An Experiment with Perimeter Control in Downtown Seattle

Soheil Keshavarz

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Civil Engineering

University of Washington

2024

Committee:
Xuegang (Jeff) Ban
Amelia C. Regan

Program Authorized to Offer Degree:
Civil Engineering

University of Washington

Abstract

An Experiment with Perimeter Control in Downtown Seattle

Soheil Keshavarz

Chair of the Supervisory Committee:
Xuegang (Jeff) Ban
Civil and Environmental Engineering

Efficient urban traffic control requires managing city-scale networks. Traditional methods involve partitioning a network into independent subnetworks (corridor level mostly) and controlling them separately. However, time-invariant macroscopic fundamental diagrams (MFD) offer new insights for effective macroscopic-level traffic control, introducing concepts like perimeter control methods. While most studies were tested on hypothetical networks, in this study, we aim to apply the MFD-based perimeter control on a real-world city traffic network. We first work on a traffic network simulation of Downtown Seattle. After that, the simulation network is partitioned into homogeneous and practical subnetworks (regions). A macroscopic fundamental diagram (MFD) is then trained for each region to calibrate and validate the homogeneity of the partitioned regions. An MFD-based perimeter control model is applied next to control the flows in/out of each region and balance the congestion (represented by the number of vehicles) in each region. In the end, the results and findings of testing the perimeter control strategy on our real-world regions are presented, and challenges and further improvements on the method are discussed.

Keywords: Traffic simulation, Network partitioning, Macroscopic fundamental diagram (MFD), Instantaneous dynamic user equilibrium (IDUE), Point-queue model, Perimeter control

ACKNOWLEDGMENTS

I would like to begin by expressing my deepest gratitude to my supervisor and committee chair, Prof. Jeff Ban, for his guidance, support, and encouragement throughout the past two years. I am also extremely grateful to my committee member, Prof. Amelia Regan, for her meaningful presence and insightful feedback. Special thanks to my lab mates at the iUTS lab, Ohay Angah, Yiran Zhang, whose assistance was crucial in navigating the challenges of this research.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Real-world urban traffic control needs to be done for a city-scale network in order to increase the efficiency of the whole network. Traditional methods involve partitioning the network into mostly-corridor-level independent subnetworks and then controlling each subnetwork separately. Optimizing a city-scale network by dividing the entire network into such subnetworks is almost impossible and highly expensive and inefficient. The existence of macroscopic fundamental diagrams (MFD; see [15]) has provided new insights into research for network optimization and effective multi-scale control over the recent years.

In this regard, perimeter control methods have been proposed by first partitioning a city network into homogeneous regions, fitting an MFD to each region, and then optimizing the flow in/out of each region so that the total number of vehicles in a region is kept below some critical value [17]. There is extensive literature on the concepts, methods, and algorithms for MFD-based perimeter control. One of the most recent works is proposed by Guo and Ban in 2020 [20], which considers traffic dynamics and intersections of two neighboring regions at the boundaries (i.e., the buffer zones) via the point-queue model. The buffer zone and point-queue model can better capture congestion and interactions in the boundary of two regions.

While most studies were tested on hypothetical networks, this study aims to test the mentioned MFD-based perimeter strategy on regions of a real-work network. To achieve this, first, a simulation of Downtown Seattle network during the morning peak is conducted and calibrated using Simulation of Urban MObility (SUMO) software [1]. Using one-minute and 60-minute simulation outputs, the network is then divided into homogeneous regions with well-defined MFDs based on the method introduced in [23]. In the final part of this

work, the perimeter control method developed in the iUTS lab is tested and the results are discussed.

1.1 Literature Review

Effective traffic control is crucial for managing urban congestion and enhancing the efficiency of transportation networks. The advent of the Macroscopic Fundamental Diagram (MFD) has revolutionized traffic control research by offering a comprehensive understanding of the relationships between key traffic variables such as flow, density, and speed in an urban region. Empirical studies, including those conducted in Yokohama, have validated the existence of well-defined MFDs, demonstrating their potential to optimize traffic flow on a large scale [15]. By utilizing MFDs, traffic managers can implement advanced strategies like perimeter control and adaptive traffic signal systems, which help maintain optimal traffic conditions and mitigate congestion. This macroscopic approach not only improves travel times and reduces emissions but also enhances the overall sustainability and resilience of urban transportation systems. Thus, integrating MFD-based strategies into traffic management methods and testing them on real-world simulations is a pivotal research area for addressing the growing challenges of urban mobility effectively.

Urban traffic control (UTC) in transportation networks can be broadly categorized into centralized and decentralized methods. Centralized control usually focuses on microscopic traffic control involving optimization of traffic signals and other control measures from a central point of view, aiming to minimize travel times, reduce congestion, and improve overall network efficiency by adjusting signal timings across the network simultaneously. For example, Li and Ban used a dynamic programming approach to optimize fuel efficiency and travel times of connected vehicles under a fixed-time signal control [24]. They offered their method as a component of a larger-scale optimization. However, because of being computationally demanding for large networks and vulnerable to single points of failure, having an efficient centralized UTC is a challenge.

Decentralized control methods address these issues by distributing control responsibilities

across multiple and simpler optimization problems using two approaches: distributed and hierarchical. In distributed control systems, each intersection optimizes its signal timing based on local traffic conditions while coordinating and sharing information with neighbor intersections. The main problem with distributed urban traffic controls is their limited ability to achieve global optimization and struggle with the complexity of accounting for travelers' route choices. [20]

Hierarchical control involves a multi-level optimization approach, from microscopic controls on intersections and vehicles to macroscopic controls on the network and its regions. A multi-scale or hierarchical method can split a complex control into several simpler feasible and intra-connected models. It also allows researchers to incorporate travelers' choices more conveniently. In a hierarchical control framework, proper scales with defined objectives for each scale are needed to design a practical UTC framework. A possible example is illustrated in figure 1.1, where we have a sequence of controls each of which has its specific objectives, methods, control measures, and traffic dynamics to use.

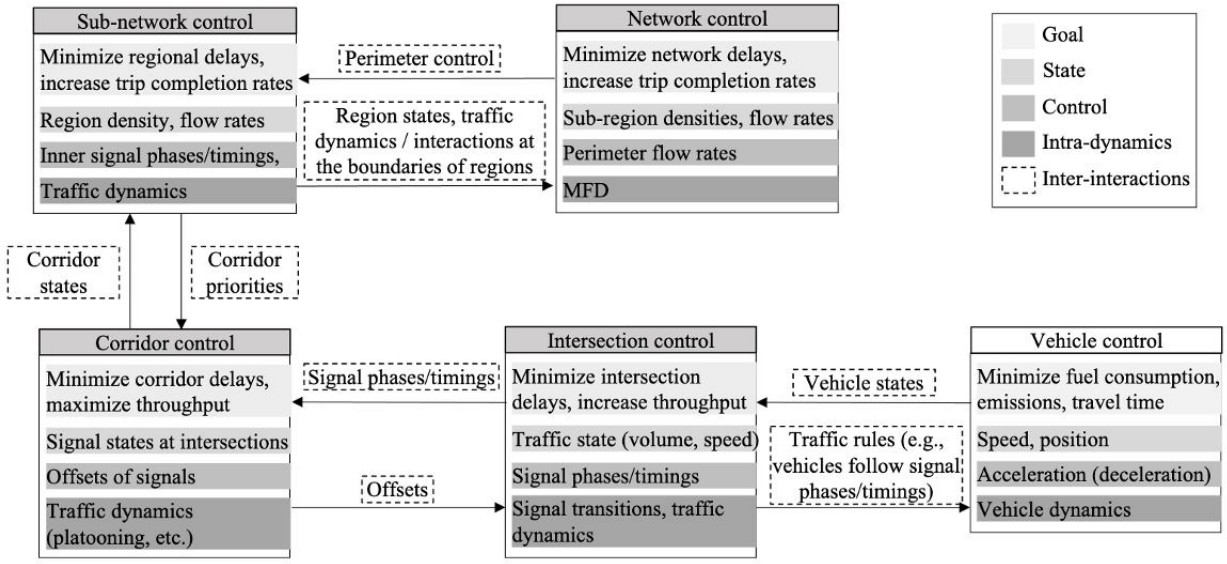


Figure 1.1: *Multi-scale control framework for UTC [19]*

A practical hierarchical (multi-scale) traffic control involves segmenting the entire network into smaller regions (sub-networks) and establishing macroscopic (regional) traffic models. To enhance partitioning and control strategies at the regional level, the concept of the Macroscopic Fundamental Diagram (MFD) has been gaining attention as an effective parsimonious model and a valuable tool for understanding the aggregate behavior of urban traffic [6].

The foundational work by [18] first introduced the idea of a macroscopic relationship in traffic networks. Later in [15], Geroliminis and Daganzo validated the existence of demand-insensitive MFDs for homogeneous urban regions in Yokohama, confirmed by other case studies (figure 3.3). Similar to a traditional fundamental diagram for a road, an MFD can depict the relationship between a flow-related measure (e.g. flow rate or rate of vehicle-miles traveled) and a density-related measure (e.g. density or number of vehicles) across a region. There has been extensive research on the properties and uses of MFD. One study found that spatial variability of vehicle density is a critical factor in the existence and shape of MFDs [5]. Another study showed the presence of hysteresis effects and heterogeneous congestion distributions can lead to significant scatter, particularly in freeway networks [16]. From a control point of view, a low-scattered and well-shaped MFD provides a comprehensive model of an urban area and can be used to balance traffic load and reduce congestion of a region.

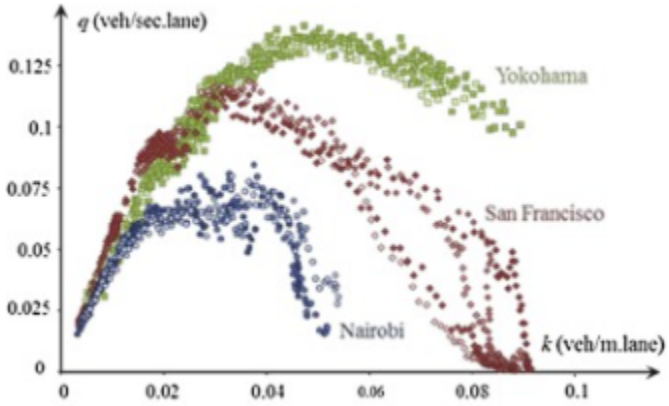


Figure 1.2: *Network flows vs. densities in empirical MFDs of Yokohama, San Fransisco, and Nairobi [2].*

As mentioned, effective hierarchical traffic control methods rely on properly partitioned networks, which is especially a challenge when working with real-world networks. In this work, we modified the proposed method by Ji and Geroliminis in [23] whose innovation lies in creating clusters with well-defined MFDs for heterogeneously congested networks, facilitating effective traffic management and control strategies. They introduce a three-step partitioning algorithm that divides the network into homogeneous regions: initial segmenting using the Normalized Cut algorithm initially proposed by [33], merging adjacent clusters with similar densities, and boundary adjustment to minimize density variance and improve spatial compactness.

In general, research on MFD-based macroscopic traffic control can be categorized as MFD-based perimeter control and MFD-based route guidance [20]. Assuming an urban network separated into regions, MFD-based perimeter control regulates the flow rates at the boundaries of neighbor regions, aiming to improve network-wide traffic performance such as the number of vehicles in each region. In one of the initial studies, a model predictive control (MPC) based perimeter control method was proposed for two urban regions with known MFDs [17]. This method was later combined with ramp metering to extend the framework for a mixed urban transportation network with two urban regions and a freeway [21]. In recent years, these frameworks have been covering more complex methods, to cover fallacies of simpler ones, as well as practical scenarios to get closer to a futuristic multi-scale control in the real world. In this regard, one work considered the interactions between macroscopic and microscopic level controls under varying penetration rates of connected vehicles (CV) [38]. In another study, the authors developed a robust perimeter control method using Sliding Mode Control (SMC) and incorporated boundary queue dynamics into MFD models [25].

MFD-based route guidance is a dynamic traffic assignment (DTA) method that on a macroscopic level, considers the distribution of travelers and their route choice among different regions of a network. This can be done using the dynamic system optimum (DSO) principle where travelers are assumed to follow the guidance of a system manager to optimize the network-wide performance of the system [22, 39, 40]. DTA can also be done assuming

the dynamic user equilibrium (DUE) principle, where individual travelers are supposed to minimize their own travel costs based on traffic conditions. Moreover, there have been recent studies attempting to combine MFD-based perimeter control with route choice behavior [7, 34]. Although such studies primarily use the DSO approach, in reality, DUE is a more realistic assumption to model travelers' behavior in current traffic systems. However, integrating DUE into perimeter-control frameworks is more challenging than DSO. DUE can be incorporated either as the predictive DUE (PDUE) or the instantaneous DUE (IDUE) [20]. PDUE assumes travelers have perfect information about future traffic conditions and choose routes based on predicted travel times, while in IDUE, travelers only know current traffic conditions and make route choices based on that, allowing route changes during travel, unlike PDUE [4]. IDUE is more practical and reflective of real-world conditions compared to PDUE [20].

1.2 Summary and Contribution

The main objective of this thesis is to apply the macroscopic control framework introduced by Guo and Ban ([20]) in a real-world setting. The mentioned work fills a gap in the literature where existing MFD-based perimeter control methods typically neglect travelers' route choices or assume that travelers always follow system-optimal routes. Their research aims to combine perimeter control and IDUE route choice behavior with the help of buffer zones as boundaries between adjacent regions.

Our work is conducted in three parts: simulation, segmentation, and perimeter control implementation. Initially, the Downtown Seattle network is simulated during the morning peak, from 5 to 10 AM using SUMO. We improved and worked on an older simulation done in the iUTS lab where PSRC activity-based travel models and OpenStreetMap were used to simulate south of Mercer St., north of South Holgate St., and west of 12th Ave. In the second part, the network is segmented using a modified algorithm based on the work by Ji and Geroliminis [23], and an overloaded simulation is performed to obtain and calibrate the MFDs of each region. These regions must be homogeneous in terms of traffic

measures, compact to prevent vehicles from re-entering a region, and have proper boundaries to facilitate practical perimeter control. Small clusters are avoided due to high statistical errors in MFD and challenges in designing effective perimeter control strategies. The final part involves applying the MFD-based perimeter control method to the partitioned regions to evaluate its effectiveness. The following chapters elaborate on these parts.

The thesis is structured as follows. First, we discuss data generation, including the initial simulation data, its shortcomings, and our improvements for the Downtown Seattle simulation regarding demand generation, network adjustments, and calibration. Chapter 3 covers network processing and cleaning, and the framework for partitioning regions and calculating MFDs, This chapter will end with the final MFD-calibrated segmentation of our simulation network. Chapter 4 details the MFD-based perimeter control method, its implementation, and the results of applying this framework to the partitioned Downtown Seattle network. The final chapter offers a summary and concluding remarks, along with a discussion of the challenges, limitations, and directions for future improvement and research.

Chapter 2

DOWNTOWN SEATTLE SIMULATION

To work on the network segmentation, an initial simulation was previously conducted in the iUTS lab. However, we noticed some issues while working on segmenting the simulated network. On one hand, demand generation was done using a simple OD table for a whole day and the time of trips in the original data was not considered. So, trips were randomly distributed based on a given distribution. This resulted in an unreliable way of inserting vehicles into the network. Moreover, vehicles were excessively using alleys whereas, in reality, they are barely used even in traffic jams. As a result, we decided to revise and refine the simulation process to address these issues before working on the network partitioning. In this document, we will refer to our initial simulation as the "old simulation".

In this research, we worked on demand generation, definition of traffic analysis zones (TAZs), speed of streets, correction of geometry, and routing. By looking at the final result and comparing it to the old simulation, we now have a better simulation in which trips coming from (or going to) downtown Seattle are better configured, trips do not use alleys, routing is more reasonable with our domain knowledge, roads with higher priority are used more, and the network is cleared around one hour after demand insertion is finished compared to former network clearance time of five hours.

To have a more robust simulation and to better replicate the real world, we worked on demand generation, simulation configuration, and demand calibration. The details for each part of this work are explained in the rest of this chapter. A more detailed report on the simulation process, including file names, can be accessed via the GitHub repository of the project [35].

2.1 Demand Generation

Data used for creating the demand input of the simulation comes from the Puget Sound Regional Council (PSRC) activity-based travel model [29]. This data includes an estimation of trips in four counties of the Puget Sound region for one workday in the year 2018 and includes household-, personal-, and trip-level information. We used the trip-level data that consisted of several trips for individuals.

In the PSRC data, origins and destinations are traffic analysis zones (TAZ). Therefore, we need to use Downtown Seattle TAZs during this simulation [28]. Moreover, to generate demand for trips coming to or going out of Downtown Seattle, we need pseudo TAZs (pTAZ) around our network. For example, if a trip is from Downtown to Northgate, it probably goes through I5 NB. So, in the SUMO demand file, we specify that this trip goes to the pseudo-TAZ defined north of I5. We call this process a "pTAZ assignment". This way, we can convert all the TAZs used in PSRC data trips that travel through Downtown Seattle into TAZs that we have in our simulation network. Figure 2.1 shows all the main TAZs and pseudo TAZs used in the simulation. Table 2.1 also shows the TAZ ID we used for each pTAZ and the location they cover. In the new simulation, pTAZs with IDs larger than 7000 were added to the old simulation where 11 pTAZ were already implemented.

Table 2.1: *The simulation network with TAZs and pseudo TAZs*

pseudo_id	5000	5001	5002	5003	5004	5005	5008	5009	5010	5011
location	I-5Nin	I-5Nout	I-5Sin	I-5Sout	I-90in	I-90out	SR-99Sin	SR-99Sout	SR-99Nin	SR-99Nout

pseudo_id	6001	7000	7001	7002	7003	7004	7005
location	Elliott Ave	south	northeast	north	northwest	upper east	lower east

In the new simulation, multiple changes are made to the demand generation process of the old simulation. Before calibration, the demand generation procedure consists of four steps. These steps are trip generation, trip filtration and pseudo-TAZ assignment, OD table and

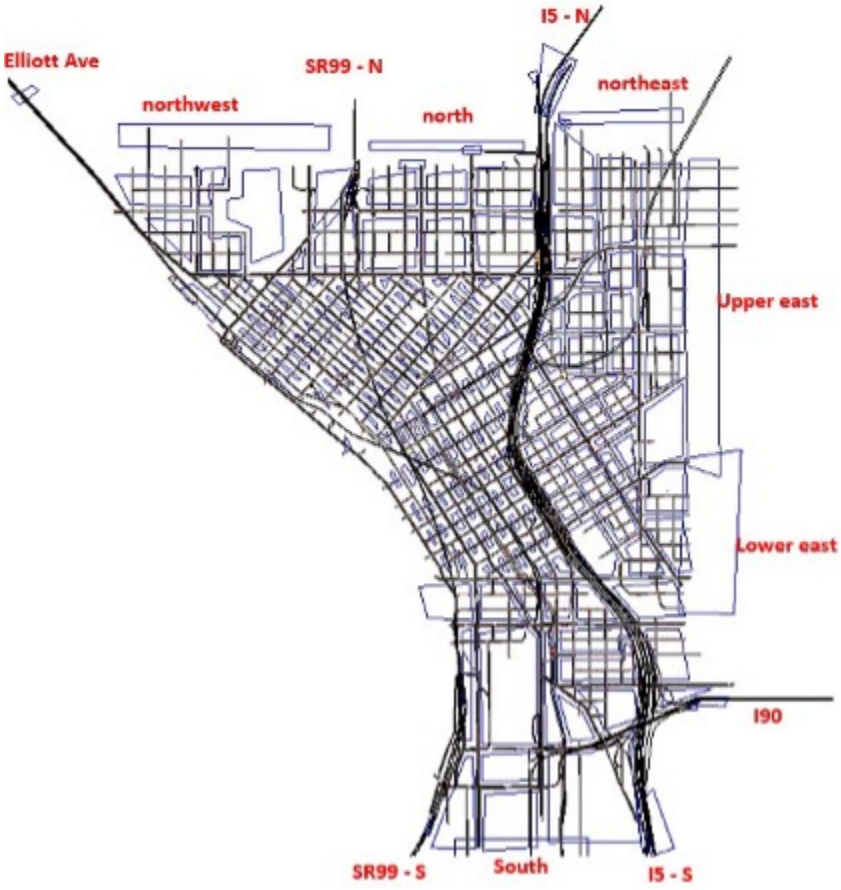


Figure 2.1: *The simulation network with TAZs and pseudo TAZs*

route file generation, and final adjustments. During these steps, unlike the old simulation process, the departure time of source data trips is taken into account, and trips that only pass through Downtown and have none of their ends in that area are also considered in data generation. Moreover, assignments to pseudo TAZs are more reasonable, and the demand format (in rou.xml or the demand file) is changed from edge-edge trips format (the result of od2trip command) into a mix of edge-edge, TAZ-TAZ, and TAZ-edge trips. We will break down each step in the following, explaining our modification to the old simulation.

2.1.1 Step 1: Trip generation

In this step, we worked on the trip data of individuals in the PSRC dataset. Each row of the input file shows a trip for a person in a working day which could be done in different modes. So for shared trips, multiple rows refer to a single trip. We first found trips done with cars. Then we tried to remove the trips that were shared but used one vehicle (duplicate car trips that the person was not the driver). In the old simulation demand generation, each of these trips was counted as one single-occupancy vehicle (SOV) trip.

Additionally, unlike the older procedure, the departure time of trips is considered and will be used later to estimate the insertion time. In the old trip generation procedure, the departure time of trips was ignored, and the insertion time into the network (departure time of the simulation demand file) was randomly assigned based on an arbitrary daily volume profile of the network (See section 2.1.3). In this case, for example, a trip from TAZ A to B that starts at 10 AM (in PSRC data) could be started at 3 PM (because of the old procedure). As a result of the modified trip generation process, we can have all the filtered vehicle trips of the PSRC data with their estimated origin and destination TAZs, departure time, and travel time.

2.1.2 Step 2: Downtown trips filtration and pseudo-TAZ assignment

Since not all Puget Sound TAZs are considered in the simulation network, and the focus is on Downtown Seattle, we only need to work on trips passing through Downtown Seattle. There are three types of trips that we can consider for our simulation network: 1) trips that start and end within our network, 2) trips that have one end out of our network, and 3) trips with both ends out of our network but passing through our network. The latter group was not present in the old procedure. The first group of trips is easy to work with, as in the simulation demand file generation (see step 3), their original information will be used. The other two groups of trips, however, are not easy to handle. When we want to simulate trips that go to (come from) Downtown Seattle, we need to guess where they exit (enter) our

network. In other words, we need to find the pTAZs that they start or finish their trip in, so their origin or destination TAZ should change. As mentioned before, this TAZ calibration process is referred to as "pTAZ assignment" in this document.

In the old procedure, a few points around the network with their respective longitudes and latitudes were selected. After that, the centroid location of out-of-network TAZs was compared to those points, they were assigned a pTAZ, and the trips with those TAZs on one end and in-network TAZs on the other end had to use those pTAZs to finish or start their trip in the simulation. In this work, however, more pseudo TAZs were used, and the assignment was done in a more deliberate way. Table 2.1 and Figure 2.1 show pseudo-TAZ IDs and the corresponding regions they cover.

To develop this step, we first assumed multiple super TAZs in King County. Super TAZs are clusters of TAZs near each other that probably use the same route when traveling to/from downtown. Some of the selected super TAZs are illustrated in Figures 2.2 and 2.4. Selection of super TAZs and extracting the list of TAZs within each cluster is done using QGIS [30].

To perform the pTAZ assignment, some random points were selected for each super TAZ and for the Downtown area. These points were used in Google Maps to guess the direction between super TAZs and different parts of Downtown and what pseudo TAZs the vehicles would pass through. Sometimes there can be two possible routes passing through different pseudo TAZs. In that case, a certain percentage of trips between the selected part of downtown and the selected super TAZ were randomly selected and assigned with the second probable pTAZ.

Furthermore, for the trips with origin out of downtown, the travel time from corresponding super TAZs to TAZs near the assigned pTAZs was estimated and then added to those trips' departure times to better estimate the time they enter the simulation network. Note that super TAZs should be neither small nor big clusters. Big clusters result in unreliable time offset estimations, and small clusters are hard to handle as the way we assign pseudo TAZs to super TAZs is based on manual work and judgment.

Figure 2.2 shows an example of how this procedure works, where a trip goes from TAZ

no.1 to TAZ no.428 (see start and end of the red arrow). The procedure assumes that this trip will enter the network through pTAZ no.5000 (I-5north in). This assumption comes from Google Maps directions between random points in the super TAZ and in the northeast TAZs in the simulation network. Based on the points we tested, in this case, vehicles use both I-5N and northeast pTAZs. Since this trip has an origin out of downtown, we take the average travel time from trips in this super TAZ to TAZs near TAZ no. 5000 (Figure 2.3). This resulted in changing the departure time from 498 (8:18) to 520 (8:40). The new departure time will be the desired time for inserting this trip into pTAZ no. 5000 (I-5 north in).

otaz	dtaz	deptm	travtime	original_o	original_d	original_deptm
5000	428	520	27	1	428	498
5000	433	458	30	1	433	433
5000	433	498	27	1	433	475
5000	433	502	27	1	433	479
5000	447	497	28	1	447	474
...	...	...	...	...	...	...
7002	432	563	25	1	432	541
7002	435	423	30	1	435	399
7003	437	422	30	1	437	395
7003	442	437	30	1	442	410
7003	442	590	25	1	442	568

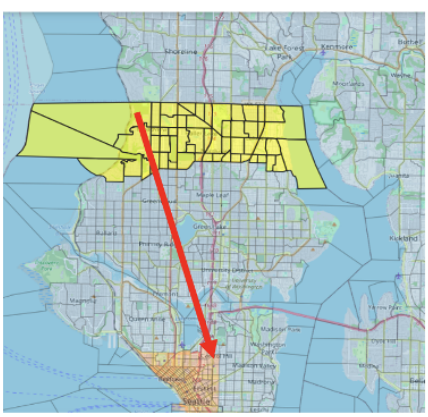


Figure 2.2: *An example of TAZ calibration process results for a trip with the origin in north Seattle*

The third type of trips that pass through our network is the trips that none of their ends are in downtown Seattle. We can assume that between some specific super TAZs, trips will pass through the downtown area. Figure 2.4 shows super TAZs that are assumed to have routes through downtown to other super TAZs.

During step 3, the total number of trips of types one and two was around 610k trips, and for the last type was about 70k. This difference could be a result of not taking all super TAZs into account. For example, trips between Bellevue and Queen Anne (northwest of

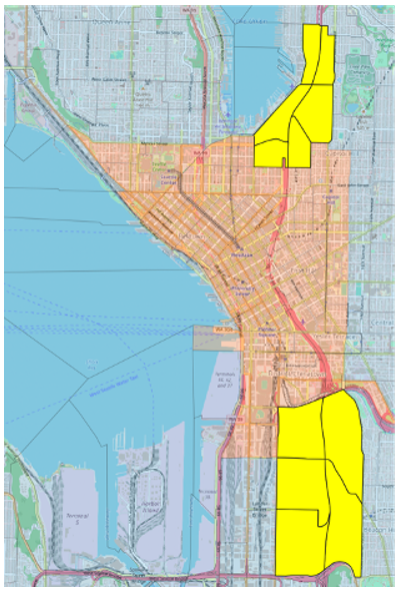


Figure 2.3: *TAZs near I5N and I5S entrances into simulation network. They are used for time offset estimation.*

downtown) were not added. Considering all possible super-TAZ pairs with routes through downtown needed more time, and we preferred not to spend too much on it.

2.1.3 Step 3: OD tables and route file (SUMO input) generation

In the third step for demand generation, route files are generated. Route files [12] are the files that SUMO directly uses to generate demand in the simulation. They consist of two components: trips and vehicle types (that define physical properties and car-following and lane-changing behaviors). These files can be generated in two ways:

1. Using `od2trips` built-in command in SUMO [9]. This command uses two files as inputs: an OD table and a TAZ file [10]. Based on TAZ source and sink links and weights, this command gives a source and a sink link for each trip. If TAZs are not defined properly or there is no connection in the simulation network between the origin and destination links, the simulation will crash. In the old simulation, one OD table for 24 hours was generated, and based on given probabilities for each hour, the departure

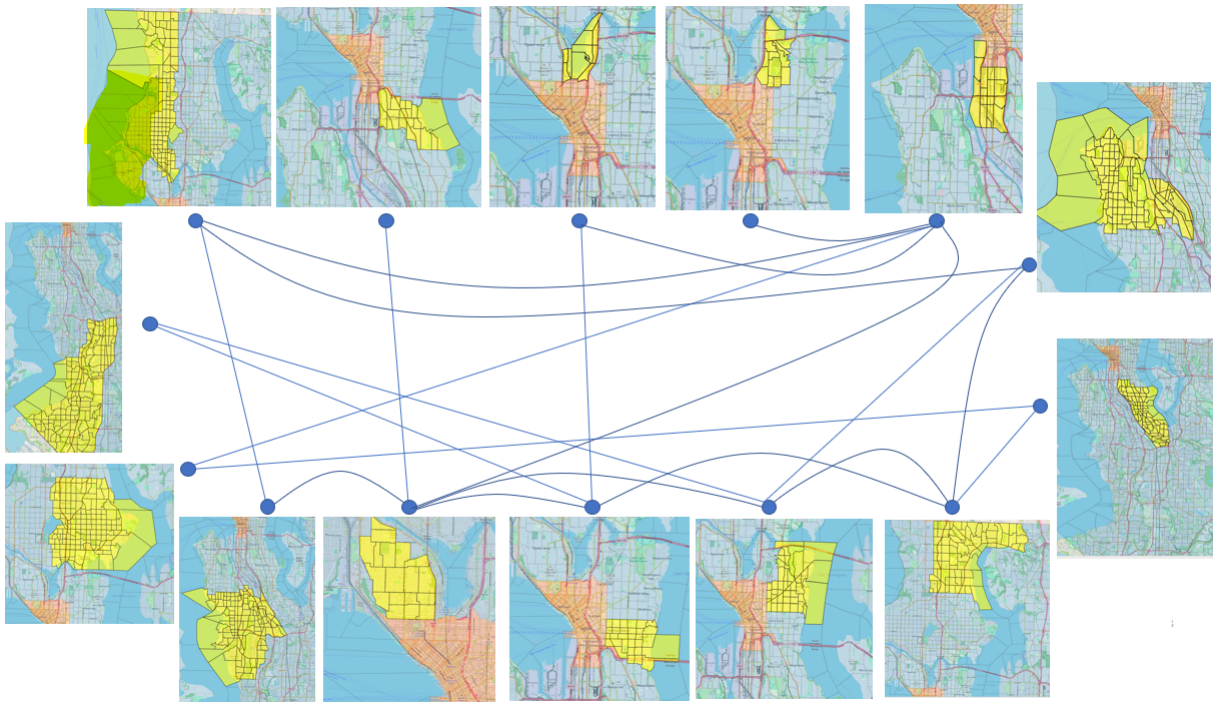


Figure 2.4: *Some of the super TAZs. Trips between connected ones are considered the third type of trips we generated.*

time of each trip was randomly generated. In the current method, we made an OD table for trips of each 15-minute interval and then called `od2trips` for each OD table. Vehicle types were manually added later to the output of this command.

2. Manually writing trips based on their origin and destination TAZ. In this method, simulation will be faster and SUMO will choose origin and destination links in a way that the trip has the shortest route. As a result, trips will start and end on the outer links of TAZs.

The first method was preferred since all links will be used in each TAZ, it looks more realistic, and the network will be more homogenous. In the second method, one would see some links in a TAZ are the destination of many trips, while their neighbor links in the same TAZs are not utilized. However, in the second method, the simulation will not crash and all the

trips will be inserted. For the final output of this step, the demand between 5 AM and 10 AM was filtered, and then the route file was generated by using the `od2trips` command and manually adding a vehicle type such as the example provided below:

```
<vType id="passenger" vClass="passenger" accel="3.2" color="yellow"
decel="3.5" length="4" maxSpeed="200" lcStrategic="1" lcSpeedGain="1"
lcCooperative="1" lcSpeedGainLookahead="5" minGap="1.5" cc1="0.9" cc2="4"
cc3="-8" cc4="-0.1" cc5="0.1" cc6="11.44" cc7="0.25" cc8="3.5" cc9="1.5" />
```

2.1.4 Step 4: Final adjustments

Adjustment refers to changing the format of the demand (route) file. In the 3rd step, demand could be generated in two formats: 1) TAZ to/from TAZ and 2) edge to/from edge. However, there is another format that is not mentioned in SUMO documentation, TAZ to/from edge format. For all trips resulted from the od2trip command, in addition to TAZs, origin, and destination links are also provided. By default, these trips will start and end on the assigned links but if for one end of the trip, the assigned link gets removed and for the other end of the trip TAZ gets eliminated, then SUMO will use the remaining edge and TAZ to start and finish a trip. In this work, this format is referred to as the TAZ-edge format. For example, as a result of only keeping the origin TAZ in the demand file for a trip, SUMO will insert the vehicle in a link in the origin TAZ that is the closest link to the destination link assigned in the od2trips command.

For two reasons, the result of the `od2trip` command from edge-edge is changed to TAZ-edge for some TAZs:

1. In practice, when people are coming into the network through pseudo TAZs, one can assume that the link by which they enter the network is the closest link to their destination in that pTAZ. So for using pTAZs, we do not have to randomly set a specific link to be used by each trip, as is the case in edge-edge format.

2. One way to get into and out of the TAZs west of I5 and north of Denny Way is by

passing Mercer St. Unfortunately, the old simulation network did not include Mercer St. and only included on/off ramps connecting this street and I5 highway. Therefore, and to make the simulation more realistic, first, some new edges and signals were made and connected to those ramps to weakly represent Mercer St. (Figure 2.13) and the time vehicles spend before reaching the ramps, and then, these edges were added to the definition of those northern TAZs. We assumed that if someone wants to travel to (come from) those TAZs using Mercer St., they will use those proxy edges. This could be the case for trips with a destination or origin far from those TAZs. Trips from closer proximity, like trips with one end in the Denny triangle, won't use those ramps. So, in the demand file, trips with one end in north-west TAZs should be in TAZ-link format and we don't use specific links in those TAZs.

2.2 Simulation Configuration

Although the demand for this simulation was generated through a better process, the simulation was not reasonable yet. The first motivation for simulation re-configuration was the excessive use of alleys, which is a rare case in the real world even in traffic jams. Moreover, in the old simulation, the network gets cleared around three hours after the time demand desired insertion times ends (10 AM), compared to the old demand where network clearance time was five hours. This shows that although the clearance time is closer to our expectation, around 1 hour, the second motivation to change the simulation was trips being inefficient with long travel times in unreasonable network clearance time.

To solve these issues, first, the pedestrian demand, which was a part of the old simulation and was not in our interest in this work, was eliminated to reduce the computational burden. Then, many flaws were discovered by observing SUMO GUI during running simulations. In the following, we will talk about the changes made in the old simulation to obtain the latest TAZ files, network file, signal controls, and configurations such as better routing and driving behaviors.

Regarding the TAZ file, we did some manual work on TAZs (such as deleting alleys

in TAZs and adding the express lane to TAZ no. 5000) and added more pseudo TAZs to the TAZ file (TAZ numbers higher than 7000 in table 2.1). Also, in the old TAZ file, the sink and source weights of links in each TAZ did not seem to follow a rule (Figure 2.5(b)). Therefore, we decided to change the weight of each link in each TAZ based on their road priority. As mentioned before (see section 2.1.3), weights show the probability of being used as origin/destination link as a result of the `od2trips` command. Figure 2.5(c) shows how a TAZ's weights would change after changing TAZ weights.

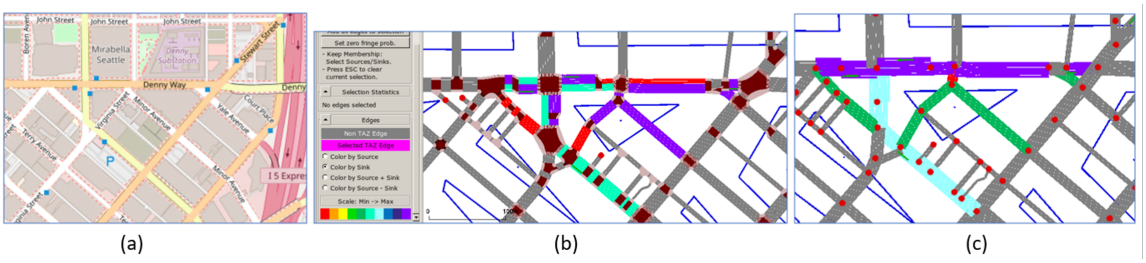


Figure 2.5: *Changing TAZ probability of being used in od2trip function from (b) to (c)*

Regarding the routing of vehicles inside the network, as mentioned before, alleys were used too much in the original simulation. Figure 2.6(a) shows an example of how alleys were used in case of traffic jams. Also, it was observed that even if main streets are under-utilized, some vehicles make additional turns and use minor streets to reach the same location. The second issue will be addressed later. To reduce the use of alleys several actions were done. These actions include increasing `weights.priority-factor`, `device.rerouting.probability`, and `device.rerouting.period` in the simulation config file (`.sumocfg`) file in the first place. After that, low-priority roads (alleys) were still used so this time, the speed in alleys was decreased then. Observations showed lower usage of alleys but it was not enough. What we wanted was rare or no use of alleys. In addition to this problem, minor roads had caused another issue. When alleys intersect streets they create junctions and SUMO by default prevents junction blocking, which is not realistic for alley junctions (figure 2.6(b)). Due to these two problems raised by the existence of alleys, we finally decided to manually remove alleys

from our network. The reason for manual removal was that deleting minor roads directly from the network file (`.net.xml` file) could have resulted in keeping some components in the network that were redundant after the deletion of alleys.

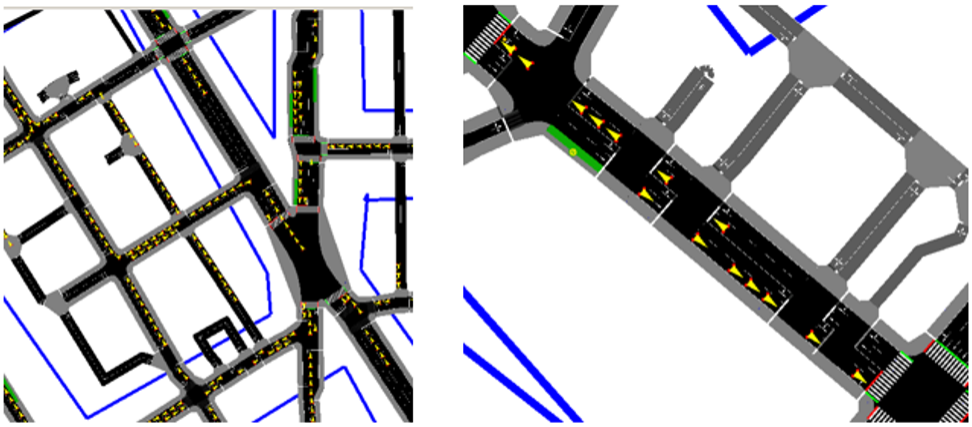


Figure 2.6: *Two problems of alleys in the main simulation: a) excessive utilization, and b) junction blocking prevention*

The next issue was the routing of vehicles. For the same O/D, people chose the roads with less priority in the old configuration, instead of choosing direct higher-priority roads without having to make a turn (Figure 2.7). Although removing alleys (roads with priority = 2) improved the routings (in instances such as figure 2.7(a)), the problem was still present (figure 2.7(b)). In addition to removing minor roads, there were other actions expected to influence the routing problem:

- One was checking the speed limit. Based on SDOT, speed limits in downtown vary between 20, 25, 30, and 40 mph in non-highway roads [27]. In the initial network, speed limit distribution was not accurate, in addition to a few links with unrealistically high speeds (in figure 2.7(b) case). In our network, speed limits started from 25 mph. We did not change the base speed limit back to 20 mph since we did not want to slow down the simulation. So while checking the speed limits of streets in our network, we used

the speeds of 25, 30, 35, and 40 mph instead of 20, 25, 30, and 40 mph respectively to correct the wrong speed limits. However, we switched to the actual speed limits later during the calibration phase.

- Another action was working on the parameters of `weights.priority-factor`, `device.rerouting.probability`, and `device.rerouting.period` in the config file, as stated previously. We hoped that the first parameter would force vehicles to use the high-priority roads more. Later we will discuss how this parameter was ineffective. For the other two parameters, we hoped that by applying rerouting, vehicles make use of higher-priority roads more and avoid constant use of lower-priority roads.

- Moreover, we reconfigured some traffic lights not calibrated before in the old simulation. The reason was that the signal timing for some intersections was inefficient, and the allocation of green time for different approaches was not well-distributed. Furthermore, some corridors lacked proper signal coordination which could result in changes in vehicles route choice. Therefore, we changed the type of some to actuated controllers and tried to adjust offset times for some signals.

Furthermore, we put effort into the correction of the network geometry and connections between lanes to have a smoother, more reasonable, and more representative simulation of the real world, as well as better routings. Regarding these changes, we:

- revised connections in some intersections. For example, a connection in an intersection could be missing, or two consecutive edges, joined via a node, were not properly connected.

- reduced U-turns. U-turns are partially frequent on local/residential/low-priority streets and are important to keep the network connected to the links on network edges, especially if those edges are assigned to some trips as origin or destination. While U-turns on network edges and some intersections seem reasonable, there are U-turn connections

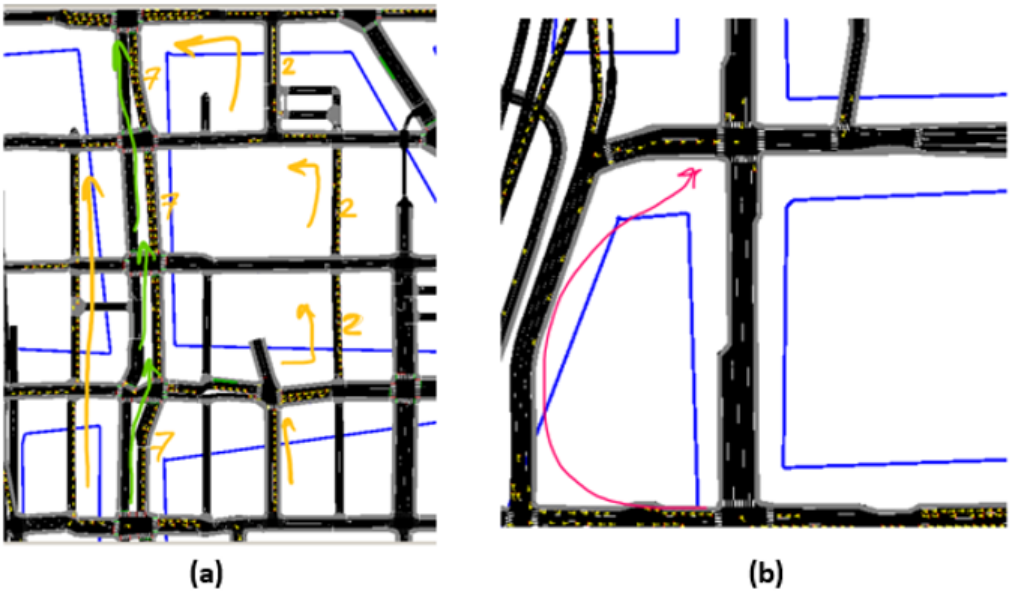


Figure 2.7: *Unreasonable routing in two instances. a) When vehicles use alleys (with priority=2) in yellow directions, and b) when vehicles use a longer route (red arrow) instead of a straight direction.*

in junctions in the middle of streets that were mostly deleted. This way, when a vehicle enters a street and wants to go in the opposite direction, it turns around at the nearest intersection, so the demand for that street in the final output files will be counted for the whole street and not just half of the street.

- removed some streets. Figure 2.8 shows an example of this. These streets could be closed (Alaskan Way viaduct) or utilized only by some organizations such as city of Seattle and King County Metro.

- updated the geometry and bust lanes of some streets, in 7th Ave N for example (figure 2.9), and Alaskan Way.

Although the changes discussed significantly improved the routing, the issue was not gone yet, and awkward routing instances in some parts of the network were still observed,

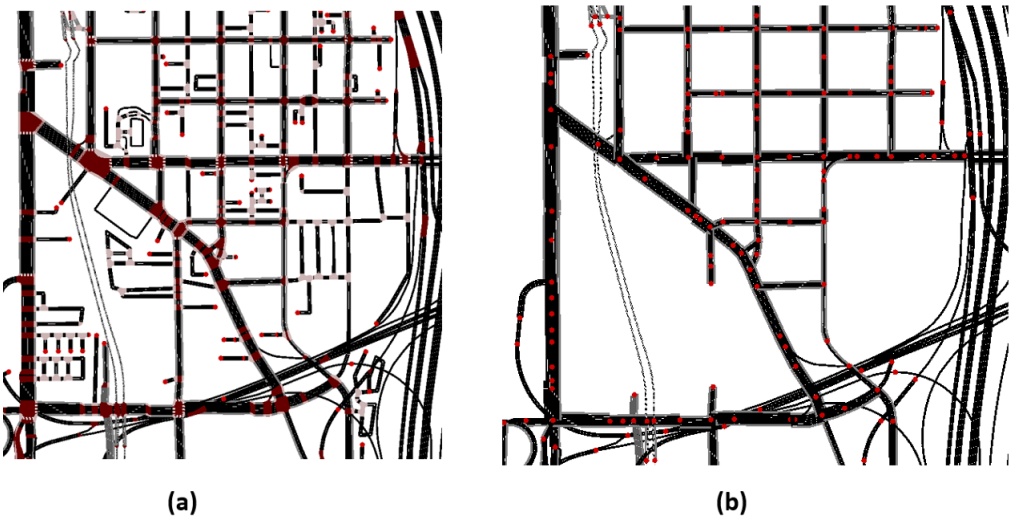


Figure 2.8: *a) a part of the original network, b) the current network after removing alleys and streets with no passenger car access*

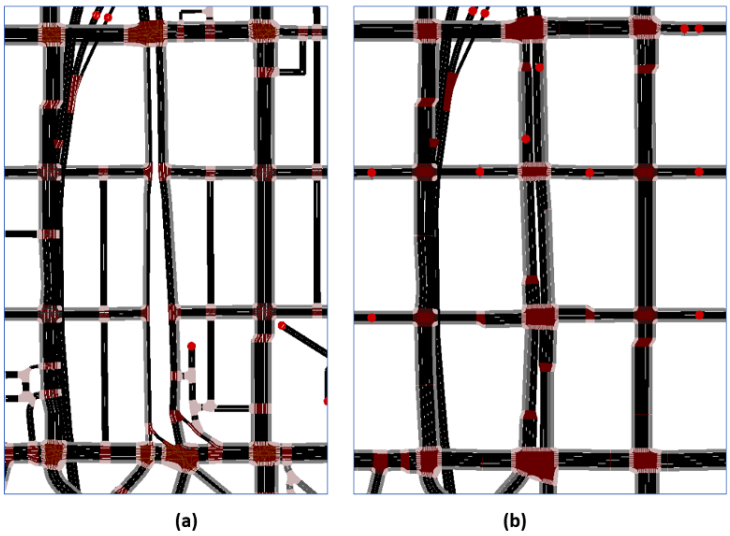


Figure 2.9: *7th Ave N a) in the original network, and b) after updating the geometry and bus lanes*

for instance, north of Denny Way and east side of I-5 (figure 2.10). Figure 2.11 shows a couple of these observations. In Figures 2.11(a) and 2.11(b), vehicles that want to travel

from red circle to blue circle should follow the green arrow in normal cases, while in the simulation they would not. In figure 2.11(c), vehicles are using the road with lower priority (=5), and although there is a queue, they still do not use the main road.

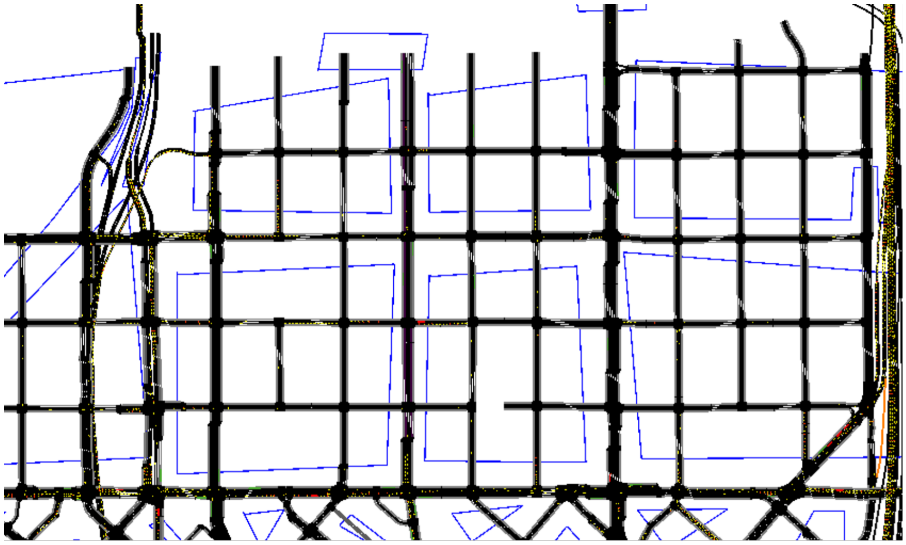


Figure 2.10: *Area of study for the latest stages of routing configuration*

To find the solution and to force vehicles to use main links in the area shown in figures 2.10 and 2.11, multiple treatments were tried such as network geometry checking, shorter rerouting periods, higher link speeds, and coordinated signals. Also, we changed the priority of some links to 12 and used `weight.priority-factor` of 1000, but this was ineffective. We concluded that this parameter does not function correctly in our simulation setting.

Finally, we realized that the calculated travel time is incorrect for the links in which the speed of the pedestrian lane (sidewalk) is lower than the speed of the vehicle lanes. The reason is that for routing a vehicle inside a network edge, SUMO considers the lane with the lowest speed in the calculation of travel time, and for some links, the pedestrian lane speed is much lower than the vehicle speed limit. In general, this should not be a big concern since by extracting a network from OSM into SUMO sidewalks and streets are modeled as separate edges. In this network however, due to the cleaning process done before

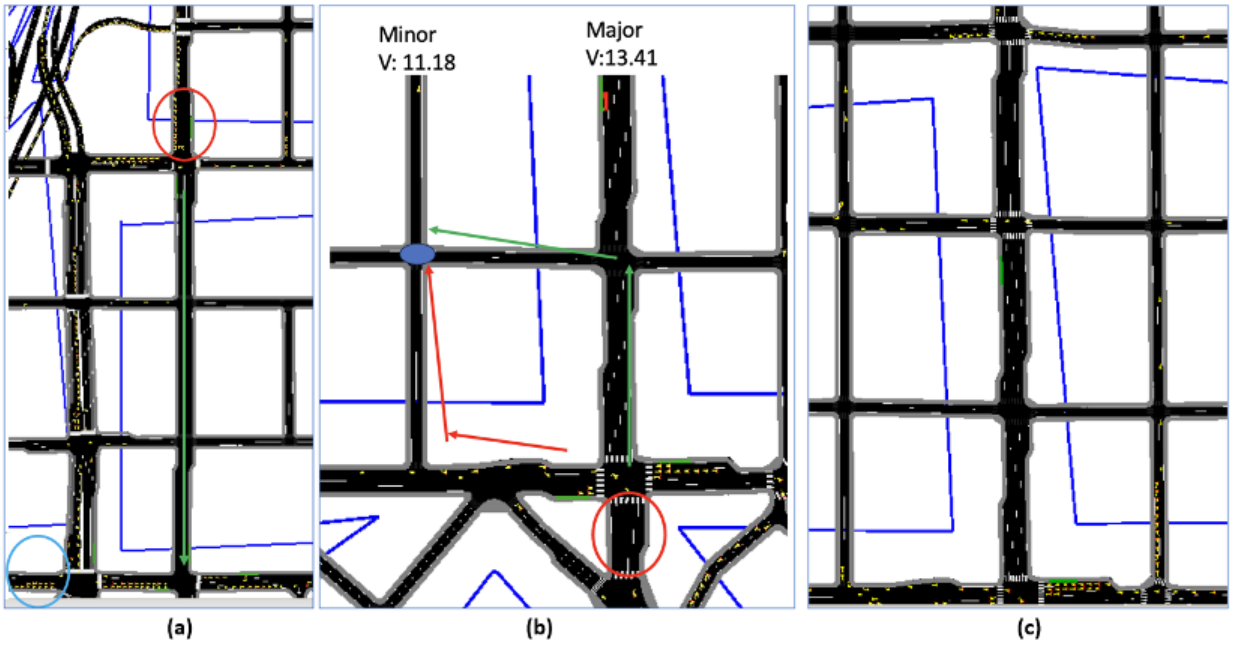


Figure 2.11: *Examples of how routing was still defective after previous treatments.*

this project, sidewalks and streets are merged, and the speed of sidewalks in some cases is different than the speed limit on their adjacent street. In the end, we ensured that the speed of sidewalks was equal to the speed of streets during the simulation to fix this issue. After that, observed routes looked better than ever. Note that in case of introducing pedestrians to the simulation, our solution will not affect their speed.

Before we started calibration, we worked on the lane-changing model parameters that are defined for each vehicle type a route file. In SUMO, there are four motivations for changing lanes: strategic (`lcStrategic`), cooperative (`lcCooperative`), tactical (`lcSpeedGain`), and regulatory (`lcKeepRight`) [14]. In this simulation, tactical and regulatory are not very important as all vehicles have the same type. The lane-changing hyperparameters are almost set to SUMO's default values. The most important parameter is `lcStrategic` which is related to lookahead distance. The more `lcStrategic` is, the earlier a vehicle will change its lane in order to make a strategic turn, a turn that allows a vehicle follow its defined route. In the original simulation, the value of `lcStrategic` was set to 2000, which was very high and

far from reality, causing super long queues for vehicles that wanted to turn. In the current setting, the value is at its default (= 1) and queues look more natural, with some vehicles trying to cut in queues before they reach an intersection or an off-ramp. Other values such as 5 or 10 were also checked but they did not affect the simulation time significantly, so we just used the default value.

2.3 Calibration

Calibration is an essential part of the simulation since it allows our simulation to represent real-world behavior using ground truth data. In this work, we mainly focused on the calibration of demand. In this section, we will explain each step we did during the calibration phase in order.

2.3.1 Travel time calibration

Looking at the main data source of this simulation, the PSRC activity-based model estimation for the Puget Sound population in 2018, all estimated trips include a departure time from the origin TAZ and an estimated travel time assumed as ground truth. In our first attempts to calibrate the simulation, we compared the travel times between all OD pairs inside the network between the simulation and the ground truth data. Trips that included pseudo TAZs were not included since there was no precise method to estimate the actual travel time of those trips inside the simulation network. To make the comparison among OD pairs inside the network, we calculated the average travel time of all trips within each OD in PSRC data and simulation output. Then the percentage error was calculated using the following equation:

$$\text{travel time error} = \frac{\text{simulation travel time} - \text{real travel time}}{\text{real travel time}} \tag{2.1}$$

In a well-calibrated simulation in terms of travel time, we need over 85% of simulation travel times to have less than one minute or 15% difference from the ground truth [26].

Figure 2.12 shows the distribution of positive and negative errors among all TAZs inside the network using the latest simulation configurations and results before calibration. In the uncalibrated simulation, the travel time error between almost all OD pairs is negative; except for some pairs. We first tried to remove very high positive errors and make more travel time errors negative. We then aimed to make a balance between positive and negative errors by lowering the speed limits of some links.

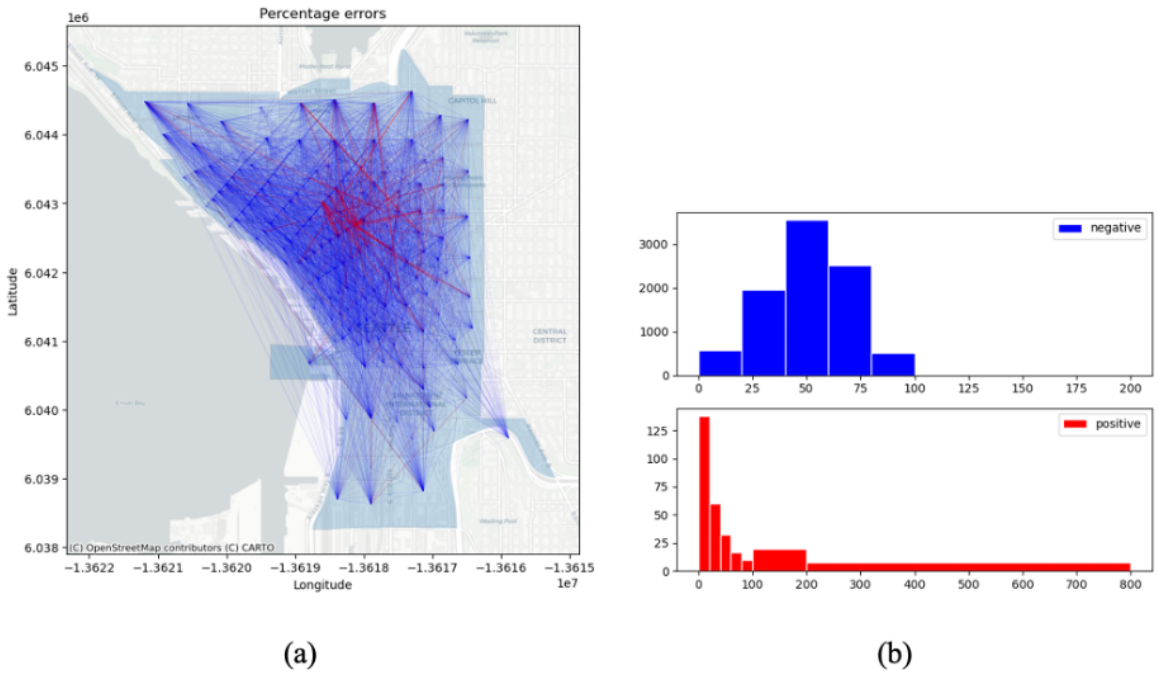


Figure 2.12: *Distribution of traveltime errors among OD pairs inside simulation network*

To deal with highly large positive errors, some signal coordination was added to intersections around Westlake. Some of these errors were for trips going into edge TAZs in the north and using I5 off-ramps to Mercer St. In reality, using those ramps was not recommended by Google Maps. To make fewer trips use those ramps as their destination, we added to the length of those ramps and reduced their speed from 50 mph to 35mph (in reality it is also 35mph). In later stages of calibration, we added some additional links and signals to those ramps to replicate Mercer St. Although we are not sure of how these changes affected

travel times, in terms of routing, these changes in the geometry should penalize the use of I5-Mercer ramps as a part of north TAZs and should result in more realistic routing decisions by SUMO for trips that include north TAZs. Figure 2.13 shows the changes in I5-Mercer St. ramps. As a result of these changes, the number of positive errors changed from around 290 to around 180, and the maximum positive error became around 240% from 700%.

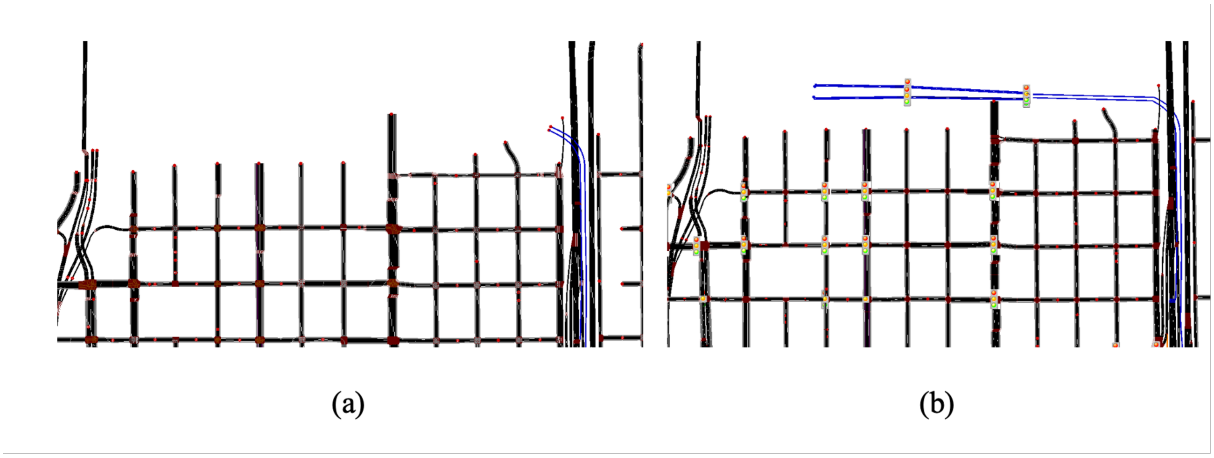


Figure 2.13: *North edge of the network a) before, and a) after changing the ramps properties and adding some edges to represent Mercer St.*

So far, we tried to deal with highly positive errors. As mentioned before (section 2.2) speed limits in the main network were starting from 20. In the simulation network, the speed limit for most edges was 5 mph higher than their real speed limit. We decided to change those to their original limits so that the travel time errors can become more balanced and skewed to zero. Figure 2.14 depicts new travel time errors as a result of decreasing speed limits. In this stage, among nearly 9300 OD pairs inside the network, there were only 300 positive errors, and mean absolute error (MAE) was 46.7%, with only 9.8% of travel time errors being less than 20%. As can be seen, we were not able to reach a good calibration of travel times by lowering the speed limits and making some adjustments in the network. In the next part of the calibration, we emphasized calibrating traffic volumes.

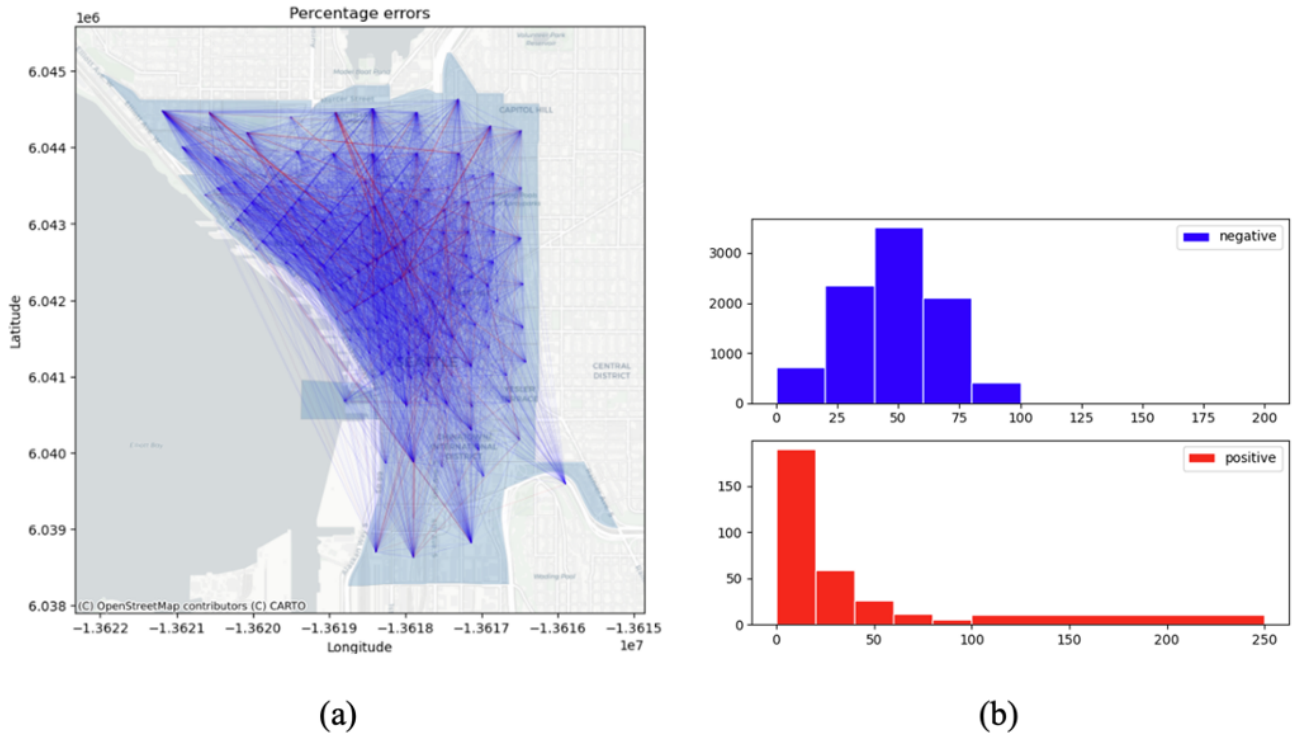


Figure 2.14: *Distribution of traveltime errors among OD pairs inside simulation network after fixing speed limits*

2.3.2 Demand and flow calibration

Calibration of demand data was done by comparing volume profiles of real data versus simulation data. Demand calibration was done in two stages. First, the volume profiles of inflows to and outflows from pseudo TAZs were calibrated (out-of-network demand calibration). Then, for some selected corridors, we tried matching the simulation profile with ground truth (in-network calibration). See the subsequent subsections for more details.

out-of-network (pseudo TAZs calibration)

We first decided to calibrate the number of vehicles that travel to or from pseudo TAZs. In other words, for each hour of simulation (5 to 10 AM), each pTAZ, and either direction (inflow or outflow) we want to make simulation numbers replicate ground truth as much as

possible. In this simulation, there are five pTAZs that represent highways and seven pTAZs that represent a part of arterials (figure 2.1). Having five hours of demand (5-10 AM) and two directions for each pTAZ, there are 120 pairs of number to be compared between real and simulation data.

We tried to use the data for the 2017-2019 period to be consistent with the PSRC model output. Only the loop readings for Tuesday, Wednesday, and Thursday were used. The data for highway pseudo TAZs was from the WSDOT loop data center [36], and for other pTAZs, SDOT traffic count studies by hour were used [32]. To obtain the volume profile for a pTAZ (either for inflows or outflows) we look for studies done on the edges of pseudo TAZs. For example, one of the useful studies for generating outflow of pTAZ 7005 (southeast) was the one done on E Cherry St on west of 12th Ave for westbound movement. Finding and aggregating such studies allows us to estimate real data for each pseudo-TAZ.

Figure 2.15 shows an example of volume profiles for the demand from I5-north before starting the demand calibration process. This profile is shown for the whole day, however, we are only interested in the calibration of our simulation period which is 5 to 10 AM (the green area in the figure). To validate our calibration, we mainly used GEH statistic which is calculated as:

$$GEH = \sqrt{\frac{(E-V)^2}{(E+V)/2}} \tag{2.2}$$

Where E is the model estimated volume, and V is the field count [26]. In figure 2.15, this validation statistic for all hours was calculated and then some values for the 5-10 AM period were reported. For total GEH, we used summation of field count and simulation data for V and E. Another statistic, valid flows, refers to when 1) model flow is within 100 vehicles per hour (vph) from real flows under 700 vph, 2) model flow is within 15% for field flows between 700 and 2700 vph, or 3) model flow is within 400 vph for flows higher than 2700 vph [26]. In an ideal scenario, the calibrated simulation demand should be either valid or with GEH of less than five for more than 85% of study hours (between 5 to 10 AM in this work). Remember that for this stage, there are 12 pseudo TAZs, five hours, and two directions,

resulting in 120 hours to be compared for pTAZ demands. However, this cannot be easily done for a complicated network and simulation like this.

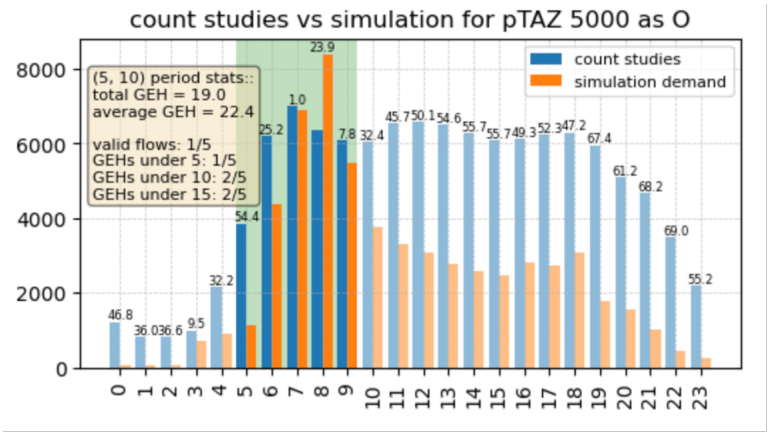
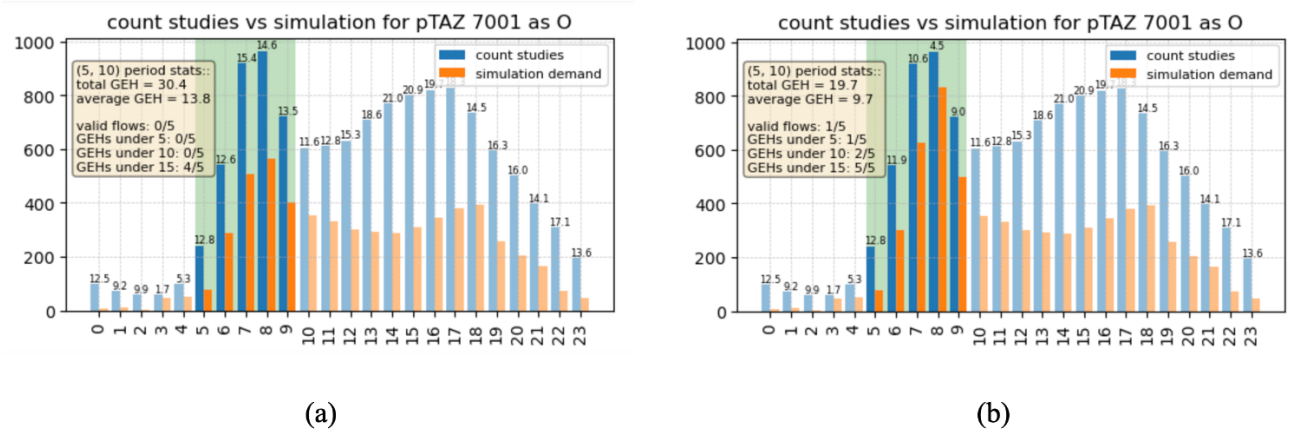


Figure 2.15: *An example of volume profiles showing trips traveling from I5N pseudo TAZ into the simulation network*

Out-of-network calibration consists of four steps: 1) pTAZ re-assignment, 2) trip transfer, 3) trip synthesis, and 4) trip reduction. In the pTAZ assignment process, the goal is to find a suitable pseudo TAZ for trips that travel into or from our network. In the re-assignment process, we added more details into the second step of demand generation, especially for trips with one side in the east or south of Seattle which were processed with huge super TAZs before. This process was explained in section 2.1.2. In the second step, trip transfer, some specific trips from over-saturated pseudo TAZs were moved into under-saturated and feasible alternatives. For example in figure 2.16(a), the pseudo-TAZ 7001 (northeast) is still under-saturated after the re-assignment step. On the other hand, trips generated from other northern pseudo TAZs (I5N, SR99N, north, and northwest) were oversaturated and had higher inflow volume than expected. As a result, we tried to select the trips that were supposed to be inserted into the network and travel to Capitol Hill via those pseudo TAZs. Then, some percentages of those trips were transferred into the northeast (7001) TAZ, while trying to improve validation statistics of pseudo TAZs involved in this example.

In the third step, trip synthesis, new trips are generated for volume profiles and for the

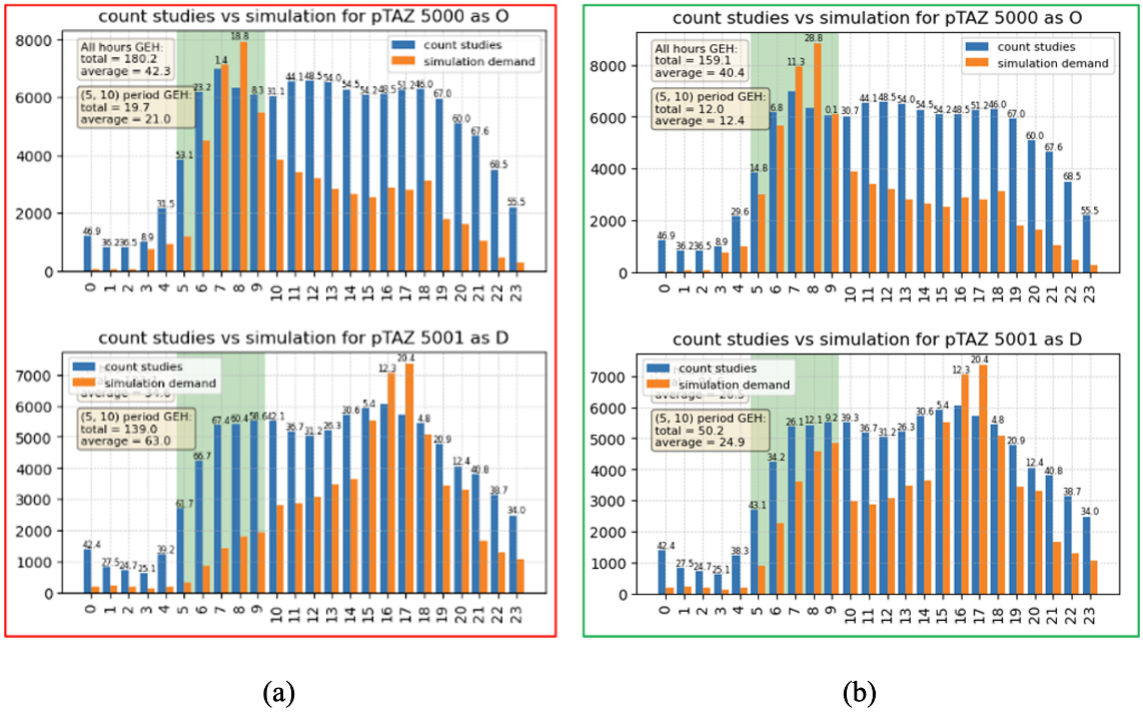


(a) (b)

Figure 2.16: *Volume profiles for northeast pTAZ originated trips (a) before trip transfer, (b) after trip transfer*

hours that do not have enough volume. In previous steps, we had tried to match simulation data with field data by better distributing the trips with an end out of our simulation network. However, we need to synthesize some trips since PSRC model (the basis of our demand generation) underestimated the total number of trips for downtown Seattle. To make new trips, we first specified a pTAZ, an hour, the direction (whether pTAZ is origin or destination), and the percentage of current trips to be added later during this process. We then looked at the trips that match the specification and by sampling with replacement, the trips to be copied were found. After that, the departure time for each sample was changed. New departure times are within 15 minutes of the original departure times. New trips with new departure times were then added to the demand file. In figure 2.17, for the trips traveling from I5N, 10% of the 6 AM demand was regenerated and added to the same hour. Additionally, for all trips going into I5N, we added 150% for all hours. The next paragraph explains this percentage.

During the out-of-network calibration, we figured out that only considering GEH statistics will not result in a good simulation as it would result in a simulation with around 240k trips between 5 AM and 10 AM and an end time of around 2 PM. Before calibration, downtown



(a) (b)

Figure 2.17: *Volume profiles for I5N pTAZ trips (a) before trip synthesis, (b) after trip synthesis*

Seattle's morning peak simulation had around 165k trips with the end time around 10:50 AM. Therefore, some rules were needed to avoid dysfunctional simulation in cost of not meeting validation criteria. In the end, we decided not to add more than 150% of the current demand for any selected specification. Moreover, when adding less than 100%, we aim for a GEH of less than five, and a GEH value of less than 10 when adding more than that. However, based on the observed congestions in the simulation we occasionally needed to get higher GEH to have a functional simulation. For example, we aimed for GEH of less than 10 (instead of five) for I90, I5S, and two east pseudo TAZs.

The final step, trip reduction, includes randomly deleting the demand based on a selected specification (pTAZ, hour, and percentage). This was done because sometimes the demand was not meeting the validation criteria (GEH statistic) due to demand saturation, and sometimes the simulation was not functioning well and there were severe congestions and we had

to reduce the demand regardless of validation criteria. Since traffic jams were not observed at entrances of pseudo TAZs, the trip reduction was only applied to the trips traveling from pseudo TAZs. Figure 2.18 shows how simulation trips from I5S to downtown Seattle were compared to field count (a) after the demand synthesis phase and b) after reducing some percentages for hours 7, 8, and 9. In the end, Table 2.2 compares GEH statistics before out-of-network demand calibration and after all four steps. As can be seen, while keeping the network clearance time around one hour after desired demand insertion time (10 AM), we were able to significantly improve statistics for inflow and outflow volumes of pseudo TAZs.

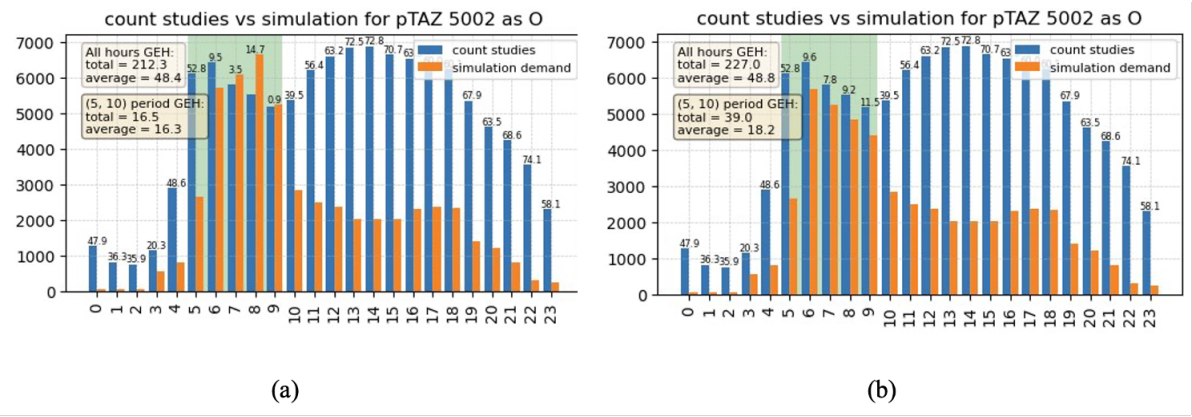


Figure 2.18: *Volume profiles for I5S pTAZ originated trips (a) before trip reduction, (b) after trip reduction*

in-network (flow) calibration

After fixing the distribution and the number of vehicles going into pseudo TAZs or coming from them, changes were made to the demand generated inside the simulation network. This was done by calibrating the flow of selected corridors. To calibrate flows, first, some corridors in the network were selected. Then, SDOT count studies for each corridor were found to get the field count data. After that, according to the exact location of selected count studies, loop detectors were implemented along selected corridors in the simulation. This way, we

Table 2.2: *Volume profiles for I5N pTAZ trips (a) before trip synthesis, (b) after trip synthesis*

pTAZ	Location	Direction	old pTAZ assignment		after reassignment, transfer, synthesis, and reduction		Changes	
			total	average	total	average	total	average
5000	I5 N	O	19	22.4	11.2	6.3	-7.8	-16.1
5001		D	136.2	61.7	57	27.6	-79.2	-34.1
5002	I5 S	O	27.2	28.7	39	18.2	11.8	-10.5
5003		D	185.4	83.5	122.3	55.9	-63.1	-27.6
5004	I90	O	71	34	15.8	7.8	-55.2	-26.2
5005		D	164.9	73.1	109.6	49.8	-55.3	-23.3
5008	SR99 S	O	11.1	12.4	1.7	2.6	-9.4	-9.8
5009		D	0.4	4.9	3.1	3.1	2.7	-1.8
5010	SR99 N	O	32.8	13.3	4.1	2.6	-28.7	-10.7
5011		D	62.2	28.2	12.5	5.6	-49.7	-22.6
6001	Elliott ave	O	27.9	13.4	4.8	3.3	-23.1	-10.1
		D	64.1	29.2	20.2	10.5	-43.9	-18.7
7000	south	O	168.6	74.1	59	28.7	-109.6	-45.4
		D	65.4	30.2	19	8.9	-46.4	-21.3
7001	north-east	O	34.3	15.6	9.6	4.6	-24.7	-11
		D	42.1	19	12.9	5.8	-29.2	-13.2
7002	north	O	11.8	8.2	9.7	4.8	-2.1	-3.4
		D	51.2	23.6	17.4	8.7	-33.8	-14.9
7003	north-west	O	10.6	5.8	2.4	2.4	-8.2	-3.4
		D	31.7	14.7	10	4.7	-21.7	-10
7004	upper-east	O	6.8	3.9	9	4.2	2.2	0.3
		D	40.9	19.7	22.2	10.5	-18.7	-9.2
7005	lower-east	O	72.5	32.2	14.6	6.8	-57.9	-25.4
		D	64.2	28.8	20	9.1	-44.2	-19.7
simulation volume			165k		211k			
simulation end time			10:50		11:05			
GEH<5			14/120 (12%)		50/120 (42%)			
GEH<10			29/120 (24%)		95/120 (79%)			
GEH<15			40/120(33%)		102/120(85%)			

were able to compare the real flow with their respective flow in the simulation. The selected corridors included: Boren Ave, Denny Way, First Ave, Yesler Way, Pine St, Stewart Ave, and Seventh Ave N. In total, 30 SDOT count studies were used for this stage.

After finishing the setup, we make changes to the demand file after comparing the 'base simulation' to the field data. By base simulation, we mean the simulation with the demand output of out-of-network demand calibration. After comparing field data and simulation data, we need to reduce or synthesize trips for specific links, hours, and percentages. Also, we may want to avoid reducing or copying trips that pass other loop detectors. Figure 2.19(a) shows field and simulation data for the first study location in the eastbound direction of

Denny Way, on the east side of Denny Way and 2nd Avenue. We then tried to synthesize trips using 30, 20, and 20 percent of trips passing this location at 7, 8, and 9 AM, resulting in Figure 2.19(b).

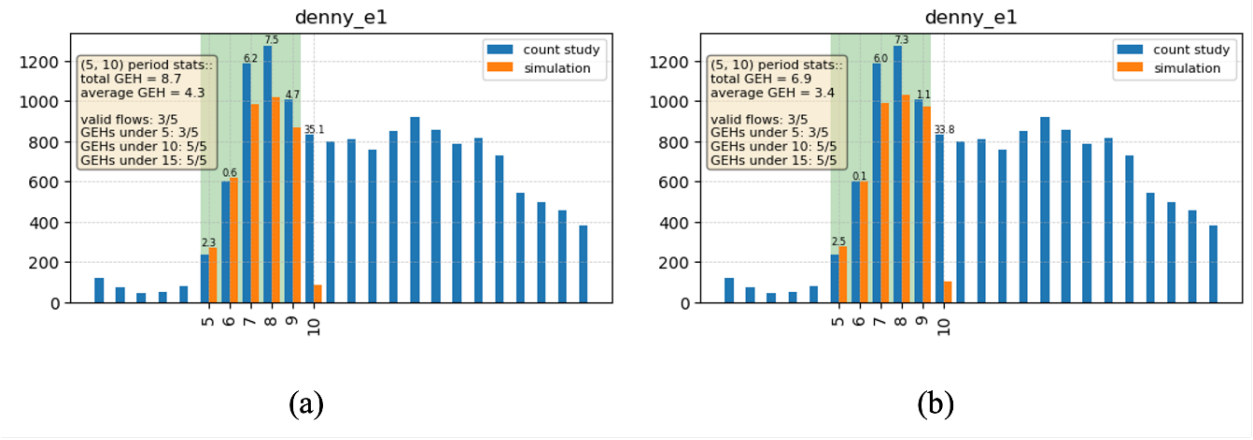


Figure 2.19: *flow profiles for east of Denny Way and 2nd Ave intersection (a) before trip synthesis, (b) after trip synthesis*

To make changes to the demand file, we first look at the vehroute output of the base simulation and at the route of individual trips (identified by trip id) [13]. Then, based on the selected specification (target link, departure hour, and links to avoid) after comparing real and simulation data, matching trips from the base simulation are sampled. If based on our desired specification for a link, we want to synthesize trips, we set the departure time of new trips within 10 minutes of the sampled trips. Also, we perform sampling with replacement to keep sampling the same trip distribution for a link. Note that to generate new trips, we do not sample from trips that have at least one end in pseudo TAZs as we already calibrated pseudo TAZs' demand in the previous section. Figure2.19 illustrates an example of how this calibration was not very helpful, and by deleting or adding new trips, we were not able to replicate our desired flow on the selected corridors. The only way by which we think corridors can be better calibrated is by using much lower rerouting values (probability and period) in the config file. However, this might potentially result in more traffic jams and less functional

simulation. Table 2.4 shows improvements for selected corridors after being calibrated at this stage.

2.3.3 Calibration of configuration parameters

While working on the simulation configuration, we touched upon multiple parameters regarding lane changing and rerouting. Since a lot of time was spent on lane changing parameters in the configuration section, we did not work further on that during the calibration process. During that, some experiments were done on car-following parameters, vehicle insertion parameters, and rerouting parameters for which we only changed `rerouting.probability` from 0.8 in previous sections to 0.7. Table 2.3 shows the parameters in the final simulation.

Regarding vehicle insertion, `departLane` and `departSpeed` parameters for all trips in the demand file were set to `free` (the most free (least occupied) lane is chosen) and `max` (The `maxSpeed` is used, the speed may be adapted to ensure a safe distance to the leader vehicle) respectively. There are two problems with this though. One is that on highways, especially I90 and I5S we see that a huge number of vehicles are not inserted in the right lane and therefore, need to change lanes. Given the large number of vehicles, this results in a lot of lane changes and conflicts between vehicles. So, we assumed that 70 percent of vehicles in those pseudo TAZs are inserted in the `best` lane (the `free` lane of those who allow the vehicle the longest ride without the need to lane change).

The other problem is that when a trip starts in a real urban network, it typically starts from the rightmost lane and with a speed of zero. To make the simulation more realistic, we assumed 50 percent of trips that were generated inside the network (and not from pTAZs) have `departLane` of first (the rightmost lane the vehicle may use) and `departSpeed` of `random` (A random speed between 0 and `maxSpeed` is used, the speed may be adapted to ensure a safe distance to the leader vehicle). For these trips, the `departSpeed` was not set to zero as we thought it might delay the departure of some vehicles, according to SUMO documentation [12].

Lastly, we adjusted some of the car following parameters from the old simulation. These

parameters include `cc0` (`minGap` in SUMO configurations) to `cc9`, and the default values in Wisconsin DOT suggestions for VISSIM were used before [37]. These parameters represent Wiedemann 99 car-following model which applies to freeways and highways. Another car-following model is Wiedemann 74 model which is applicable to arterials. In a very detailed simulation, cars follow the first model when they are on highways and the second model when they are on arterials. However, we only used the 10-parameter model since it is more detailed and compared to the 3-parameter model, the default values better matched our common sense. Also, the WisconsinDOT numbers are based on two road types, base segments and weaving/merging/diverge segments. Initially, we wanted to use type calibrators to make vehicles use minimums for the first road type in arterials and use minimums for the second road type in highway segments [11]. The downside of SUMO calibrators is that the changes they make are persistent and to make our idea work, we need to insert calibrators on inbound and outbound edges of highways. Since this needed some time to implement, and given the fact that in an urban setting, highways are influenced by the properties of the urban area they are in, we decided not to use SUMO calibrators. Instead, one general vehicle type for all vehicles in the simulation was used, and for the first three parameters (the main parameters), values between the minimum for base segments and the minimum for weaving/merge/diverge segments were used. This resulted in less congestion and therefore faster network clearance time.

2.4 Summary and results

In this work, significant changes to the initial simulation of the downtown Seattle network were established. More pseudo TAZs were created, that help us generate the trips that travel to or from out of the simulation network, and a lot of attention was paid to their assignment.

Table 2.4 shows the progress made during the calibration process by comparing validation statistics for loop detectors studied in this project. Additionally, this table shows the number of vehicles and simulation time among different demand scenarios, before calibration, after pseudo-TAZ calibration, after corridor calibration, and finally, after finalizing simulation

Table 2.3: *Calibration parameters in route(.rou.xml) file and config (.sumocfg) file*

	Parameter	Old simulation	New simulation
car following	CC0 (Standstill Distance) (m)	1.5	1.4
	CC1 (Headway Time) (s)	0.9	0.8
	CC2 ('Following' Variation) (m)	4	3
	CC3 (Threshold for Entering 'Following')	-8	-8
	CC4 (Negative 'Following' Threshold)	-0.1	-0.1
	CC5 (Positive 'Following' Threshold)	0.1	0.1
	CC6 (Speed dependency of Oscillation)	11.44	11.44
	CC7 (Oscillation Acceleration) (m/s2)	0.25	0.25
	CC8 (Standstill Acceleration) (m/s2)	3.5	3.5
	CC9 (Acceleration with 50 mph) (m/s2)	1.5	1.5
rerouting	probability	0.7	0.7
	period	-	200
lane changing	lcStrategic	2000	1
	lcSpeedGain	5	1
	lcCooperative	1	1
	lcSpeedGainLookahead	5	5
other	priority factor	0.5	10*
	departLane	always 'free'	varies**
	departSpeed	always 'max'	varies***

* This factor seems to be ineffective in our simulation

** For departure in I5S and I90: 'best' with the probability of 0.7, for departures out of pTAZ: 'first' with the probability of 0.5

*** For departures out of pTAZ: 'random' with the probability of 0.5

parameters. Additionally, Figure 2.20 shows the final distribution of travel time errors among different OD pairs in the final simulation. Compared to Figure 2.14 we can see more balanced errors, having around 1500 OD pairs with positive travel time error versus 7500 OD pairs with negative error. In the final simulation, travel time MAE is 46% with 18.0% of errors being less than 20%.

Evidently, we were able to achieve a better representation of the world during this calibration process. However, this big and complex simulation is still far from an ideally calibrated one, and there are details missing in the simulation. For example, right-turning on red lights is not permitted while in simulation of the Seattle network, this needs to be considered.

Table 2.4: *GEH statistics for 30 loops detectors around the simulation network during calibration process*

loop group	number	old pTAZ assignment		after out-of-network demand calibration		after in-network demand calibration		final simulation after fixing all parameters		Changes	
		total	average	total	average	total	average	total	average	total	average
Denny-EB	1	28.8	12.2	13.9	6.1	8.7	4.3	6.5	3.3	-22.3	-8.9
	2	46.7	20.5	41.9	18.1	36.7	15.8	35.2	15.3	-11.5	-5.2
	3	23.3	10.2	15.4	6.7	11.9	5.2	10.5	4.6	-12.8	-5.6
Denny-WB	1	33.7	14.6	18.1	10.1	13.9	9.2	11.7	9.3	-22	-5.3
	2	56.7	23.9	52.1	23.1	37	16.3	36.5	16.1	-20.2	-7.8
	3	9	4.9	4.1	2	1.4	1.6	1.5	1.4	-7.5	-3.5
Yesler-EB	1	18.9	9	3	3.5	1.2	3.6	0.6	3.4	-18.3	-5.6
	2	33.2	15.6	23.7	15.7	26.9	12.7	25.7	12.2	-7.5	-3.4
Yesler-WB	1	22.7	10.8	0.2	5.5	6.8	4.2	5.8	4.1	-16.9	-6.7
	2	5	3	9.6	4.4	6.3	3.5	6.4	3	1.4	0
Boren-SEB	1	21.9	9.6	10.9	4.9	8.7	3.9	9.7	4.3	-12.2	-5.3
	2	2.4	2.6	7.7	5.4	8	5.4	3.6	3.6	1.2	1
	3	35.1	15.7	26.8	11.7	22.1	9.7	20.5	9.1	-14.6	-6.6
Boren-NWB	1	34.3	14.9	22.7	10.3	21.6	9.9	23.6	10.5	-10.7	-4.4
	2	29.9	13.3	29.5	12.9	24.2	10.8	26.3	11.5	-3.6	-1.8
	3	26.7	12.3	31.9	13.7	30.6	13.6	32.2	14.2	5.5	1.9
Pine-EB	1	2.2	1.4	2.7	1.7	2.3	1.6	2.6	1.5	0.4	0.1
	2	15.6	7.8	15.9	6.7	10.9	4.5	11.6	4.8	-4	-3
Pine-WB	1	11.4	5.1	12.5	6.1	9.1	4.4	9.2	4.5	-2.2	-0.6
	2	4	5.2	1	8.2	2.7	3.6	0.2	4	-3.8	-1.2
	3	20.7	8.9	22.7	9.4	2.7	3.6	15.7	6.8	-5	-2.1
First-SEB	1	46.3	20.4	42.2	18.5	29.6	13.1	30.1	13.3	-16.2	-7.1
	2	21.8	9.3	2.8	2.6	0.3	2.8	0.9	2.7	-20.9	-6.6
	3	64.8	29.1	56.2	25.5	52.1	23.6	51.6	23.5	-13.2	-5.6
First-NW	1	5	3.9	5.5	5.3	3.8	4.5	2.2	3.7	-2.8	-0.2
	2	46.2	19.9	44.2	18.3	42.2	17.5	41.8	17.3	-4.4	-2.6
	3	33.4	14.2	29.2	12.3	23	10.1	23.4	10.3	-10	-3.9
Stewart-SWB	1	17.5	9.7	25	12.2	21	10	21.3	10	3.8	0.3
Seventh-NB	1	0.5	5.6	13.7	7.3	6.9	4.9	6.4	4.7	5.9	-0.9
Seventh-SB	1	14.9	11.7	1.7	8.4	2.3	3.7	1.1	3.9	-13.8	-7.8
trips		165k		211k		207k		207k			
simulation end		10:50		11:05		11:00		10:55			
valid flows		49/150(33%)		60/150(40%)		75/150(50%)		80/150(53%)			
GEH<5		33/150(22%)		44/150(29%)		61/150(41%)		64/150 (43%)			
GEH<10		74/150(49%)		90/150(60%)		104/150(69%)		104/150(69%)			
GEH<15		106/150(71%)		116/150(77%)		128/150(85%)		128/150(85%)			

Moreover, travel times for some OD pairs have very high errors, with 41 OD pairs with higher than 250% errors. We decided not to further improve the travel times for those pairs. The main reason was lack of time, but there were other limitations for this as well. For example, one way to improve travel times (for slow corridors) is increasing the speed limit. However, we could not adjust the speed of specific links because priority-based routing is not working, and the routing only relies on main streets having higher speeds than minor streets. Moreover, signal lights were mostly calibrated before in the old simulation. As a result, signal coordination could not be implemented sometimes as we decided not to change the already calibrated signals dramatically. Lastly, since the focus of this simulation was on vehicle dynamics, pedestrian simulation was not considered to reduce the simulation time

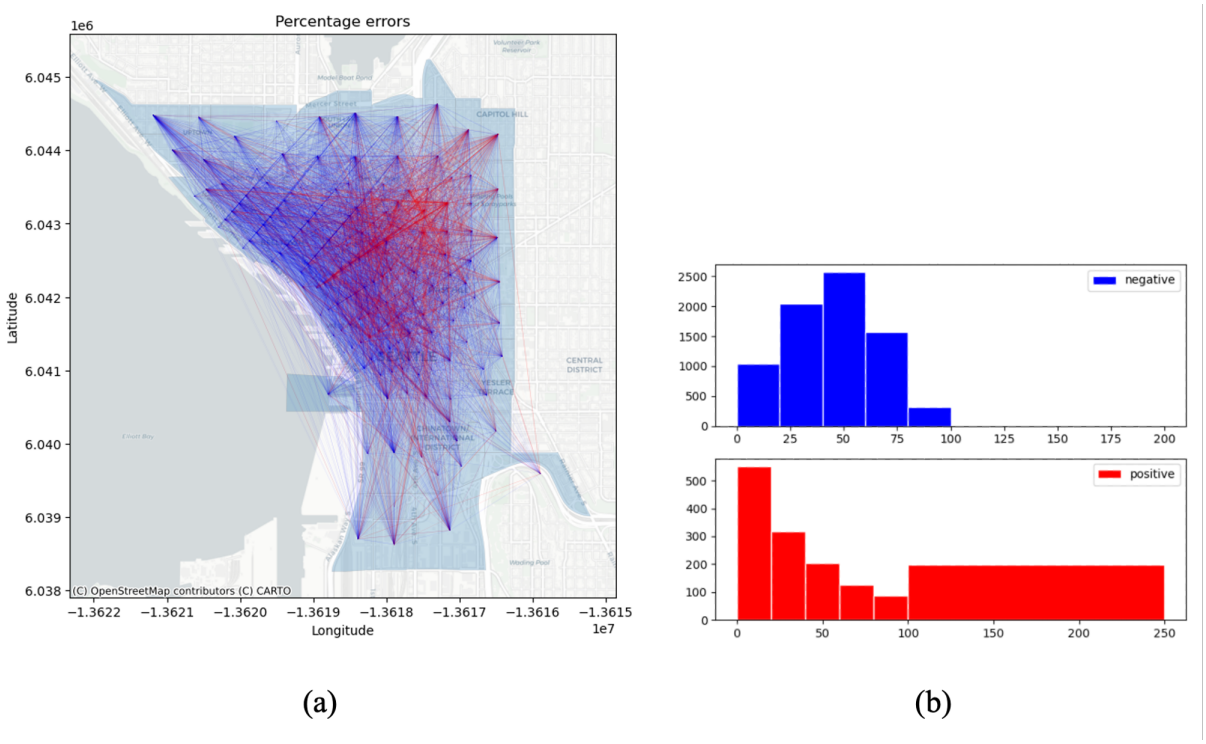


Figure 2.20: *Distribution of traveltime errors among OD pairs after calibration*

and conflicts among network users. Therefore, for future improvements of this simulation, pedestrian demand and connectivity of sidewalks should be taken into account.

Table 2.5 shows the difference between the old simulation (our starting point in this project) and the final simulation (after demand regeneration, simulation reconfiguration, and calibration), summarizing all the changes made to improve this simulation. Figure 2.21 illustrates the network performance and distribution of lane densities in the old versus final simulation between 8 to 9 AM in the simulation, showing a more similar congestion pattern to traffic patterns observed on Google Maps. Also, figure 2.22 compares empty edges between the new simulation and the old one within the same hour. Here is how the new demand format is helping our simulation, whereas in the previously used TAZ to TAZ demand format, some inner edges of TAZs could not be the origin or destination of any trips. Both figures show the lower importance of minor streets in Downtown Seattle network in the

Table 2.5: *A summary and comparison between the new simulation and old simulation*

	Old simulation	New Simulation
# trips for 5-10 AM	158k	207k
Attention to departure time	No	Yes
Pseudo TAZ assignment	Less reliable	More reliable
Including trips w/o sumo-TAZs as O/D	No	To some extent
Demand format	TAZ-TAZ	Link/TAZ-link/TAZ
Routing	highly unrealistic	more realistic
Use of alleys	Frequent	Eliminated
Simulation end time	~15 PM	~11 AM

new simulation, indicating an improvement in vehicles' route choice.

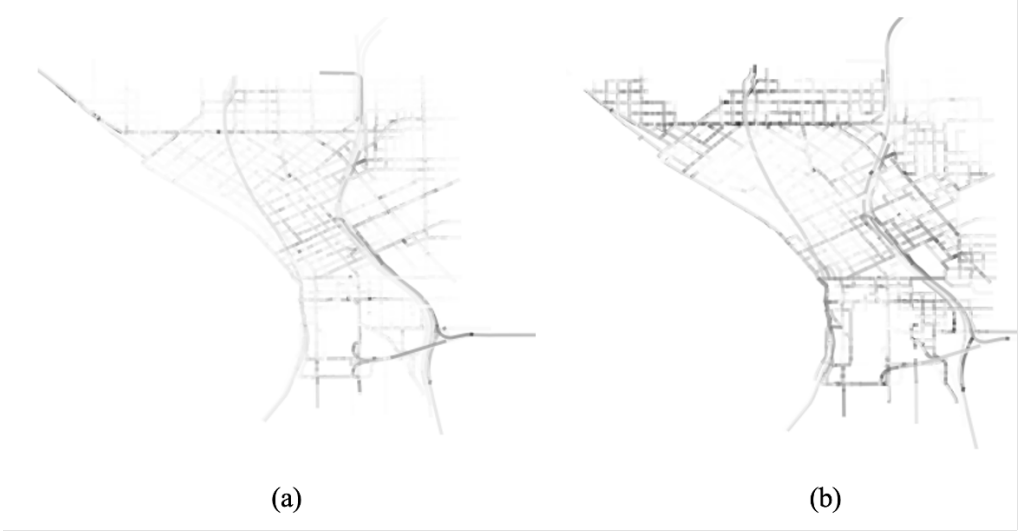


Figure 2.21: *Normalized lane density in veh/km in a) final simulation, and b) old simulation*

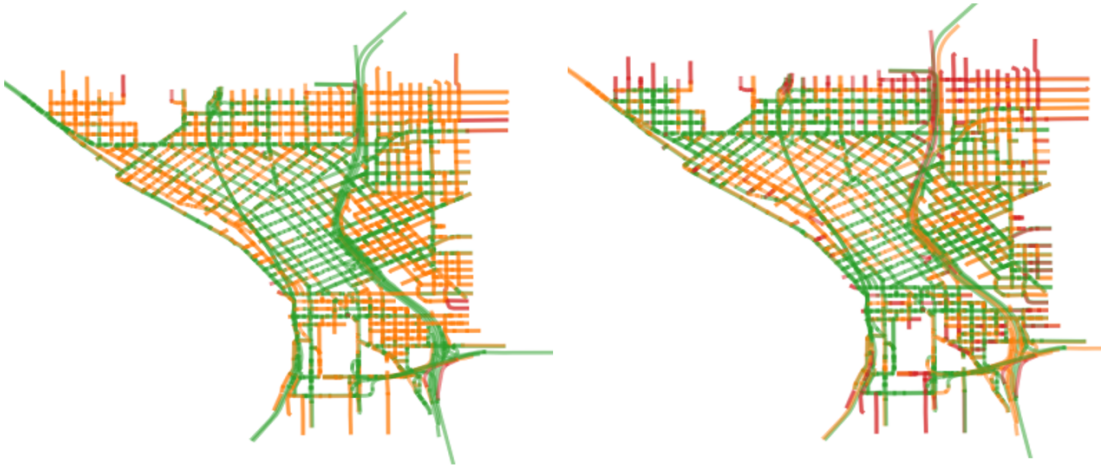


Figure 2.22: *Edges with distribution of zero (red), less than five veh/km (orange) and more (green) in a) final simulation, and b) old simulation*

Chapter 3

NETWORK PARTITIONING

In this chapter, we try to use the simulation results of the previous chapter to create a set of regions with well-defined macroscopic fundamental diagrams (MFD) within Downtown Seattle's network. The regions need to be homogeneous regarding the traffic measure chosen for segmentation and have compact shapes to ensure that vehicles will not re-enter a region after they leave one during their trip. Additionally, small clusters are undesirable due to high statistical errors in MFD and difficulties in designing perimeter control strategies [23]. Also, boundary queues resulting from perimeter control strategies will significantly affect the MFDs of small regions, which are not bigger than a few blocks, and that is contrary to assumptions of the perimeter control strategy used in the next chapter. In the end, the physical properties along with the simulation traffic data for each region will serve as outputs of this chapter.

3.1 Methodology

Region partitioning is implemented mainly based on the method proposed by Ji and Geroliminis in [23]. The authors used a 2.5 square mile area of Downtown San Francisco and corresponding link densities to test their partitioning algorithm. They aimed to partition the network into segments considering three criteria to guarantee well-defined MFDs and further facilitate the design and implementation of control strategies. These criteria include: (1) having a small variance of density values within each region, (2) extracting a small number of regions and ignoring details and local features, and (3) having spatially near compact shapes of clusters. They developed the region partitioning method with three consecutive algorithms: (1) initial segmentation, (2) merging, and (3) boundary adjustment. Next sub-

sections will elaborate on the partitioning method in this work that was built on the original method by [23]

3.1.1 Initial segmentation

In initial segmentation, the Normalized Cut algorithm (NCut) is used multiple times to divide a selected region into two smaller regions each time until we obtain several more clusters than desired. The NCut algorithm was first introduced as a method for image segmentation [33]. To use this algorithm, we turn our simulation network into a graph by modeling each street as a node and defining an adjacency matrix A where $A(i,j) = 1$ for each pair of links i and j if they are connected, otherwise $A(i,j) = 0$. With the help of the adjacency matrix, a similarity matrix $\mathbf{W}$ is then constructed as:

$$\mathbf{W}_{ij} = w(i,j) = \begin{cases} exp(\frac{-(d_i - d_j)^2}{\sigma_d^2}) & \text{if } A_{ij} = 1 \\ 0 & \text{if } A_{ij} = 0. \end{cases} \tag{3.1}$$

where d stands for the set of densities and σ_d is the standard deviation of the densities. The use of σ_d^2 to normalize the similarity can help produce more stable results, which was not used in the original method.

Assuming the NCut algorithm partitions a graph into two subnetworks of A and B, the similarity between these two regions is defined as $cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$ or the total weight of removed edges by the cut. The algorithm works based on finding the cut in the graphs that results in the maximum normalized association ($Nassoc$) and the minimum disassociation ($Ncut$) as defined below:

$$Ncut(A,B) = \frac{cut(A,B}{cut(A,V)} + \frac{cut(B,A)}{cut(B,V)} \tag{3.2}$$

$$Nassoc(A,B) = \frac{cut(A,A}{cut(A,V)} + \frac{cut(B,B)}{cut(B,V)} \tag{3.3}$$

The sum of these two values is 2, therefore, minimizing $Ncut$ value, which is an NP-

complete, is enough. Here, the region partitioning can be done based on [33] and by solving the generalized eigenvalue system below:

$$(\mathbf{D} - \mathbf{W})\mathbf{y} = \lambda \mathbf{D}\mathbf{y} \tag{3.4}$$

where $\mathbf{W}$ is the symmetrical similarity matrix, and $\mathbf{D}$ is a $N \times N$ diagonal matrix with vector $\mathbf{d}$ on its diagonal where $\mathbf{d}(i) = \sum_j w(i,j)$. Based on [33], the eigenvector with the second smallest eigenvalue is the solution to the normalized cut problem which needs to be discretized to bipartition the graph. In this work, K-means algorithm was used to find two segments out of the parent region.

We need to run the NCut algorithm iteratively until there are several more regions than desired. For each iteration, a parent region should be chosen to be partitioned into two smaller ones. In the original method, the subgraphs resulting from the previous step should be bipartitioned in the next iteration. In this work, however, a parent region is chosen based on the highest value of P which is calculated as below:

$$P_i = \alpha.Var_i^d + (1-\alpha).N_i \tag{3.5}$$

where Var_i^d and N_i are the variance of densities and the number of links in region i, respectively. α is a nonnegative and smaller than one tuning parameter that determines the weights in the linear combination of variance and number of edges. The value of α was set to 0.5 during our tests. The method used in this work helps produce better partitioning results by penalizing the regions with the highest variance in their densities and avoiding major differences between the sizes of regions during the initial segmentation.

3.1.2 Merging

When a series of initial clusters are formed, a merging algorithm can be run in order to reduce the effect of unnecessary cuts in the initial segmentation algorithm. Each time the merging

algorithm runs, we need to find two neighbor regions with the least difference between their mean densities to be combined and form a single region.

To find two neighboring regions to be merged, we also need to consider the number of boundary links so that the new region is more likely to form a compact shape. In the original method, this was not considered, leading to a lack of compactness in our network when merging neighbors with a few boundary links. In this work, we define B_{min} as the minimum number of boundaries required between two adjacent regions to consider them as neighbors in the merging algorithm or the calculation of partitioning metrics. Our tests showed that a value of $B_{min} = 4$ works reasonably for our simulation network.

3.1.3 Partitioning metric and MFD calibration

In this work, a primary metric, average NcutSillhutte for k clusters (NS_k) [33], was used to evaluate the effectiveness of each iteration of initial segmentation and merging algorithms and to help us find the optimal number of clusters. The NS metric for two neighbor clusters of A and B measures the average quadratic distance between values of clusters A and B and follows the formula below:

$$NS_k(A,B) = \frac{\sum_{i\in A}\sum_{j\in B}(d_i - d_j)^2}{N_A N_B} = Var(A) + Var(B) + (\mu_A - \mu_B)^2, \tag{3.6}$$

where Var(A) and Var(B) are the variances of the densities in clusters A and B, and μ_A and μ_B are the mean densities of these clusters.

To evaluate that if a cluster A is properly partitioned, we use the NS value of this cluster as follows:

$$NS_k(A) = \frac{NS_k(A,A)}{min\{NS_k(A,K)|K \in Neighbor(A)\}}, \tag{3.7}$$

where $Neighbor(A)$ denotes the set of neighbors of cluster A. This way, we can compare the inter-cluster versus the intra-cluster similarity between cluster A and its most similar

neighbor. Therefore we can evaluate overall segmentation by observing the average NS as:

$$NS_k = \frac{\sum_{A \in C} NS_k(A)}{k}. \tag{3.8}$$

A lower NS value indicates more homogeneous and well-partitioned clusters. So in our work, the goal is to reach the best segmentation by reducing this metric as much as possible while maintaining a good number of practical and compact regions.

Another metric used during partitioning was the normalized total variance defined in [31] as:

$$TV_N = \frac{\sum_{i=1}^{N_c} N_i \times Var(C_i)}{N_c \times Var(C)}, \tag{3.9}$$

where N_c and N_i refer to the total number of regions in the network C and the number of links in cluster C_i, respectively.

One way to maintain good segmentation is to check if regional MFDs are well-defined with low-scattered points and good shape. An MFD captures these dynamics by showing flow characteristics versus utilization of an area. To fit an MFD for each region, we use characteristics of each edge for each minute, such as space-mean speeds and the number of vehicles. In this work, we aim to estimate the productivity of region i ($P_i(*)$), hourly vehicle-km traveled, based on:

$$P_i(t) = n_i(t)V_i(t), \tag{3.10}$$

where $n_i(t)$ is the number of vehicles in a region in time t, and $V_i(t)$ is the space mean speed of a region in km/hr which can be generated from average speeds of edges in a region weighted by the number of vehicles in a given minute. Finally, the MFD functions of regions $P_i(t)$ will be approximated by a third-order function of $n_i(t)$. Good MFDs should illustrate a low scatter of productivity estimates from formula 3.10 versus the number of vehicles in a region, and the fitted line should be well-shaped and approach zero in high vehicle numbers.

3.1.4 Finalizing the segmentation

The last step of region partitioning is boundary adjustment, which is a means to further satisfy the compactness criterion by refining the edges of a rough sketch. The boundary adjustment runs on the optimal form of segmentation and merging resulting from previous steps. In the main method, an algorithm adjusts the boundaries among clusters to minimize the variance of link densities within each cluster while maintaining smooth shapes.

In this work, however, we do not have an algorithm. Instead, we manually alter the region labels for boundaries of regions, if needed. Although this will not result in the least variance for regions, this way, we can keep regions practical by ensuring that boundaries are on critical intersections of the network and a control strategy can be applied using traffic signals. We can also increase the compactness of a region using our own judgment.

3.2 Results

The input data for the proposed work is the calibrated Downtown Seattle network simulation done in the previous chapter. The network used in the simulation (Figure 2.1) consists of various types of roads, including minor streets (alleys or parking entrances) some of which were not removed in section 2.2. The network used in the algorithm excluded minor streets as they could act as outliers in our analysis and adversely affect the homogeneity of resulting regions. For partitioning, edge-based data was used, consisting of five consecutive 60-min intervals (aggregation period of 60 minutes) from 5 to 10 AM [8]. To perform region partitioning, we chose the 8-9 AM time interval as the peak hour of the Seattle Downtown network and modeled the SUMO network with adjacency matrix-based graph representation. The algorithms in this chapter were implemented in Python using packages such as SciPy, scikit-learn, Matplotlib, NumPy, and Pandas.

To obtain MFDs, we used minute-by-minute edge-based data (aggregation period of 60 seconds) from 5-10 AM of the simulation, 300 data points for each MFD, to better capture the traffic dynamics of each region. Despite ending after 10 AM, we did not consider the

data after that time since the network dynamic would change dramatically after 10 AM as a result of the network getting cleared. In this work, the calibrated simulation could give us the data for the uncongested side of fundamental diagrams. However, it did not allow us to capture productivity for a full range of vehicle accumulation from zero to near-jam traffic. To obtain well-shaped MFDs and fit better lines, we used another simulation where the demand was highly increased in a way to ensure congestion in most of the network. This helped us have data for both congested and uncongested conditions of our network. Note that an MFD is a property of a traffic network and is demand-independent. Our tests also showed that MFDs from the congested simulation were indeed completing our scatter plots from the calibrated simulation and there were no significant differences. Therefore, partitioning the network based on one simulation and checking MFDs based on another simulation is acceptable.

Before segmentation, we first divided the network into multiple pre-defined regions. This was done based on our understanding of the city and due to differences between land use geometry, and roads in these regions. Figure 3.1 shows how this manual pre-definition was done. The south region includes a residential neighborhood (Chinatown) and an industrial district. The east side is a mixture of leisure activities and residential uses, having local streets in most parts, and separated from the west part by I5 highway. North and northwest regions, especially in 2018 when our simulation demand is for, had more residential use and low-rise buildings than the central region. The central region, the host of many businesses and high-rise buildings, is the core of Downtown Seattle and is the focus of our segmentation. In addition to these, we also had highways (I5, I90, and SR-99) as a separate region for their different rules and dynamics from an urban network containing roads with lower classifications. Moreover, the existence of a well-defined MFD for freeway systems has been questioned in the literature. Yet, for the implementation of the perimeter control strategies, we need to include highway segments as one region since the flow between highways and downtown clusters is not negligible. Partitioning of the network in Figure 3.1 was done in four stages: I) segmentation of the central region, II) segmentation of other regions, III)

merging side and central subnetworks if needed, and IV) adjusting boundaries.

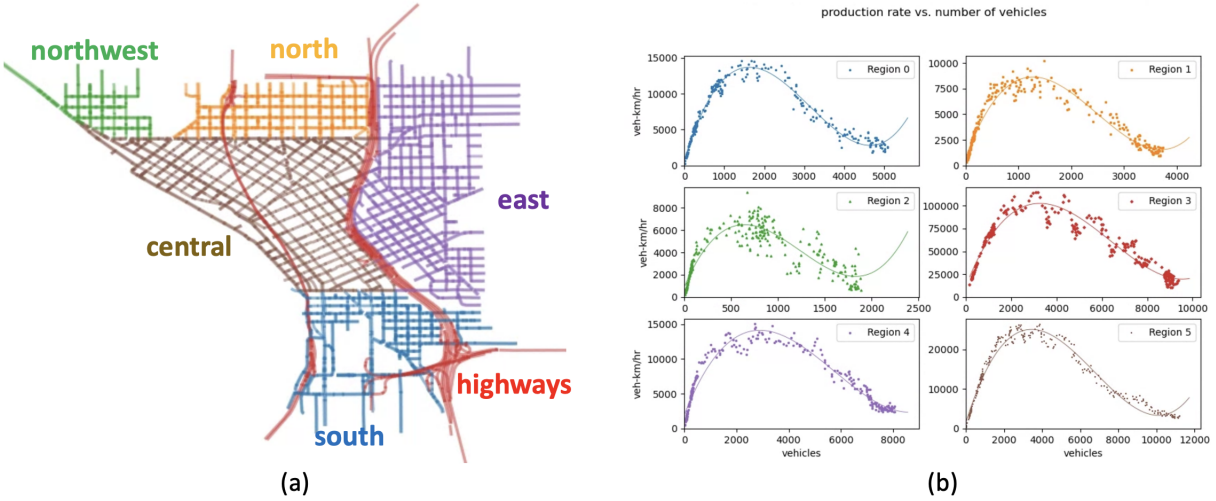


Figure 3.1: *Initial setting of the network before segmentation with (a) pre-defined segments and (b) their MFDs*

Figure 3.2 shows different stages of the network during the segmentation process starting from a density map resulting from the calibrated simulation between 8-9 AM. In figure 3.2(a), darker segments mean higher density than lighter edges. Note that in SUMO, there are alleys and parking entrances between intersections, so the network file needed multiple segments between each two intersections to create little junctions. Although we removed most of the minor roads in chapter 2, junctions still exist in the network file. In other words, a segment of the road between two intersections is undesirably modeled with more than one segment in our graph. When having traffic lights, one of these segments could have queues and be jammed, while upstream segments have flow and lower densities. For this reason, there are a few dark points in the density map that indicate constantly having queues during the 8-9 AM period.

Before we started, the NS_k value of the pre-segmented network was 1.07 with TV_N of 0.958, so the final partitioning needed lower values for the whole network. To start, the

initial segmentation algorithm was applied 14 times in the central region which resulted in figure 3.2(b) with the central region divided into 10 regions. Note that after applying the initial segmentation 14 times, we manually merged 4 super small segments, caused by the dark points mentioned, with the bigger region they were in. Then, the merge algorithm was applied seven times to get figure 3.2(c). Since the pink region was too big and not compact, we used one more time of Ncut algorithm on it to get figure 3.2(d). We then focused on side regions, except the freeway region which was untouched in all steps. It started with 10 initial cuts and manually merging two super small regions to get figure 3.2(e), and then four times of automatic merge and one final manual merge, which gave better metrics than another automatic merge, to generate figure 3.2(f). For the next step, there was one side region and one central region with close mean densities and they were chosen for the next merge. Merging other side regions and central regions could not improve the metrics and did not seem practical. as explained in section 3.1.4, manual boundary adjustment was applied and final segmentation in figure 3.2(h) was obtained.

In the end, Table 3.1 shows partitioning statistics for the initial network with pre-defined regions and the steps shown in figure 3.2, showing smaller values in the final segmentation than the initial one. Additionally, figure 3.3 shows MFDs for each of the regions in the final segmentation, all of which are relatively well-defined. One way to further improve the MFDs is to merge the scattered ones with their neighbors. In this work, MFDs for regions 9, 10, and 11 are more scattered than other MFDs. Merging these regions with their neighbors would have resulted in better-defined MFDs. In that case, however, we could have a high discrepancy in the size of regions and the NS_k value would get near to the original value. Therefore, we did not further change the segmentation.

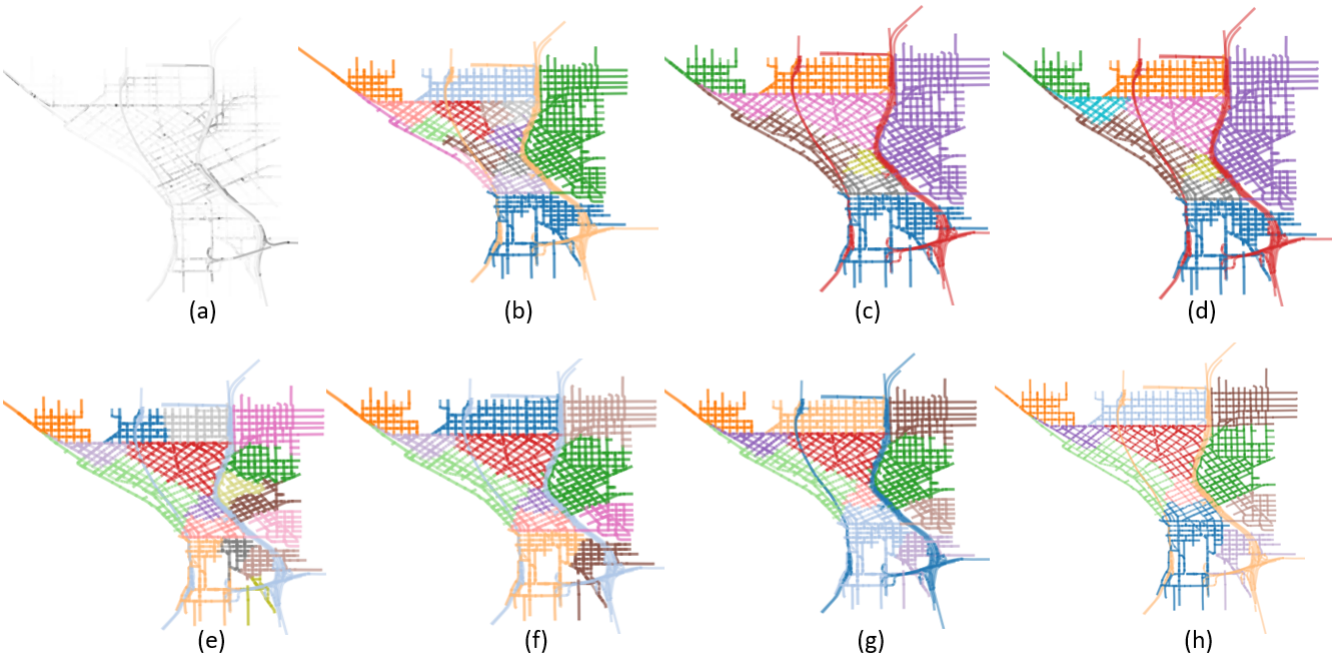


Figure 3.2: *Donwtown Seattle network (a) relative lane density values (darker means higher density) and (b-h) different steps during the partitioning*

Table 3.1: *Partitioning metrics during the segmentation*

Segmentation step:	Pre-defined	b	c	d	e	f	g	h
Average NS	1.070	1.160	0.98	1.03	1.070	1.00	0.99	1.020
Normalized TV	0.958	0.936	0.94	0.94	0.925	0.93	0.93	0.931

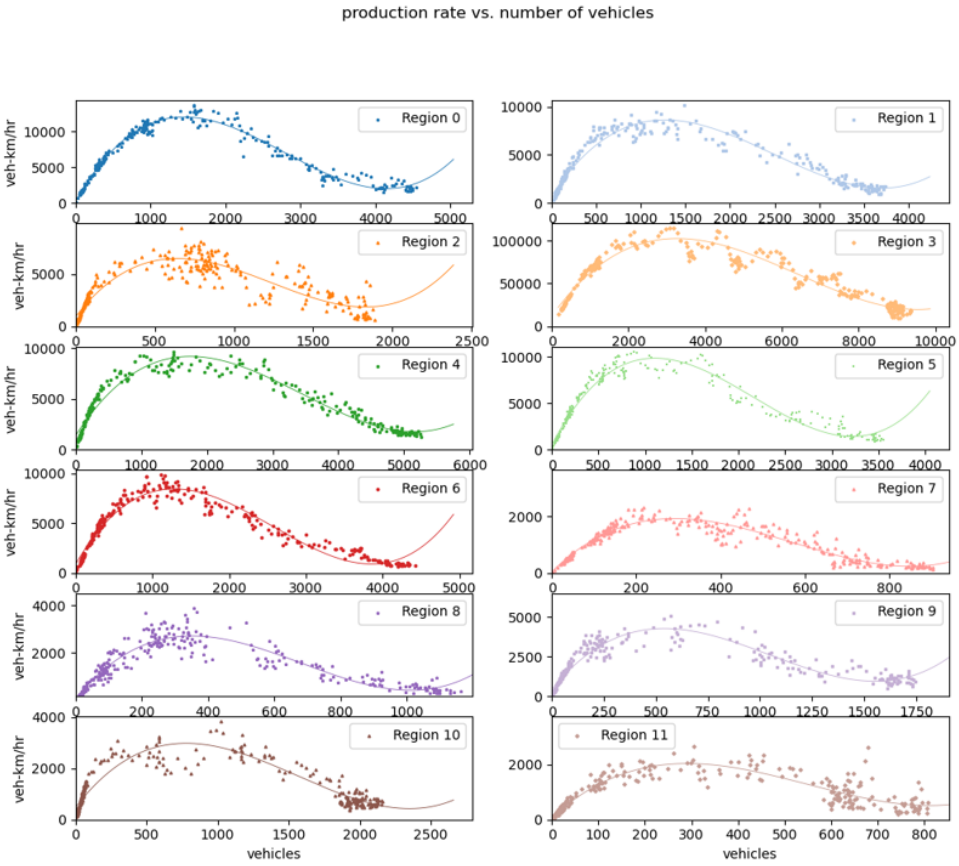


Figure 3.3: *MFDs of the final segmentation in Figure 3.2(h)*

Chapter 4
MFD-BASED PERIMETER CONTROL

In previous chapters, Downtown Seattle traffic network was divided into homogeneous regions, each described by an MFD, using two simulations, a calibrated and a highly congested one. The final step involves using the partitioned regions for implementing the perimeter control method, building on recent work in the iUTS Lab. We will apply the framework proposed in [20], which is an MFD-based perimeter control method that captures travelers' route choice and the queue capacities at the boundaries of the partitioned regions. Queue dynamics at boundaries are handled by buffer zones between each two adjacent regions with infinite vehicle storage, and vehicles in buffer zones are excluded when accumulating vehicles in each region for evaluating congestion. An efficient-range method to implement perimeter control at region boundaries. Meanwhile, travelers' route choices are modeled based on the IDUE principle as formulated by [3].

The perimeter control system and the travelers are considered as two players, in which the perimeter control system aims to improve overall network performance by controlling inflows and outflows at regional boundaries, while travelers try to minimize their instantaneous travel times. In this regard, [20] proposes a non-zero sum, non-cooperative differential game framework where the system manager (perimeter control system) and travelers interact dynamically.

4.1 Methodology

The framework by [20] was built on three main assumptions listed below:

- The urban transportation network is divided into homogeneous regions with well-defined MFDs, where the average trip length within each region is assumed to be

constant ($l_i(t) = li$), and the demands between all region pairs are known.

- The control system has access to macroscopic information of the network, such as the number of vehicles in a region at any time. Additionally, travelers have access to real-time information and choose the routes (a sequence of regions to traverse from an origin to a destination) that minimize their instantaneous travel times once they enter a region.

- A buffer zones between each two adjacent regions, containing only one one-lane signal-controlled entrance between each two region (Figure 4.1). Buffer zones are assumed to have enough space for vehicle queues, modeled as point queues, and do not affect within-region dynamics.

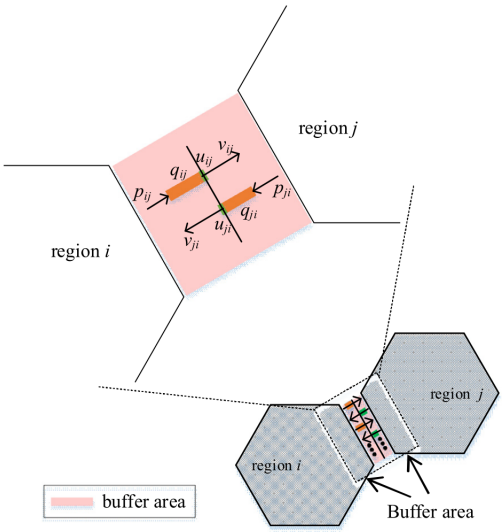


Figure 4.1: *Illustration of the point queue at the boundary of two regions [20].*

The framework is designed using four major components: the point queue model, the MFD dynamics, the IDUE route choice behavior, and the efficient-range perimeter control. Figure 4.2 shows how this framework performs, where there are multiple homogeneous regions with the travelers and the system manager as two players, trying to minimize travel times and

regulating boundary flow rates respectively. The players are involved in a non-cooperative game and connected by traffic dynamic variables at both regional and queue-wide levels. To reflect real-world conditions, the original work uses a one-second re-calculation period for the network dynamics such as route choices, and a 60-second one for the perimeter control which is the cycle time of many traffic lights in practice. The rest of this section briefly explains each of the aforementioned components. The reader can see [20] for a detailed explanation.

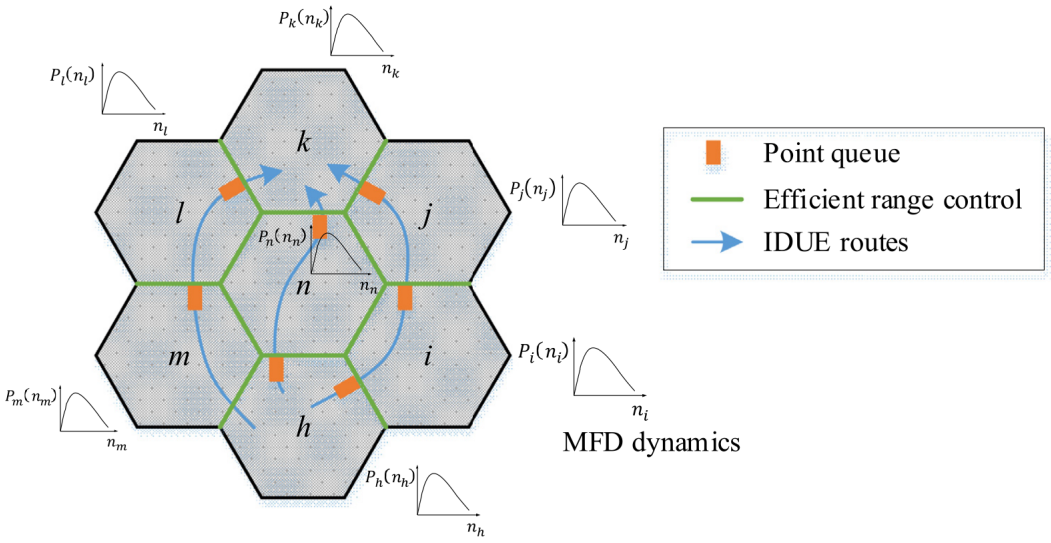


Figure 4.2: *Model overview (for traffic from region h to region k). [20].*

1. **Point Queue Model:** The point queue model is used to describe the dynamics of vehicle queues at regional boundaries (buffer zones) [3]. The dynamics between regions i and j are defined as:

$$\dot{q}_{i,j}(t) = \begin{cases} 0 & \text{if } q_{i,j}(t) = 0 \text{ and } p_{i,j}(t) - u_{i,j}(t) < 0 \\ p_{i,j}(t) - u_{i,j}(t) & \text{otherwise} \end{cases}, \tag{4.1}$$

where $q_{i,j}(t)$ is the queue length at boundary (i, j) at time t, $p_{i,j}(t)$ is the inflow rate from region i to the boundary (i, j), and $u_{i,j}(t)$ is the controlled discharge rate from

the queue. For exit flow and the travel time to pass the boundary, we have:

$$v_{i,j}(t) = p_{i,j}(t) - \dot{q}_{i,j}(t), \tag{4.2}$$

$$\tau^q_{i,j}(t) = \frac{q_{i,j}(t)}{u_{i,j}(t)}, \tag{4.3}$$

where at boundary (i, j) at time t, $v_{i,j}(t)$ and $\tau^q_{i,j}(t)$ show the actual exit flow rate and the travel time to pass the queue.

2. **MFD dynamics:** The traffic dynamics within each region are modeled using the MFD, which provides a relationship between the regional vehicle density and space mean flow. For a homogeneous region i at time t we have:

$$V_i(t) = \frac{P_i(n_i(t))}{n_i(t)}, \quad c_i(t) = \frac{P_i(n_i(t))}{l_i}, \tag{4.4}$$

where $V_i(t)$ is the average speed in region i at time t, $n_i(t)$ is the number of vehicles in region i at time t, $c_i(t)$ is the trip completion rate in region i at time t, l_i is the average trip length in region i, and $P_i(n_i(t))$ is the MFD function, approximated by a third-order function, as in figure 3.3. Using these basic formulas, some of other relationships regarding regional dynamics include:

- Trip completion rate for vehicles with destination d in region i at time t, given $n^d_i(t)$ as the number of vehicles with the same specification:

$$c^d_i(t) = \frac{n^d_i(t)}{n_i(t)} c_i(t) \tag{4.5}$$

- Travel time in region i:

$$\tau^r_i(t) = \frac{n_i(t)}{c_i(t)} \tag{4.6}$$

- And using flow conservation, we can show vehicle dynamics as:

$$\dot{n}_i^d(t) = \sum_{k:(k,i)\in L_d} v_{k,i}^d(t) + D_i^d(t) - \sum_{j:(i,j)\in L_d} p_{i,j}^d(t), \tag{4.7}$$

where $\dot{n}_i^d(t)$ is the rate of change of vehicles with destination d in region i at time t, $v_{k,i}^d(t)$ is the exit flow rate from neighbor regions k to region i at time t, and $D_i^d(t)$ is the travel demand in region i for destination d.

3. **Instantaneous Dynamic User Equilibrium (IDUE):** Travelers' route choice behavior is modeled using the IDUE principle, which assumes that during a trip, travelers select routes based on current traffic conditions to minimize their instantaneous travel times. This principle was formulated in [4] using complementarity conditions:

$$0 \leq \theta_{i,j}^d(t) \perp \left(\tau_{i,j}(t) + \eta_j^d(t) - \eta_i^d(t)\right) \geq 0, \tag{4.8}$$

where $\theta_{i,j}^d(t)$ is the flow assigned to route (i,j) for destination d, $\tau_{i,j}(t)$ is the travel time from region i to its neighbor j $(= \tau_i^r + \tau_{i,j}^q)$, and $\eta_i^d(t)$ is the minimum instantaneous travel time from region i to destination d .

Equation 4.8 suggests that a traveler in region i would select neighbor j as the next region (i.e., $\theta_{i,j}^d(t) > 0$) if and only if region j is on the route with the minimum instantaneous travel time from region i to destination d (i.e., $\tau_{i,j}(t)+\eta_j^d(t)-\eta_i^d(t) = 0$). Otherwise, we will have $\tau_{i,j}(t) + \eta_j^d(t) - \eta_i^d(t) > 0$ and $\theta_{i,j}^d(t) = 0$. Note that as a traffic assignment principle, flow conservation was also considered in the main paper in the form of another complementarity condition which is not depicted here.

4. **Efficient-Range Perimeter Control:** This control method aims to help the system manager maintain vehicle accumulation within an efficient range by regulating the inflow rates at the region boundaries. The control variable $u_{i,j}(t)$ is adjusted based on

the vehicle accumulation $n_j(t)$ in the downstream region j as below:

$$u_{i,j}(t) = \begin{cases} u^*_{i,j} + \epsilon, & \text{if } n_j(t) \leq n^*_j, \\ 0 + \epsilon, & \text{otherwise,} \end{cases} \tag{4.9}$$

where $u^*_{i,j}$ is the maximum allowed discharge rate at boundary (i, j) and n^*_j is the efficient range threshold. In the main paper, all MFD functions were the same and a threshold of 4000 vehicles was used for all regions. In our work, however, MFDs are different from each other, making maximum capacities different from each other. We used the optimal point, where the MFD is in maximum, of fit lines to set the maximum capacity of regions as seen in table 4.1. Moreover, the original framework added a small value of $\epsilon(= 6veh/min)$ to Eq.4.9 to avoid zero division errors. Lastly, the paper transforms the equation above into a complementarity condition in order to solve it as an optimization problem.

These components are integrated into a Differential Complementarity System (DCS) framework, allowing for formal analysis and solving of the combined traffic control problem. The DSC formulation approach treats the problem as a non-zero sum, non-cooperative differential game involving two players: the system manager and travelers. The system manager aims to optimize overall traffic performance by controlling inflow rates at region boundaries, while travelers choose routes to minimize their travel times based on real-time traffic conditions. The model combines the dynamics of traffic flow, described by the Macroscopic Fundamental Diagram (MFD) and point queue models, with complementarity conditions from Instantaneous Dynamic User Equilibrium (IDUE) route choice and perimeter control methods. These conditions ensure solution existence and convergence. The reader can refer to the main paper in [20] for a detailed solution analysis using a time-stepping approach.

4.2 Resutls

The explained perimeter control framework was implemented in the iUTS lab using MATLAB and GAMS. The framework originally used in the main paper directly used MFDs and average trip length in each region, which was assumed to be constant, to generate completion rates as formulated in Eq. 4.4. Our tests therefore started with the same framework and used the MFDs from the previous chapter. Figure 4.3 shows the partitioning and MFDs used in initial tests before finalizing the segmentation in chapter 3, with an average NS of 1.03 and TV_N of 0.944 and without boundary adjustment. For each region, the average length of trips was collected from the sum of the length of edges each trip traveled while being in the region, weighted by the time each trip spent in the region.

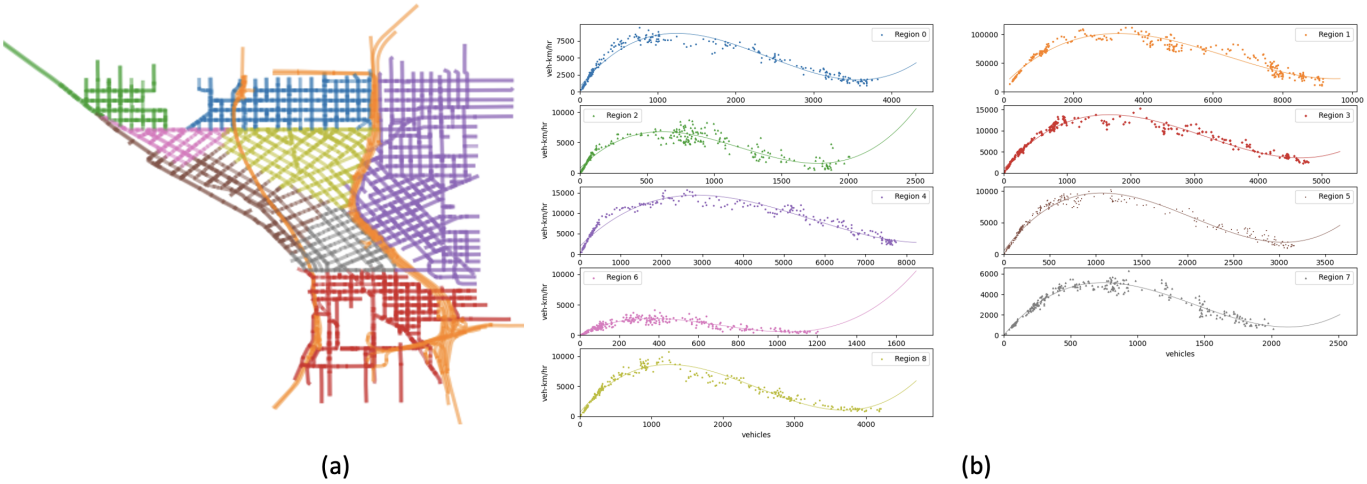


Figure 4.3: *The partitioning used for initial tests and regions' MFDs*

However, in the initial tests, vehicle accumulations were not as expected, similar to figure 4.5(a), because of very short regional travel times, resulted from very high completion rates (Eq. 4.6). This made us investigate more into the completion rates and create them without using Eq. 4.4, and directly from the simulation observations. Based on the definition, the completion rate of a region shows the rate of trips exiting the region or being finished in

the region given the total number of vehicles in the region ($n_i(t)$). Using the information of departure time of trips in the simulation, the time they were on each edge, and the region each edge is in, the completion rates scatter plots were generated and a third-order line was fit to them. Figure 4.4 shows the difference between calculated completion rates and empirical rates. Compared to the calculated graph, the completion rates in figure 4.4(b) seem to be almost one-third of the ones in figure 4.4(a). Consequently, a new test with three times higher average lengths for each region($l_i' = 3l_i$) was conducted. Figure 4.5(b) shows that in the case of applying perimeter control on a calibrated simulation and for the test segmentation, the number of cars in region 2 (freeways) will not exceed the threshold set (3283 from table 4.1), number of vehicles in maximum MFD function in figure 4.3(b). Note that the fluctuations are caused by the 60-second period between changing control parameters in boundaries.

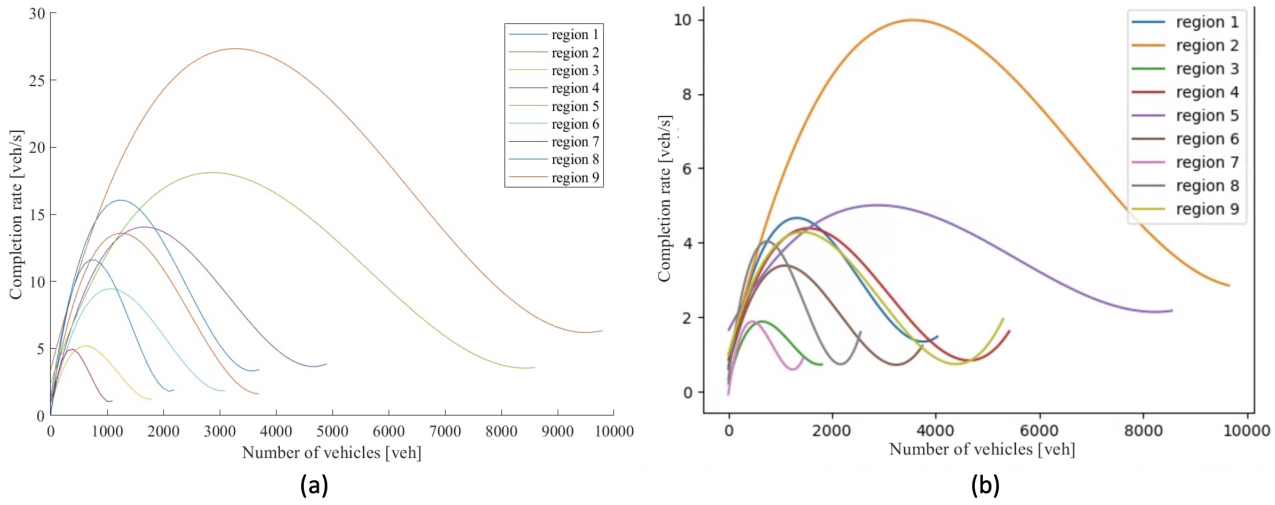


Figure 4.4: *Test network completion rates from (a) equation 4.4, and (b) simulation observations.*

After finalizing the segmentation in chapter 3, adjustments were made to the framework in order to directly use empirical completion rates, which appeared to be different than expectations, instead of MFDs and set the maximum capacity of regions based on them. Figure 4.6 shows the main network segmentation intended to be used in the perimeter control

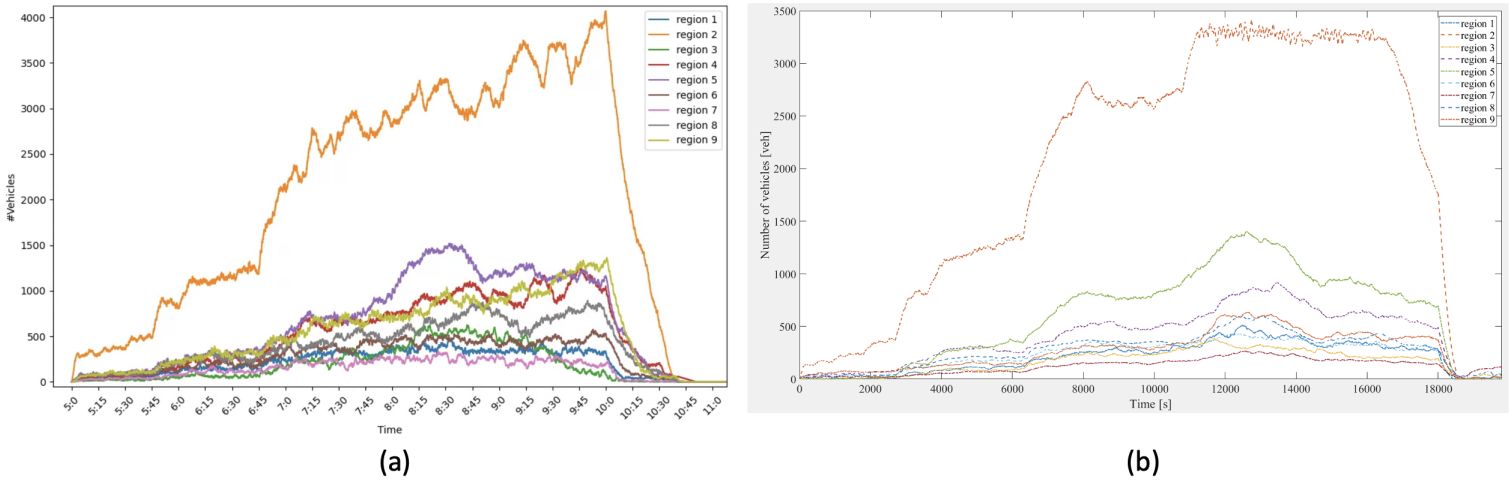


Figure 4.5: *Accumulation of vehicles in regions during the calibrated simulation and without control(a) in SUMO simulation, and (b) in the perimeter control framework in MATLAB.*

Table 4.1: *Thresholds set for segments in the perimeter control*

Region#	1	2	3	4	5	6	7	8	9	10	11	12
Test segmentation	1242	3283	616	1664	2871	1066	370	744	1249	-	-	-
Main segmentation	1408	1323	656	3569	1766	1154	1457	326	436	514	785	305

framework and related completion rates. Unfortunately, figure 4.7(b) shows that directly using completion rates with the new segmentation method did not work as expected, where the number and shape of vehicle accumulations for regions are not as similar as it was in figure 4.5. Therefore, more investigation and debugging are still needed to make the framework compatible with our real-world replications in previous chapters.

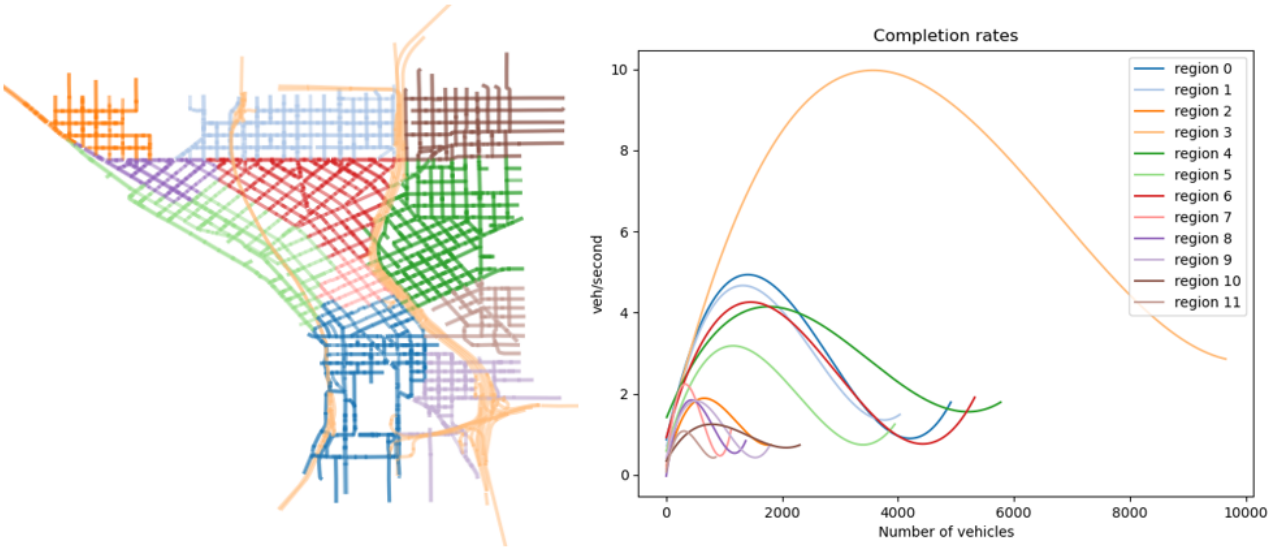


Figure 4.6: *Regions of the network and their completion rate estimations.*

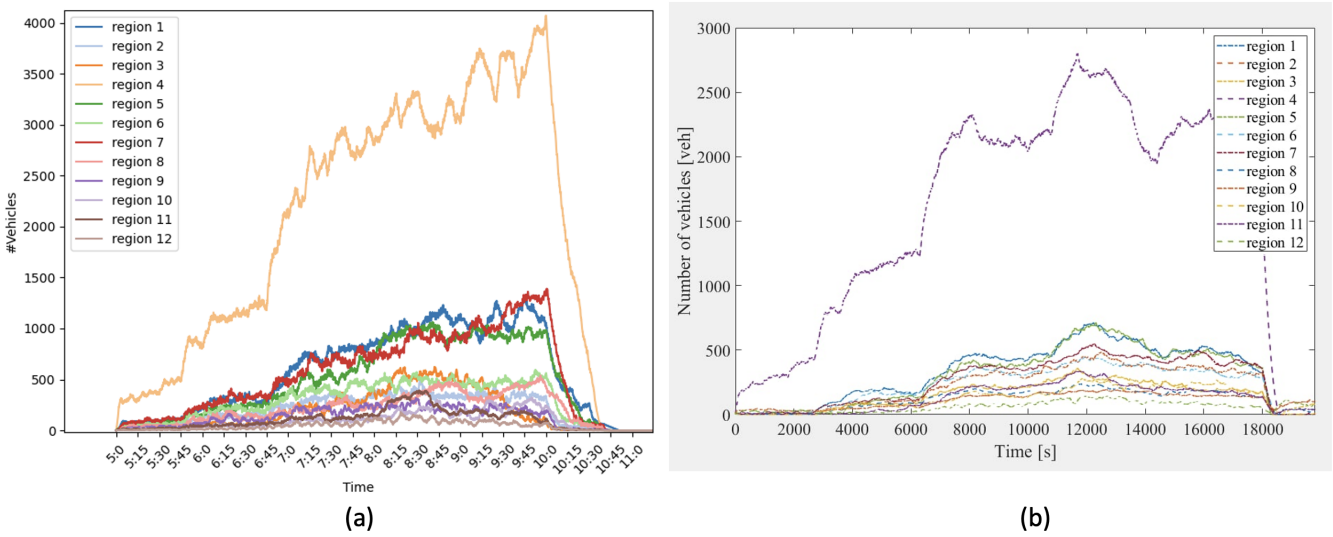


Figure 4.7: *Number of vehicles in regions in (a) SUMO and (b) MATLAB simulations.*

Chapter 5

SUMMARY AND CONCLUSION

The main objective of this thesis was to apply the point-queue IDUE- and MFD-based perimeter control proposed by Guo and Ban [20] in a real-world setting. After a literature review, we put details of our work on a Downtown Seattle morning peak simulation on SUMO software. Chapter 2 explains how the demand generation process, network geometry, and simulation configurations were improved and calibrated to some extent. In chapter 3, a previous algorithm by [23], which included the Normalized-Cut-based segmentation algorithm, merging, and boundary adjustment, was modified to divide the traffic network into several regions. This resulted in final segmentation with a practical number of regions, low variance on densities, compact shapes, and well-defined macroscopic fundamental diagrams (MFD). Finally, we tested the proposed MFD-based perimeter control method using a test segmented network and the result of chapter 3 as a main segmented network. Although initial runs of the perimeter control algorithm on the test segmentation sounded promising, the algorithm was unable to properly run for the main segmentation. This necessitates more work from the author to resolve compatibility issues between the framework initially proposed for a hypothetical network and the real-world replication attempted in Chapters 2 and 3.

Conducting this work presented multiple challenges. The simulation process included many little details that made the simulation differ from our expectations, making us spend a lot of time even though we had not planned to work on this section in the beginning. The main challenges of that part were coming up with a strategy to generate reliable demand based on source data, solving highly unrealistic routing decisions from travelers, and trying to calibrate a very big network. The main challenge of the second part, segmentation, was

the amount of coding needed to get the job done. Many tasks had to be done in that section including reading and processing multiple sources of simulation data, implementing the partitioning algorithm, visualization of multiple components, generating correct MFDs, generating proper inputs for the perimeter control framework such as maximum flow at boundaries, and making changes to the main algorithm due to differences between inputs of the main paper and the current study.

Lastly, in chapter 4, the main challenge has been debugging the framework in an unfamiliar environment, MATLAB. For example, we had to figure out the reasons behind low, and sometimes negative, vehicle accumulations in regions. As mentioned before, we recently discovered a difference between calculated and real completion rates, forcing us to make some changes and we still need more time and more tests to understand how we can successfully apply the perimeter control to our real-world augmentations and analyze it with different demand scenarios and capacity thresholds.

Each of the three parts of this project had its limitations that could be further improved. Regarding the simulation, first, Mercer St. which acts as a main arterial road that users use to enter the downtown region can be accurately added to the network file and TAZ files. Second, as the tables and figures in section 2.3 illustrate, more time and effort can be put into having a more effective calibration of Downtown Seattle network. Moreover, more work can be done in terms of network geometry, where many segments are split into tiny segments resulting in more variance among edges and super-small segments that need to be handled manually.

While a clean network geometry is highly crucial to an effective partitioning algorithm, some innovations can help us reduce the effect of very small links of OSM-extracted traffic networks. In our tests, smoothing density data before segmentation or using other traffic measures (e.g. edge occupancy) have shown promise. Additionally, the length of edges can be included in definition of the similarity matrix (Eq. 3.1). Other possible improvements in chapter 3 include fine-tuning α in Eq. 3.5 and experimenting with other discretizing methods for the solution of the generalized eigenvalue system in Eq. 3.4. Moreover, visual

judgment needs to be replaced with more exact and analytical definitions of well-defined and low-scattered MFDs.

And finally, in addition to issues regarding our tests, the perimeter control strategy could be improved in multiple ways. To start with, the control parameter in 4.9 cannot be zero or a small number as in the real-world, regions are divided by intersections, and not every intersection has a signal. Therefore, complete control over the flow between regions is simply not practical with current technologies and a realistic minimum flow is always present. Regarding the methodology, travelers' behavior includes multiple topics, such as route choice, mode choice, and departure time choice. The framework in [20], however, only includes IDUE route choice behavior while a more comprehensive control framework can also incorporate other behaviors. In addition, the boundaries between adjacent regions were modeled as a one-lane signal-controlled way, but adjustments can be made to consider multiple entrances, multiple lanes, and more complex driving behaviors between each two adjacent regions. Lastly, the infinite vehicle storage assumption can underestimate the impact of congestion spillover (due to lack of storage spaces) within buffer zones in heavily congested situations. Therefore, future work can be done to account for the capacity of buffer zones and their impact on the traffic dynamics of regions.

BIBLIOGRAPHY

[1] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2575–2582. IEEE, November 2018. URL https://elib.dlr.de/127994/.

[2] Mahyar Amirgholy, Mehrdad Shahabi, and H. Oliver Gao. Optimal design of sustainable transit systems in congested urban networks: A macroscopic approach. *Transportation Research Part E: Logistics and Transportation Review*, 103:261–285, 2017. ISSN 1366-5545. doi: https://doi.org/10.1016/j.tre.2017.03.006. URL https://www.sciencedirect.com/science/article/pii/S1366554516309061.

[3] Xuegang (Jeff) Ban, Jong-Shi Pang, Henry X. Liu, and Rui Ma. Continuous-time point-queue models in dynamic network loading. *Transportation Research Part B: Methodological*, 46(3):360–380, 2012. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2011.11.004. URL https://www.sciencedirect.com/science/article/pii/S0191261511001706.

[4] Xuegang (Jeff) Ban, Jong-Shi Pang, Henry X. Liu, and Rui Ma. Modeling and solving continuous-time instantaneous dynamic user equilibria: A differential complementarity systems approach. *Transportation Research Part B: Methodological*, 46(3):389–408, 2012. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2011.11.002. URL https://www.sciencedirect.com/science/article/pii/S0191261511001688.

[5] Christine Buisson and Cyril Ladier. Exploring the impact of homogeneity of traffic measurements on the existence of macroscopic fundamental diagrams. *Transportation Research Record*, 2124(1):127–136, 2009. doi: 10.3141/2124-12. URL https://doi.org/10.3141/2124-12.

[6] Carlos F. Daganzo, Vikash V. Gayah, and Eric J. Gonzales. The potential of parsimonious models for understanding large scale transportation systems and answering big picture questions. *EURO Journal on Transportation and Logistics*, 1(1):47–65, 2012. ISSN 2192-4376. doi: https://doi.org/10.1007/s13676-012-0003-z. URL https://www.sciencedirect.com/science/article/pii/S2192437620600085.

[7] Heng Ding, Fang Guo, Xiaoyan Zheng, and Weihua Zhang. Traffic guidance–perimeter control coupled method for the congestion in a macro network. *Transportation Research Part C: Emerging Technologies*, 81:300–316, 2017. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2017.06.010. URL https://www.sciencedirect.com/science/article/pii/S0968090X17301638.

[8] SUMO Documentation. Lane- or edge-based traffic measures, . https://sumo.dlr.de/docs/Simulation/Output/Lane-_or_Edge-based_Traffic_Measures.html.

[9] SUMO Documentation. od2trips, . https://sumo.dlr.de/docs/od2trips.html.

[10] SUMO Documentation. Importing o/d matrices, . https://sumo.dlr.de/docs/Demand/Importing_O/D_Matrices.html.

[11] SUMO Documentation. Calibrator, calibrating vehicle types, . https://sumo.dlr.de/docs/Simulation/Calibrator.html#type-dependent_mapping.

[12] SUMO Documentation. Definition of vehicles, vehicle types, and routes, . https://sumo.dlr.de/docs/Definition_of_Vehicles%2C_Vehicle_Types%2C_and_Routes.html#departlane.

[13] SUMO Documentation. Vehroutes, . https://sumo.dlr.de/docs/Simulation/Output/VehRoutes.html.

[14] Jakob Erdmann. Sumo's lane-changing model. In Michael Behrisch and Melanie Weber, editors, *Modeling Mobility with Open Data*, pages 105–123, Cham, 2015. Springer International Publishing. ISBN 978-3-319-15024-6.

[15] Nikolas Geroliminis and Carlos F. Daganzo. Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings. *Transportation Research Part B: Methodological*, 42(9):759–770, 2008. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2008.02.002. URL https://www.sciencedirect.com/science/article/pii/S0191261508000180.

[16] Nikolas Geroliminis and Jie Sun. Properties of a well-defined macroscopic fundamental diagram for urban traffic. *Transportation Research Part B: Methodological*, 45(3):605–617, 2011. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2010.11.004. URL https://www.sciencedirect.com/science/article/pii/S0191261510001372.

[17] Nikolas Geroliminis, Jack Haddad, and Mohsen Ramezani. Optimal perimeter control for two urban regions with macroscopic fundamental diagrams: A model predictive approach. *IEEE Transactions on Intelligent Transportation Systems*, 14(1):348–359, 2013. doi: 10.1109/TITS.2012.2216877.

[18] J. W. Godfrey. The mechanism of a road network. *Traffic Engineering & Control*, 11 (7):323–327, 1969.

[19] Qiangqiang Guo and Xuegang (Jeff) Ban. A multi-scale control framework for urban traffic control with connected and automated vehicles. *Transportation Research Part B: Methodological*, 175:102787, 2023. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2023.102787. URL https://www.sciencedirect.com/science/article/pii/S0191261523001121.

[20] Qiangqiang Guo and Xuegang(Jeff) Ban. Macroscopic fundamental diagram based perimeter control considering dynamic user equilibrium. *Transportation Research Part B: Methodological*, 136:87–109, 2020. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2020.03.004. URL https://www.sciencedirect.com/science/article/pii/S0191261519301407.

[21] Jack Haddad, Mohsen Ramezani, and Nikolas Geroliminis. Cooperative traffic control of a mixed network with two urban regions and a freeway. *Transportation Research Part B: Methodological*, 54:17–36, 2013. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2013.03.007. URL https://www.sciencedirect.com/science/article/pii/S0191261513000477.

[22] Mohammad Hajiahmadi, Victor L. Knoop, Bart De Schutter, and Hans Hellendoorn. Optimal dynamic route guidance: A model predictive approach using the macroscopic fundamental diagram. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 1022–1028, 2013. doi: 10.1109/ITSC.2013.6728366.

[23] Yuxuan Ji and Nikolas Geroliminis. On the spatial partitioning of urban transportation networks. *Transportation Research Part B: Methodological*, 46(10):1639–1656, 2012. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2012.08.005. URL https://www.sciencedirect.com/science/article/pii/S0191261512001099.

[24] Wan Li and Xuegang Ban. Connected vehicles based traffic signal timing optimization. *IEEE Transactions on Intelligent Transportation Systems*, 20(12):4354–4366, 2019. doi: 10.1109/TITS.2018.2883572.

[25] Ye Li, Mehmet Yildirimoglu, and Mohsen Ramezani. Robust perimeter control with cordon queues and heterogeneous transfer flows. *Transportation Research Part C: Emerging Technologies*, 126:103043, 2021. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2021.103043. URL https://www.sciencedirect.com/science/article/pii/S0968090X21000747.

[26] Louisiana Department of Transportation and Developement. Calibration validation. http://www.dotd.la.gov/Inside_LaDOTD/Divisions/Engineering/Traffic_Engineering/ManualsPublications/TEPR/Files/CalibrationValidation.pdf.

[27] Seattle Department of Transportation. Speed limits, 2022. https://www.seattle.gov/transportation/projects-and-programs/safety-first/vision-zero/speedlimits.

[28] Office of Financial Management. Census geographic files, 2010. https://ofm.wa.gov/sites/default/files/public/legacy/pop/geographic/tiger10/taz10.zip.

[29] Puget Sound Regional Council . Activity-based travel model: Soundcast, 2018. https://www.psrc.org/activity-based-travel-model-soundcast.

[30] QGIS Development Team. *QGIS Geographic Information System*. QGIS Association, 2024. URL https://www.qgis.org.

[31] Mohammadreza Saeedmanesh and Nikolas Geroliminis. Dynamic clustering and propagation of congestion in heterogeneously congested urban traffic networks. *Transportation Research Part B: Methodological*, 105:193–211, 2017. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2017.08.021. URL https://www.sciencedirect.com/science/article/pii/S0191261517304411.

[32] SDOT Traffic Counts Group. Traffic count studies by hour bins. https://data.seattle.gov/Transportation/Traffic-Count-Studies-by-Hour-Bins/g32r-fjzp/about_data.

[33] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. doi: 10.1109/34.868688.

[34] Isik Ilber Sirmatel and Nikolas Geroliminis. Economic model predictive control of large-scale urban road networks via perimeter control and regional route guidance. *IEEE Transactions on Intelligent Transportation Systems*, 19(4):1112–1121, 2018. doi: 10.1109/TITS.2017.2716541.

[35] Soheil Keshavarz. Simulation of downtown seattle am peak, 2024. https://github.com/BlueSoheil99/DowntownSeattleSUMO.

[36] Washington State DOT. Trafficflow loopgroup data retrieval, 2024. https://tracflow.wsdot.wa.gov/loopdata/loop_data_map.

[37] Wisconsin Department of Transportation. Vissim calibration settings. https://wisconsindot.gov/dtsdManuals/traffic-ops/manuals-and-standards/teops/16-20att6.3.pdf.

[38] Kaidi Yang, Nan Zheng, and Monica Menendez. Multi-scale perimeter control approach in a connected-vehicle environment. *Transportation Research Procedia*, 23:101–120, 2017. ISSN 2352-1465. doi: https://doi.org/10.1016/j.trpro.2017.05.007. URL https://www.sciencedirect.com/science/article/pii/S2352146517302843. Papers Selected for the 22nd International Symposium on Transportation and Traffic Theory Chicago, Illinois, USA, 24-26 July, 2017.

[39] Mehmet Yildirimoglu, Mohsen Ramezani, and Nikolas Geroliminis. Equilibrium analysis and route guidance in large-scale networks with mfd dynamics. *Transportation Research Part C: Emerging Technologies*, 59:404–420, 2015. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2015.05.009. URL https://www.sciencedirect.com/science/article/pii/S0968090X15001813. Special Issue on International Symposium on Transportation and Traffic Theory.

[40] Mehmet Yildirimoglu, Isik Ilber Sirmatel, and Nikolas Geroliminis. Hierarchical control of heterogeneous large-scale urban road networks via path assignment and regional route guidance. *Transportation Research Part B: Methodological*, 118:106–123, 2018. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2018.10.007. URL https://www.sciencedirect.com/science/article/pii/S0191261518301152.