->
        <param name="activityPriority_0"        value="1" />
        <param name="activityTypicalDuration_0" value="12:00:00" />
        <param name="activityMinimalDuration_0" value="08:00:00" />

        <param name="activityType_1"            value="w" /> <!-- work -->
        <param name="activityPriority_1"        value="1" />
        <param name="activityTypicalDuration_1" value="08:00:00" />
        <param name="activityMinimalDuration_1" value="06:00:00" />
        <param name="activityOpeningTime_1"     value="07:00:00" />
        <param name="activityLatestStartTime_1" value="09:00:00" />
        <param name="activityEarliestEndTime_1" value="" />
        <param name="activityClosingTime_1"     value="18:00:00" />
    </module>

    <module name="strategy">
        <param name="maxAgentPlanMemorySize" value="3" /> <!-- 0 means unlimited -->

        <param name="ModuleProbability_1" value="0.8" />
        <param name="Module_1" value="BestScore" />

        <param name="ModuleProbability_1" value="0.1" />
        <param name="Module_1" value="ReRoute" />

        <param name="ModuleProbability_3" value="0.1" />
        <param name="Module_3" value="TimeAllocationMutator" />
    </module>
```

Figure 8: Screenshort of `Configuration.xml`

1. `global` – Please leave this module along with its sub-parameters unchanged in the `Configuration.xml`.
2. `network` – The value of parameter `inputNetworkFile` should be the name of network file. In this case, the value is `Network.xml`.
3. `plans` – The value of parameter `inputPlansFile` should be the name of agent's plan file. In this case, the value is `AgentPlan.xml`.

4. `controler` – Typical parameters of `controler` are:

   - `outputDirectory` – The value should be the directory for MATSim outputs.
   - `firstIteration` – The numerical value indicates the first iteration. Normally starting from iteration 0 to execute the initial plan.
   - `lastIteration` – The terminating iteration.
   - `writeEventsInterval` – the interval of writing the `event.xml` file.

   The `event.xml` will be detailed in the later chapter.

   - `mobsim` – Leave the value of this parameter as "qsim".
   - `routeingAlgorithmType` – The recommended value is "AStarLandmarks"

5. `planCalcScore` – Typical parameters of `planCalcScore` are:

   - `lateArrival`, `earlyDeparture`, `performing`, `traveling`, `waiting` – These are the rewards or penalties (scores) that each agent could get. And a score is used to indicate whether the plan is performing better than the others.
   - `activityType` – As mentioned in section 3.2, in the agent plan, the `act` has to be cleared. The value of `act` in `AgentPlan.xml` file should be consistent with the value defined in `activityType`. Some commonly utilised values are 'h' for home-based activity, 'w' for work-based activity, and 'edu' for educational-related activity.

6. `strategy` – At the end of each iteration, MATSim evaluates each agent's plan and performs the re-planning task to optimise agents' daily activities by executing several `strategy`-related modules as below:

   - `Module_X` – This parameter has the value describing the name of strategy-replanning module. Typical replanning modules are `BestScore` (selecting the plan with the highest score from the previous iteration), `TimeAllocationMutator` (changing the departure time of agents), and `ReRoute` (reassigning an agent a new route).
   - `maxAgentPlanMemorySize` – In each iteration, MATSim generates a number of plans for each agent and stores internally. At the end of each iteration, some generated plans will be chosen as the plan for execution in next iteration. The best `X` number of plans will be kept, which is determined by the value of `maxAgentPlanMemorySize`.

Note that maxAgentPlanMemorySize $= 0$ means that MATSim will keep all the generated plans, but this would reduce the MATSim operation speed.

4 How to Use Share Folder

Two folders are included in the share folder i.e. Victoria_Road_Input and TripPlan .

MATSim input files (`Configuration.xml`, `Network.xml` and `AgentPlan.xml`) are in Victoria_Road_Input. And data generation scripts and other processed data are in TripPlan folder. The files in TripPlan folder will be detailed in this section.

1. Folder Dynamic_OD_matrices_Victoria_Rd
   - 9 original unprocessed demand matrices (`.txt`) which are the outputs generated by AimSun.next, describe the number of trips produced by one centroid and attracted by the other centroid.
   - `missing_centroids_from_shp_file.txt` – some nodes that are in the O-D matrices could not be referred back to the shape file. This `.txt` file includes all the missing centroids.
2. Folder Demand_csv
   - 9 demand matrices with coordinates of centroids (`.csv`). Note that only those O-D pairs that have non-zero trips are presented in these matrices, meaning that those zero-trip O-D pairs have been filtered.
3. Folder Network_Data
   - `Node_GJ.csv` is the original node related data that extracted from `nodes.shp`, with *EPSG:32608 – WGS84 / UTM zone 8N* coordinate system.
   - `Final_Nodes_Victoria_Road.csv` contains all nodes data including the created fake nodes as mentioned in section 3.1 along with the coordinates.
   - `Final_Links_Victoria_Road.csv` contains all links data with the essential information such as free speed, lanes, nodes connecting the links, length, etc.
   - `Centroid_Coordinate.csv` has coordinates of centroids that actually generated or received trips.
   - `Edges_Dijkstra_Times.csv` is a tuple-based dataset, containing the information about from_node, to _node, link travel time and link ID. This file is used to initialise the Dijkstra's algorithm.
4. Folder Regional_Nodes

- Vicinity_Nodes_Centroid.csv stores the allocated nodes that belong to their nearest centroids.

5. Folder Script

- 1.Node_Pairing_Trip_Assignment_Nodes_Centroid.ipynb - This code converts the centroid-to-centroid trips to node-to-node trips in order to comply with the MATSim requirement. All the trips that generated or received by each centroid is evenly distributed to the adjacent nodes within one TAZ unit. Thus, the Vicinity_Nodes_Centroid.csv and 9 O-D matrices are loaded. The output files are stored in .csv format under Demand_Node 〉 Complex.

- 2.Agent's_Plan_Creation.ipynb – This program creates the actual agent's plan by integrating node-to-node O-D matrices stored in Demand_Node 〉 Complex, Dijkstra's algorithm, departure time generator, and household / workplace generator. Additionally, this program is able to combine all the agenda files in one file, which is stored in Victoria_Road_Agent_Plan_Complex_Final.csv under Path_Trip_Agenda 〉 All_Time_Agenda. Note that this output file is not the final AgentPlan.xml.

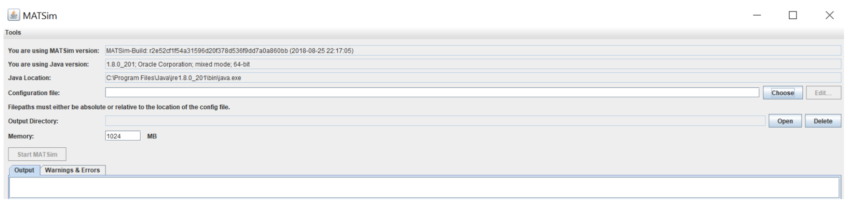
- 3.Agent's_Plan_xml_Writing.ipynb – This is the script that converts the Victoria_Road_Agent_Plan_Complex_Final.csv to AgentPlan.xml which is kept under Agent_xml.

> These three scripts should be run in a sequential order. Also, the script format is .ipynb. To execute these scripts, Juypter-Notebook (Python) is recommended. https://www.anaconda.com/distribution/#download-section

5 Running MATSim

To run MATSim, Network.xml, AgentPlan.xml and Configuration.xml have to be placed under the same directory. Here, in the share folder, please refer to folder Victoria_Road_Input.

Then launch matsim-0.10.1 placed in the extracted MATSim folder. Figure shows the MATSim basic GUI.



You are using MATSim version: MATSim-Build: r2e52cf1f54a31596d20f378d536f9dd7a0a860bb (2018-08-25 22:17:05)
You are using Java version: 1.8.0_201; Oracle Corporation; mixed mode; 64-bit
Java Location: C:\Program Files\Java\jre1.8.0_201\bin\java.exe
Configuration file:
Filepaths must either be absolute or relative to the location of the config file.
Output Directory:
Memory: 1024 MB

Figure 9: MATSim GUI

Only the Configuration.xml is required. Click Choose, then load the Configuration.xml. Then to run MATSim, click Start MATSim.

6 Output

After running MATSim, the output files will be stored under the same directory as the `Configuration.xml` (path: Victoria_Road_Input › output › victoria_road). Some typical outputs are detailed as below:

1. `X.events.xml` – Every action in the simulation is recorded as a MATSim event, be it an activity start or change of network. Each event possess one or multiple attributes. By default, the time when the event occurred is included. Additionally, information like the ID of the agent triggering the event, or the link ID where the event occurred, could be also included. The events file is an important base for post-analyses, like the visualisers (will be introduced in the following section). `X.events.xml` can be found in ITERS › it.X, where X is the iteration number.

2. `X.legHistogram_all.png` – In every iteration, a leg histogram is plotted. A leg histogram depicts the number of agents arriving, departing or en route, per time unit. Histograms are created for each transport mode and for the sum of all transport modes. Each file starts with the iteration number and ends with the transport mode. Similarly, `X.legHistogram_all.png` can be found in ITERS › it.X, where X is the iteration number.

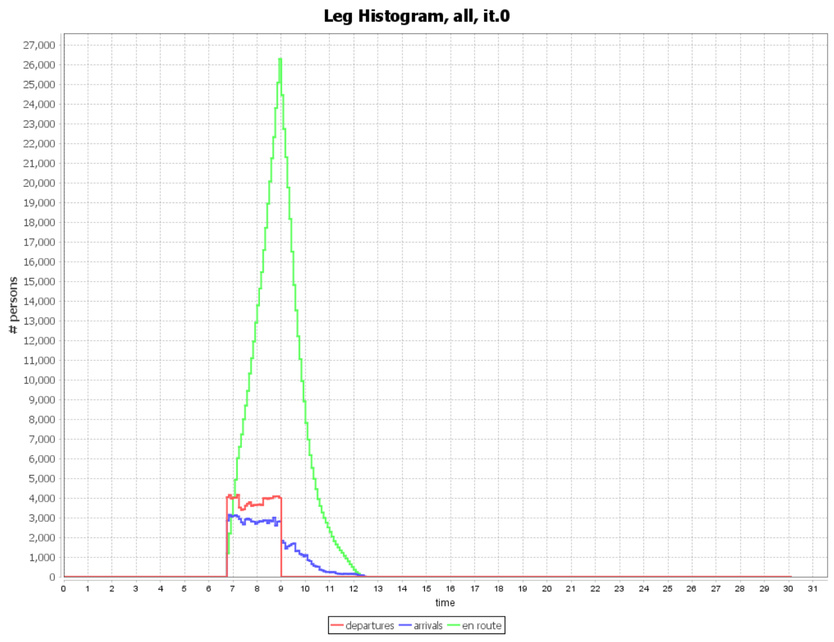


Figure 10: Screenshot of `X.legHistogram_all`

3. `scorestats.png` – Score statistics presents the average best, worst, executed and overall average o all agents' plan for every iteration. can be found in output › victoria_road.

4. `traveldistancestats.png` – Leg travel distance statistics are comparable to score statistics, but instead, they plot travel distance.

7 Visualisation Using Simunto Via

Simunto Via directly supports the loading of the typical MATSim input and output files. Network and facilities from MATSim can easily be visualized. A multitude of data can be extracted by Via from MATSim's events files, for example vehicle trajectories, activity times and link volumes. In this section, the basic Via operations are introduced.

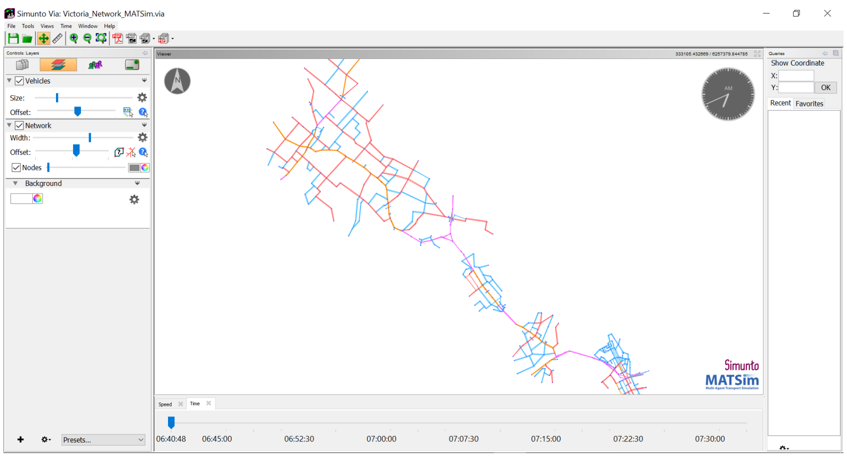
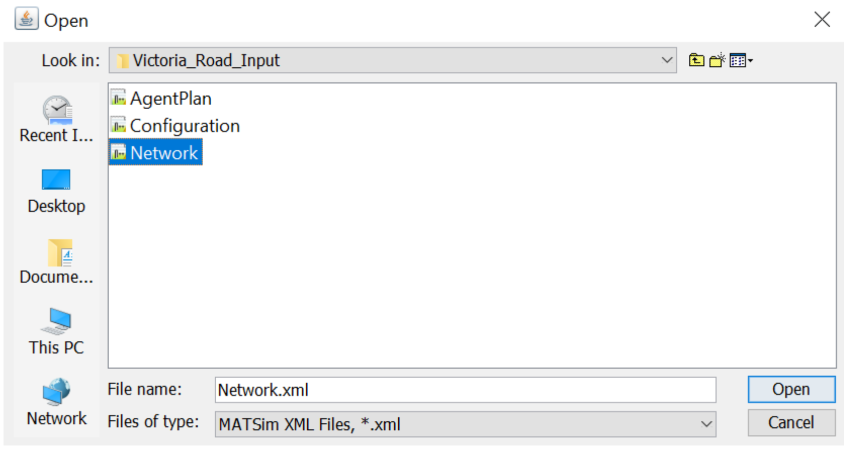


Figure 11: Screenshot of Simunto Via, the MATSim visualiser

7.1 Network Visualisation

1. Load `network.xml` - Click File 〉Add Data..., then select `network.xml`. Click Open.


Open

Look in: Victoria_Road_Input

Recent I... | Desktop | Docume... | This PC | Network

AgentPlan
Configuration
Network

File name: Network.xml

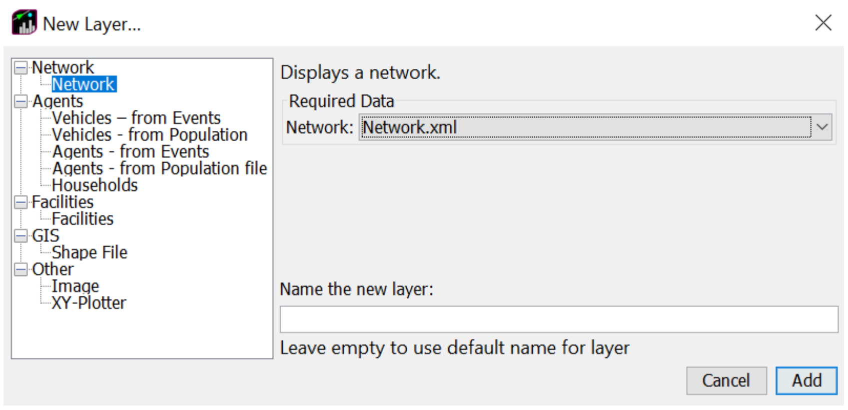
Files of type: MATSim XML Files, *.xml

Open
Cancel

Figure 12: Screenshot of `network.xml` Loading

2. Add network layer - Click File 〉Add Layer..., then select Network. In the drop down list on the right-hand side, choose `network.xml`. Then click Add (Figure 13)

3. Now, the network has been successfully loaded.



Displays a network.

Required Data

Network: Network.xml

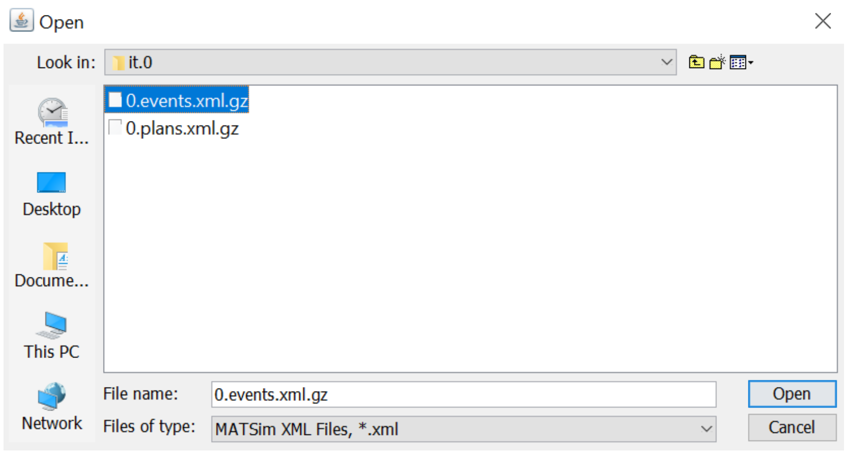
Name the new layer:

Leave empty to use default name for layer

Figure 13: Screenshot of add a new network layer

7.2 Vehicle Visualisation

1. Load `X.event.xml.gz` – Click File ⟩ Add Data..., then select `X.event.xml.gz`. Click Open.



Look in: it.0

0.events.xml.gz

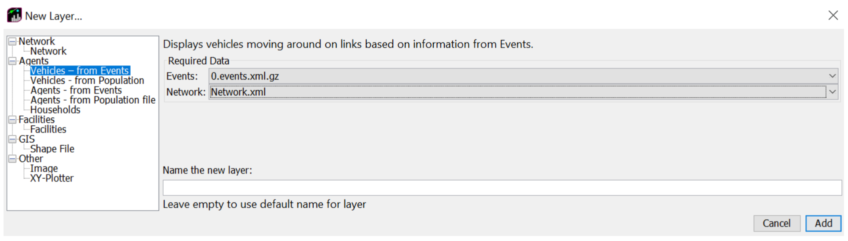
0.plans.xml.gz

File name: 0.events.xml.gz

Files of type: MATSim XML Files, *.xml

Figure 14: Screenshot of `X.event.xml.gz` Loading

2. Add event layer – Click File ⟩ Add Layer..., then select Agents ⟩ Vehicles-from Events. In the drop down list 'event' on the right-hand side, choose `X.event.xml.gz`, with Network: `network.xml`. (Figure 15)



Displays vehicles moving around on links based on information from Events.

Required Data

Events: 0.events.xml.gz

Network: Network.xml

Name the new layer:

Leave empty to use default name for layer

Figure 15: Screenshot of add a new event layer

3. Click Load Data which is located on the left-hand side of the panel to load vehicles (with maximum number 500 due to free-version limitations).

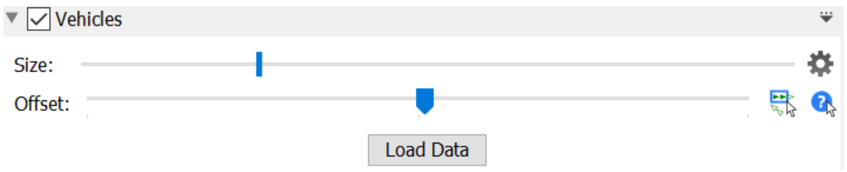


Figure 16: Screenshot of loading vehicles

4. Now, the vehicles has been successfully loaded.

7.3 Make It Move!

To observe the moving agents on the network, the speed should be set up by using the panel illustrated in Figure 17. Also, the speed can be adjusted accordingly.

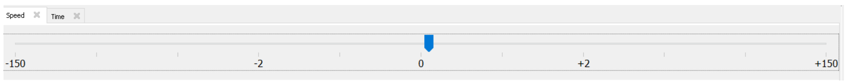
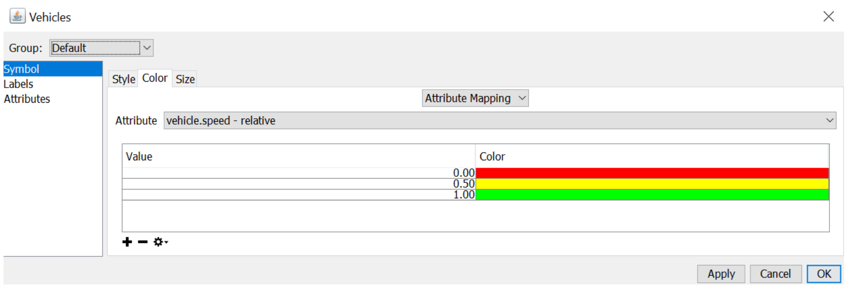


Figure 17: Speed up the agents' movement

7.4 Advanced Settings

1. Distinguish agents by their properties – The agents in the network can be distinguished by the changes of their colour based on their travel speed, mode choice, etc. This can be set by firstly clicking ⚙ on the 'Vehicles' panel. Then click color, and set the desired criteria accordingly as demonstrated in Figure 18.
2. Distinguish links by their properties – Similarly, the links can be coloured as well based on capacity, free speed, number of lanes, length or even allowed transport mode. Click ⚙ on the 'Network' panel. Then click color to set the desired criteria accordingly.



Group: Default

Attribute Mapping

Attribute: vehicle.speed - relative

Value	Color
0.00	🟥
0.50	🟨
1.00	🟩

Figure 18: Distinguishing agents