A CONNECTED VEHICLE-BASED ADAPTIVE NAVIGATION ALGORITHM

FINAL PROJECT REPORT

by

Yinhai Wang
University of Washington

Sponsorship
Pacific Northwest Transportation Consortium (PacTrans)

for
Pacific Northwest Transportation Consortium (PacTrans)
USDOT University Transportation Center for Federal Region 10
University of Washington
More Hall 112, Box 352700
Seattle, WA 98195-2700

In cooperation with US Department of Transportation-Research and Innovative Technology Administration (RITA)

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The Pacific Northwest Transportation Consortium, the U.S. Government and matching sponsor assume no liability for the contents or use thereof.

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle	5. Report Date
A Connected Vehicle-Based Adaptive Navigation Algorithm	June 27, 2019
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Yinhai Wang, Wenbo Zhu, Meixin Zhu	2018-S-UW-1

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
PacTrans Pacific Northwest Transportation Consortium University Transportation Center for Federal Region 10 University of Washington More Hall 112 Seattle, WA 98195-2700	**11. Contract or Grant No.** 69A3551747110

12. Sponsoring Organization Name and Address	13. Type of Report and Period Covered
United States of America DEPARTMENT OF TRANSPORTATION Office of the Assistant Secretary for Research and Technology	Research **14. Sponsoring Agency Code**

15. Supplementary Notes

Report uploaded at www.pacTrans.org

16. Abstract

Connected vehicle (CV) technology leads to a system in which vehicles can communicate with other vehicles, transportation infrastructure, and other devices with communication capabilities. With increases in data availability, there is a great need for algorithms to process and utilize the data to improve system efficiency and mobility.

This study developed an adaptive navigation algorithm based on the data collection and communication functions in the CV based on user cost. To quantify the travel cost associated with different paths, a link cost function was developed to estimate both the link travel time and delay at the downstream intersection. An empirical intersection delay function was derived from stochastic queueing theory models. The developed function can support link cost estimation for interrupted traffic flow on local streets, which has been a limitation of previous navigation algorithms. On the basis of CV communication capabilities, a dynamic navigation algorithm as developed to suggest the optimal paths that dynamically minimize user cost.

The developed navigation algorithms were implemented in a microscopic simulation model using VISSIM application programming interface (API) functions. Multiple experiments were conducted to test the CV navigation algorithms in a virtual traffic environment based on the urban street network in downtown Bellevue, Wash. Experiment results revealed that the CV navigation algorithms were effective at reducing user cost in comparison to the static navigation used by non-CVs. The benefits of adaptive navigation algorithms will increase with CV market penetration, and the maximum benefit will be achieved when the CV penetration rate reaches around 60 percent. In the studied network, the marginal benefit of using the dynamic system optimum navigation over dynamic user equilibrium navigation was negligible (e.g., around 1 percent) given traffic flow randomness. Further experiments showed that the developed CV navigation algorithms can work effectively during non-recurrent congestion by properly balancing historical and real-time traffic information.

17. Key Words	18. Distribution Statement
Connected Vehicle, Navigation, User Equilibrium, Travel Time, Shortest Path	No restrictions.

19. Security Classification (of this report)	20. Security Classification (of this page)	21. No. of Pages	22. Price
Unclassified.	Unclassified.		NA

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

Table of Contents

List of Figures

List of Tables

Acknowledgments

The authors would like to thank the Pacific Northwest Transportation Consortium (PacTrans) for funding this research project and the City of Bellevue for providing traffic operation data for this project.

Executive Summary

Real-time data communication and analysis are important to support many smart transportation applications. Connected vehicle (CV) technology leads to a system in which vehicles can communicate with other vehicles, transportation infrastructure, and other devices with communication capabilities. With increases in data availability, there is a great need for algorithms to process and utilize the data to improve \ system efficiency and mobility.

This study developed an adaptive navigation algorithm based on the data collection and communication functions in the connected vehicle (CV) system. Specifically, the algorithm will utilize real-time traffic information to adaptively recommend the optimal path based on user cost. To quantify the travel cost associated with different paths, a link cost function is developed to estimate both the link travel time and delay at the downstream intersection. An empirical intersection delay function was derived from \ stochastic queueing theory models. The developed function can support link cost estimation for interrupted traffic flow on local streets, which has been a limitation of previous navigation algorithms. On the basis of the CV communication capabilities, a dynamic navigation algorithm was developed to suggest the optimal paths that dynamically minimize user cost.

The developed navigation algorithms were implemented in a microscopic simulation model using VISSIM application programming interface (API) functions. Multiple experiments were conducted to test the CV navigation algorithms in a virtual traffic environment based on the urban street network in downtown Bellevue, Washington. Experiment results revealed that the CV navigation algorithms were effective at reducing user costs in comparison to the static navigation used by non-CVs. The benefits of adaptive navigation algorithms will increase with CV market penetration, and the maximum benefit will be achieved when the CV penetration rate

reaches around 60 percent. In the studied network, the marginal benefit of using the dynamic system optimum navigation over the dynamic user equilibrium navigation was negligible (e.g., around 1 percent) given traffic flow randomness. Further experiments showed that the developed CV navigation algorithms can work effectively during non-recurrent congestion by properly balancing historical and real-time traffic information

1. Introduction

1.1 Background

The past several years have witnessed a major change with regard to data availability in the transportation domain. After decades of technology development for traffic management systems, vehicle-based technology has become the major focus of the next wave of innovation (Mahmassani, 2016). Connected vehicle systems (CVS) will enable information transfer from vehicle to vehicle (V2V), from vehicle to infrastructure (V2I), and from vehicle to any devices with communication capabilities (V2X). Such systems will provide large volumes of diverse, multi-source, and high-resolution data to support numerous applications in traffic planning, operations, and system optimization.

The implementation of CVS requires the installation of on-board units (OBUs) to vehicles and road-side units (RSUs) to the roadway infrastructure. According to the United States Department of Transportation (USDOT), connected vehicle (CV) technology is readily available in the vehicle manufacturing industry (USDOT, 2016). In 2016, the National Highway Traffic Safety Administration (NHTSA) issued a Notice of Proposed Rulemaking (NPRM) that aims to eventually require V2V devices in all new light vehicles. On the infrastructure side, the Vehicle to Infrastructure Deployment Coalition (V2I DC) has created a Signal Phase and Timing (SPaT) challenge that encourages the deployment of V2I devices in at least one corridor or network in each state by January 2020 (NOCoE, 2018). The goal of the challenge is to help state and local transportation infrastructure owners and operators better prepare for vehicles equipped with OBUs, as well as test the implementation and impacts of V2I applications.

With all the efforts in the deployment of CVS, however, a gap still exists between technology readiness and practical applications. Practical challenges have been revealed in the

implementation process, such as management at scale and cost/benefit assessment. To understand the concerns and challenges before the broad deployment of CV technology, the USDOT has funded three pilot programs to evaluate the feasibility of CVS in New York City, Tampa, and Wyoming (USDOT, 2018). The CVS deployed in these pilot sites include an integration of OBUs, RSUs, and mobile devices, and each site has designed and developed a CV-based application framework that specifically meets the region's unique transportation needs. One of the important objectives of these pilot projects is to measure the impacts of CV applications to ensure that the deployment of CVS will deliver the expected benefits to safety, mobility, efficiency, and the environment.

V2I devices provide CVs with data collected and processed from the transportation infrastructure. That said, transportation operators still face challenges in collecting and processing traffic information. According to the 2012 National Traffic Signal Report Card, the overall performance of 311,000 traffic signals in the U.S. was disappointingly rated at a level of D+ (NTOC, 2012). The report also assessed detailed grading of these signals in five designated themes, including management, signal operations; signal timing practices, traffic monitoring and data collection, and maintenance. Among these themes, traffic monitoring and data collection was rated the lowest, with a score of F. The implementation of RSUs will significantly improve the data collection capabilities of traffic signals, but a great need still exists for methodologies to process and analyze the collected data.

One of the important directions of CV-based applications is to enhance the system mobility and efficiency. In a complicated road network, multiple alternative paths can exist for a particular trip. Previous ways of selecting the optimal path have usually relies on historical traffic data and users' familiarity with the road. These navigation methods usually result in a

non-optimal traffic assignment with some roads over-traveled and others less traveled. If real-time traffic information can be obtained and utilized, it will be possible to develop a more efficient navigation algorithm that can reduce both user costs and system impacts. Such an algorithm can be implemented with the communication capabilities of the CVS, and it is expected that both CVs and non-CVs will benefit from a more efficient traffic assignment in the roadway network.

1.2 Problem Statement

Although a number of existing navigation applications have already been using real-time traffic data, some limitations still exist in the current navigation systems. First of all, the estimation of link travel cost mainly relies on vehicle speed data. This method usually can generate sufficiently accurate estimates for uninterrupted traffic flow (e.g., freeways and highways with controlled access), but for urban streets, the travel time is mainly controlled by the wait time at signalized intersections. Therefore, vehicle speed data are not enough to estimate the actual travel time on each link. To efficiently navigate on different types of roads, it is necessary to obtain data from both vehicles and traffic control signals so that the algorithm can work consistently for both uninterrupted and interrupted traffic flow.

The current navigation systems help users identify the optimal path on the basis of certain criteria (e.g., shortest travel time). In such a framework, each user's optimization procedure is independent from others, and everyone aims to maximize their own benefit. This leads the system to a user equilibrium (UE) state in which no one has the incentive to switch to a different path. However, the user equilibrium is not necessarily the optimal state for the entire system. The total system cost can be further reduced if some vehicles can navigate on the basis of system cost rather than individual user cost. The equilibrium state in which total system cost is minimized is

called the system optimum (SO) state. To estimate the total cost of a transportation system, it is necessary to know the traffic volume in each link. In the existing navigation applications, however, such optimization is not achievable without real-time data communication between vehicles and the infrastructure.

The current navigation applications calculate the optimal route at the start of a trip. Given the randomness in traffic, the recommended route might not continue to be optimal as the vehicle travels in the road network. Although some applications can recommend alternative routes on the basis of real-time traffic data, users are usually reluctant to switch routes when the benefit is small. In addition, the uncertain nature of traffic makes users skeptical of the estimated benefit of the alternative route. This results in under-utilization of the real-time traffic data, which leads to inferior navigation results.

To address the aforementioned limitations, this study aimed to develop a more efficient navigation algorithm that can adaptively recommend the optimal path based on the real-time traffic information. The whole algorithm was inspired by the rapid evolution of CV technology and the increasing need for system optimization. Such an algorithm will be of significant value to public agencies and private companies willing to efficiently manage a fleet of CVs and will provide insights to traffic operators and policy makers for traffic planning and system optimization in the CV environment.

1.3 Research Objectives

The goal of this study is to develop an adaptive navigation algorithm based on the data collection and communication functions of CVS. Specifically, two types of data sources, infrastructure (including intersections and road links) and connected vehicles, were utilized in the study. It was assumed that connected vehicles could receive real-timed traffic flow and travel

time information from the infrastructure, and the proposed navigation methodology would utilize such information to determine the real-time optimal path in the road network. The proposed navigation solution combines a set of innovative algorithms, including a link cost estimation function and a dynamic shortest-path algorithm based on probabilistic dynamic programming.

As a first step for the navigation algorithm, this study first developed a link cost function with special emphasis on the estimation of intersection delay. A general form of the link cost–traffic flow relationship was derived from traffic flow and queueing theory models. The link cost function was validated in various traffic scenarios with different levels of traffic flow randomness. The second layer of the algorithm involved a dynamic searching method for the optimal path based on real-time traffic information. This is different from existing navigation methods in which the optimal path is calculated on the basis of historical traffic data or traffic information at the start of the trip. The dynamic navigation method involves a Bayesian framework to predict future traffic conditions when the vehicle is expected to arrive at downstream links. The future traffic condition is predicted on the basis of both historical and real-time traffic data. In the Bayesian framework, the historical traffic pattern is used as the "prior" belief, and the real-time traffic data are incorporated to update the "prior" belief and generate more reliable "posterior" predictions of future traffic flow. Such a framework is more robust when real-time traffic is significantly different from the historical data. The classical shortest-path algorithm was adapted into a form of probabilistic dynamic programming in the sense that the cost of downstream links depends on previously visited links and cumulative travel cost.

In terms of the navigation objective, the proposed navigation algorithm supports user equilibrium navigation, which computes the time-dependent shortest path for each vehicle. The

navigation algorithm also allows CVs to adaptively change routing decisions based on real-time traffic information obtained during the trip, so that the most efficient network traffic assignment can be consistently achieved.

Both numerical experiments and microscopic simulations were conducted to implement and validate the proposed methodology. The experiments will quantify the benefits of the proposed navigation algorithm, as well as the impacts at different CV market penetration rates. The study holds great potential for public sector agencies and private companies that would operate and optimize a fleet of CVs. The analysis results will also provide insights into the communication requirements and policies related to future deployment of connected vehicles and devices.

The following bullet points summarize the key contributions of the project

1. Incorporated intersection delay into the link cost estimation so that navigation results are more accurate on local roads;
2. Implemented the CV navigation algorithm as a VISSIM module; and
3. Quantified the benefit of the proposed navigation method in experiments based on a real-world traffic environment.

2. Literature Review

2.1 Classical Shortest-Path Algorithm

The roadway network is represented as a directed graph $G(V, E)$, where $\{V\}$ is the set of vertices representing intersections and $\{E\}$ is the set of edges representing road links. Let V_i denote a specific vertex and E_{ij} denote the directed edge from V_i to V_j. The origin and destination vertices are represented by V_O and V_D, respectively.

In this example, we use travel time as the measure of "distance," and the shortest path corresponds to the path with the minimum total travel time. Let $t_{E_{ij}}$ represent the travel time associated with the edge E_{ij} and T_i is the time of arrival at V_i (i.e., the sum of travel times from all preceding edges). The Dijkstra algorithm (Dijkstra, 1959) that searches for the shortest path is illustrated as the following steps:

1. Initialize the network, set the arrival time to all vertices as infinity ($T_{V_i} = +\infty, \forall i$) and the arrival time to the origin vertex as zero ($T_{V_O} = 0$).
2. Initialize an empty parent hash map P to store the parent vertex of each vertex. For example, $P_{V_j} = V_i$ means V_i is the parent vertex of V_j (i.e., V_i is the immediate upstream vertex of V_j based on the current path searching process).
3. Initialize the "temporarily visited" group of vertices as $\{V_O\}$, and the "unvisited" group of vertices including all other vertices. Create an empty group of "permanently visited" vertices.
4. Find the vertex V_i with the minimum arrival time in the "temporarily visited" group. For all its unvisited neighbor vertices V_j, if we have $T_{V_i} + t_{E_{ij}} < T_{V_j}$, then update vertex V_j with the new arrival time $T_{V_j} = T_{V_i} + t_{E_{ij}}$ and update the parent hash map

with $P_{V_j} = V_i$. Move V_j into the "temporarily labeled" group. This step is illustrated in figure 2.1.

5. Move V_i into "permanently labeled" group.
6. Repeat steps 4-5 until the destination vertex V_D is in the "permanently visited" group. The shortest path can then be found through steps 7-10.
7. Initialize the path vertex set as $\{V_D\}$.
8. Let V_i denote the leftmost vertex in the set; if $V_i \neq V_O$, append its parent vertex of P_{V_i} to the left of the path vertex set.
9. Repeat step 8 until V_O is the leftmost vertex in the set. The shortest path consists of directed edges connecting every two adjacent vertices in the set from left to right.

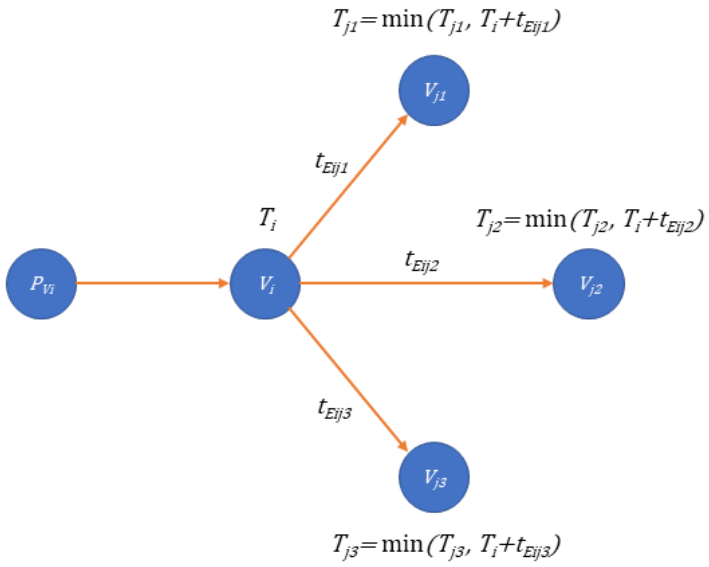


Figure 2.1 Updating step in the shortest-path algorithm

The major computation load of the Dijkstra algorithm comprises searching for the minimum distance node in the "temporarily visited" group. Research efforts have been devoted to speeding up the classical Dijkstra algorithm. From the data structure perspective, researchers

have used sorted data structures (e.g., bucket, heap/priority queue) to store the "temporarily visited" group of vertices. The vertex with the smallest distance is always placed first in the group every time after a vertex has been added or removed. In this fashion, the shortest-path searching is faster, and the computation efficiency is greatly improved. Other studies have focused on improving the algorithm from the methodology perspective (Wagner and Willhalm, 2007). Such techniques have included goal-directed searches (A* algorithm) (Hart et al., 1968), bidirectional searches (Luby and Ragde, 1989), and graph partitions (Möhring et al., 2007).

The aforementioned shortest-path algorithms are designed for road networks with a static travel cost. But, actual traffic is stochastic because of variations in multiple factors such as traffic demand, driving behaviors, and traffic control methods. Optimization of the stochastic problem depends on how the stochastic traffic is modeled, and the objective function. If the link travel cost is modeled as a random variable with known distribution, the optimal path that minimizes the total expected travel time can be found by using the classical shortest-path algorithm (Frank, 1969). With the development of real-time data collection and analysis methods, a number of studies have focused on dynamic navigation algorithms based on real-time information revealed during the trip. Eklund et al. (1996) developed a dynamic shortest-path algorithm for a changing traffic network and tested the effectiveness of the method in a simulation in which the road network was partially destroyed by an earthquake. Chen et al. (2012) incorporated the real-time data collected from traffic sensors into the navigation algorithm. The optimal path can be calculated on the basis of different criteria defined by both precise information (e.g., distance, lane type) and fuzzy data (e.g., travel time, traffic volume). Further studies developed adaptive navigation algorithms that can adaptively change the routing decision in response to new traffic information. On the basis of the conditional probability distribution of link travel time, Xiao and

Lo (2014) developed a navigation algorithm that adaptively calculated the optimal path using he en route traffic information. The algorithm is recalled to update the optimal path when the vehicle collects or receives new traffic data during the trip. Instead of calculating the complete path, Fu developed a closed-loop adaptive navigation method that suggests only the immediate next link (Fu, 2001). The future link time is modeled as a random variable with known mean and variation to quantify the expected travel time to the destination.

2.2 Travel Time Modeling

Travel time is a commonly used measure of travel cost. Knowledge of road travel time and its relationship with other traffic flow metrics is important to support the aforementioned navigation algorithms. This section introduces common methods to model travel time for different types of facilities.

2.2.1 Link Travel Time

The travel time on a link can be computed as the reciprocal of the space mean speed. The fundamental traffic flow diagram gives the relationship between traffic flow rate, speed, and density (Mannering and Washburn, 2013). The speed-flow relationship can be separated into two branches, with one representing the free-flow condition and the other representing the congestion condition. In the free-flow condition, the speed first remains almost constant at the free-flow speed when the flow rate is low. As the flow rate increases, the speed slightly decreases until the flow rate reaches capacity. The congestion branch of the speed-flow curve starts from the origin, as fully congested traffic has zero flow rate and zero speed. As traffic gradually recovers, the speed increases with the flow rate, and the trend is commonly expressed in a parabolic shape.

As shown by the fundamental traffic flow diagrams, travel time increases with the flow rate when traffic is not congested. Simplified functions have been developed to approximate the

relationship between travel time and flow. One commonly used travel time function is developed by the Bureau of Public Roads (BPR). The BPR link travel time function (Bureau of Public Roads, 1990) is expressed as

$$t(q) = t_0\left(1 + a\left(\frac{q}{c}\right)^b\right) \tag{2.1}$$

where t_0 is the free-flow travel time, q is the traffic flow rate (volume), and c is the capacity of the link. a and b are function parameters and can vary depending on the specific case studied.

The BPR function describes a general positive relationship between link travel time and flow rate. However, it is important to note that the BPR function is specifically designed for macroscopic analysis of uninterrupted traffic flow. Therefore, the BPR function is generally applied in traffic assignment tasks on highway networks for planning purposes, but it is usually not appropriate for modeling interrupted traffic flow on local roads.

2.2.2 Intersection Delay

Delay at intersections is a major component of the total travel time for interrupted traffic flow. Unlike the link travel time function, in which extra travel time is caused by congestion, intersection delay is caused by vehicles stopping and waiting at red signals. Therefore, intersection delay can be calculated by the wait time function in queuing theory. Variants of queueing models have been developed to address different types of arrival and departure flows. For a typical signalized intersection, the departure flow rate can be modeled as a periodic uniform distribution. During the red signal, the departure rate is zero, as no vehicle is allowed to travel. When the green signal starts and a vehicle queue exists, the departure flow rate becomes equal to the saturation flow rate, q_s. When the vehicle queue clears, the departure rate decreases and stays the same as the arrival rate for the remaining green signal. Note that here we only consider the case in which traffic is not saturated (i.e., the arriving traffic rate is lower than the

saturation flow rate). Given those factors, the departure flow at a signalized intersection is usually modeled as a piecewise deterministic flow in queueing analysis.

To understand the maximum arrival rate that can be served by the given signal control plan, the intersection capacity q_c can be expressed as

$$q_c = \frac{g}{C} q_s \tag{2.2}$$

where g and C are the effective green time and the cycle length, respectively, of the signal control plan.

When the average arrival rate is smaller than the intersection capacity, the average vehicle wait time can be estimated by using queueing theory functions. However, the specific intersection delay is dependent on the type of the arriving traffic. The following sections introduce models to estimate average vehicle delay for deterministic and stochastic arrival flows, respectively.

2.2.1.1 Deterministic Traffic Arrival

Deterministic traffic arrival means that the appearance of vehicles is exactly pre-defined. Note that the word "deterministic" does not necessarily infer uniform traffic arrival. The vehicle arrival is deterministic as long as the traffic flow rate and vehicle headway can be exactly described by any time-dependent function. Following this definition, deterministic traffic flow includes a uniform arrival rate, piecewise arrival rate, and changing arrival rate with some pre-defined function.

Figure 2.2 shows a traffic flow diagram assuming uniform vehicle arrival at signalized intersections. As the arrival traffic flow rate is smaller than the intersection capacity, vehicle queues can be fully discharged within each cycle.

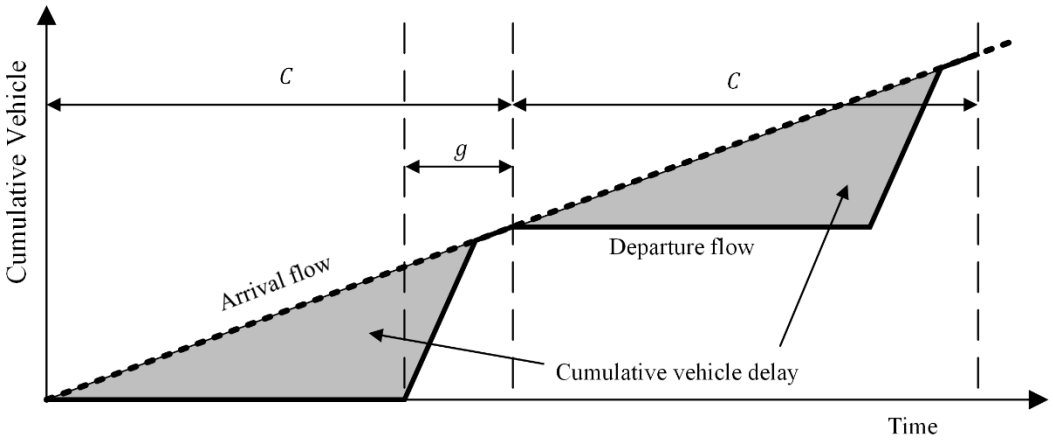


Figure 2.2 Traffic flow diagram for signalized intersection (uniform arrival)

The enclosed area between cumulative arrival departure vehicle curves (i.e., the shaded area in figure 2.2) represents the cumulative vehicle delay. If q denotes the deterministic arrival flow rate, then the average vehicle delay can be explicitly calculated as (TRB, 2010)

$$d_1 = \frac{(C-g)^2}{2C}\left(1-\frac{q}{q_s}\right)^{-1} \tag{2.3}$$

Note that equation 2.3 only works for situations in which the arrival flow rate is equal to or smaller than the intersection capacity (i.e., $q \leq q_c$). When $q > q_c$, then the green signal length is not enough to serve the traffic, and the average vehicle delay will increase as the vehicle queue extends over time. The 2010 Highway Capacity Manual (HCM) introduced an incremental delay term that increases linearly with the analysis period when the intersection is over-saturated (TRB, 2010). The incremental delay is expressed as

$$d_2 = 900T\left((X-1)+\sqrt{(X-1)^2}\right) \tag{2.4}$$

where $X = q/q_c$ is the degree of saturation, and T is the length of the analysis period.

Note that $d_2 = 0$ when $q < q_c$, as there are no undischarged vehicles when the uniform arrival flow rate is smaller than the intersection capacity. Combining the two equations, the general average vehicle delay function for deterministic arrival flow is expressed as

$$d = d_1 + d_2 = \frac{(C-g)^2}{2C}\left(1 - \min(X, 1)\frac{g}{C}\right)^{-1} + 900T\left((X-1) + \sqrt{(X-1)^2}\right) \quad (2.5)$$

2.2.1.2 Stochastic Traffic Arrival

Real-world traffic flow is not perfectly deterministic. Therefore, probabilistic models have been developed to analyze stochastic traffic flow. For a stochastic traffic flow, the number of vehicles observed (k) in a certain time period (T) is commonly assumed to follow a Poisson distribution, which can be expressed in the form of a probability mass function as

$$P(k) = e^{-\lambda}\frac{\lambda^k}{k!} \quad (2.6)$$

where $\lambda = qT$ is the expected number of observed vehicles during the time period. Given the Poisson-distributed vehicle appearance, the time interval between each two consecutive vehicles (i.e., headway, h) follows an exponential distribution, which is expressed in the form of a probability distribution function as

$$P(h) = \frac{1}{\beta}e^{-\frac{h}{\beta}} \quad (2.7)$$

where $\beta = 1/q$ is the inversed average flow rate, which is also the expected headway.

In comparison with deterministic flow, stochastic traffic arrival at an intersection causes additional delay because of occasional cycle failures. The arrival flow may occasionally exceed the intersection capacity during a short period, which leads to undischarged vehicle queues. It is important to note that the randomness in traffic can also depend on the flow rate. Under high traffic volumes, for example, vehicles move closer, with less variation in the headway. As a result, the traffic flow is approximately uniformly distributed. Additionally, under low traffic

volumes, randomness in traffic flow has little impact on vehicle delay. Therefore, the average vehicle delay for stochastic traffic arrival should be close to the results from equation 2.5 when the flow rate is very high or close to zero. A widely accepted way to model delay is to approximate the deterministic function results when the degree of saturation (X) is very small or very large, and then use a smooth curve to connect these two sections. The HCM recommends adding a random delay term to the deterministic incremental delay function to approximate the stochastic incremental delay (TRB, 2010), which is expressed as

$$d_2' = 900T\left((X-1) + \sqrt{(X-1)^2 + \frac{mkI}{q_c T}X}\right) \tag{2.8}$$

where parameters m, k, I are adjustments for flow randomness, signal control plan, and signal propagation, respectively.

Note that in a roadway network, the degree of randomness in traffic depends on the upstream traffic control plan. When an upstream signalized intersection exists, the downstream traffic flow is likely to follow a piecewise distribution similar to the departure flow rate from the upstream intersection. However, given multi-way intersections and different types of traffic control policies (e.g., turns on red), actual traffic flow is usually a mix of deterministic piecewise distribution and stochastic Poisson distribution. An accurate estimation of intersection delay needs to consider both types of arriving traffic flow.

3. Study Data

This study focused on developing a CV-based navigation algorithm that works for local road networks with signalized intersections. The urban street network from downtown Bellevue, Washington, was used as the test site for this study. Downtown Bellevue has a grid road system from Main Street (south) to NE 12th Street (north) and from Bellevue Way NE (west) to 112th Ave NE (east). The road network includes 35 intersections and 57 major bi-directional road links. The grid road system is effective for testing navigation algorithms because it provides sufficient alternative paths for any pair of origin and destination.

3.1 Roadway Geometrics

Geometrical information (e.g., length, curve, and number of lanes) for the road links was obtained from the City of Bellevue. Figure 3.1 presents the studied road network and the distribution of the road link lengths. The average link length was 664.4 ft, and differences in link lengths were mostly within 10 percent. Road links in two directions had similar lengths, creating square-sized blocks in the area.

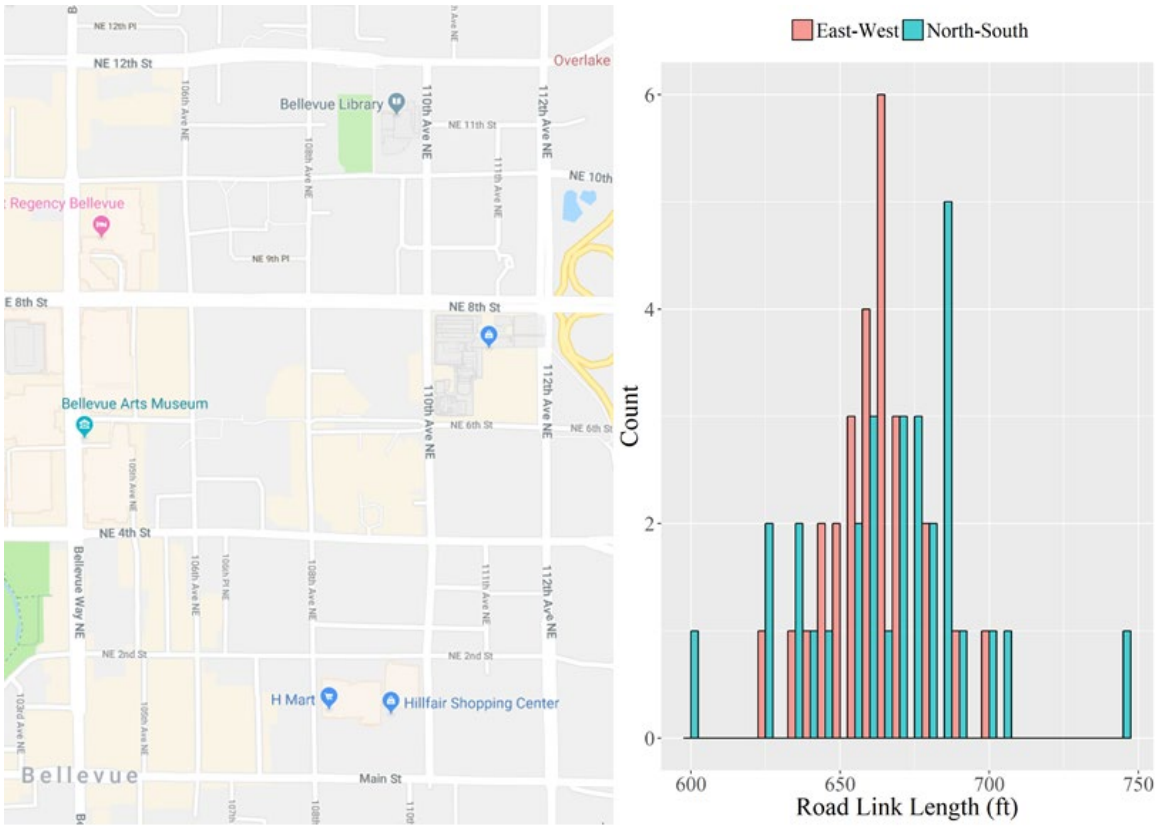


Figure 3.1 Studied roadway network and link length distribution

3.2 Traffic Data

To test the navigation algorithm in real-word traffic conditions, this study also included background traffic data in the analysis. The City of Bellevue has collected traffic counts by movement for each intersection in the midday off-peak period (i.e., 1:00 – 2:00 PM). Figure 3.2a shows the traffic counts for the NE 12th Street (north) and Bellevue Way NE intersection. For each direction, traffic counts are summarized for each of the three movements (i.e., left-turn, through, and right-turn).

Link traffic volumes were calculated by aggregating traffic counts for movements in the same direction. Note that there are two ways of calculating link traffic volumes. For a particular link, the traffic volume can be calculated as 1) the flow entering the link from the upstream

intersection; and 2) the flow exiting the link to the downstream intersection. Results from the two methods do not necessarily agree, as some trips may begin or end in the middle of the link (e.g., at buildings and parking lots alongside the link). This study applied the latter method to calculate link traffic volumes, as we were primarily interested in the link travel time, which is mainly controlled by the downstream intersection.

Link traffic volumes (veh/h/ln) are shown in figure 3.2b. According to the figure, traffic conditions were different in different parts of the area. More traffic was observed in the boundary roads and near the Bellevue Transit Center (in the center of the area).

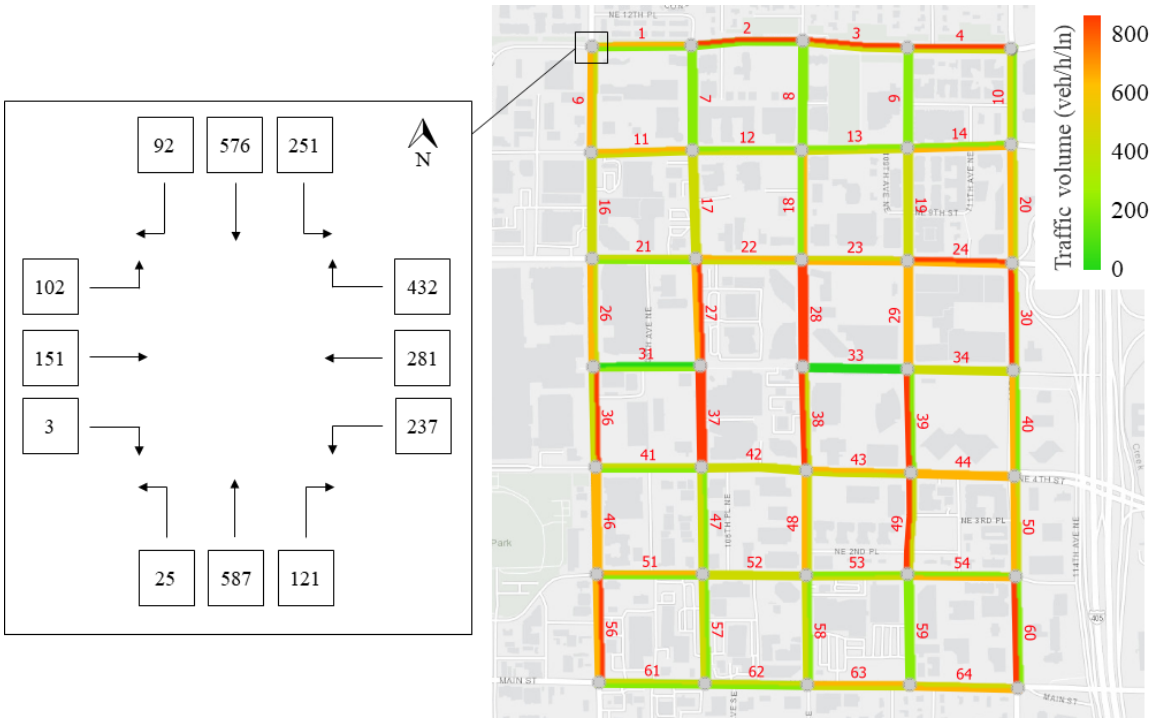


Figure 3.2 Intersection traffic counts and link traffic volumes for the studied network

3.3 Simulation Model

A microscopic traffic simulation model was developed by the City of Bellevue on the basis of the actual road geometry and signal control in the studied network. According to the

microscopic model, all intersections are controlled by an actuated traffic signal with the “gap-out” method. The microscopic simulation model was calibrated by the City of Bellevue with actual traffic data and has been used for planning and management purposes in the downtown Bellevue area.

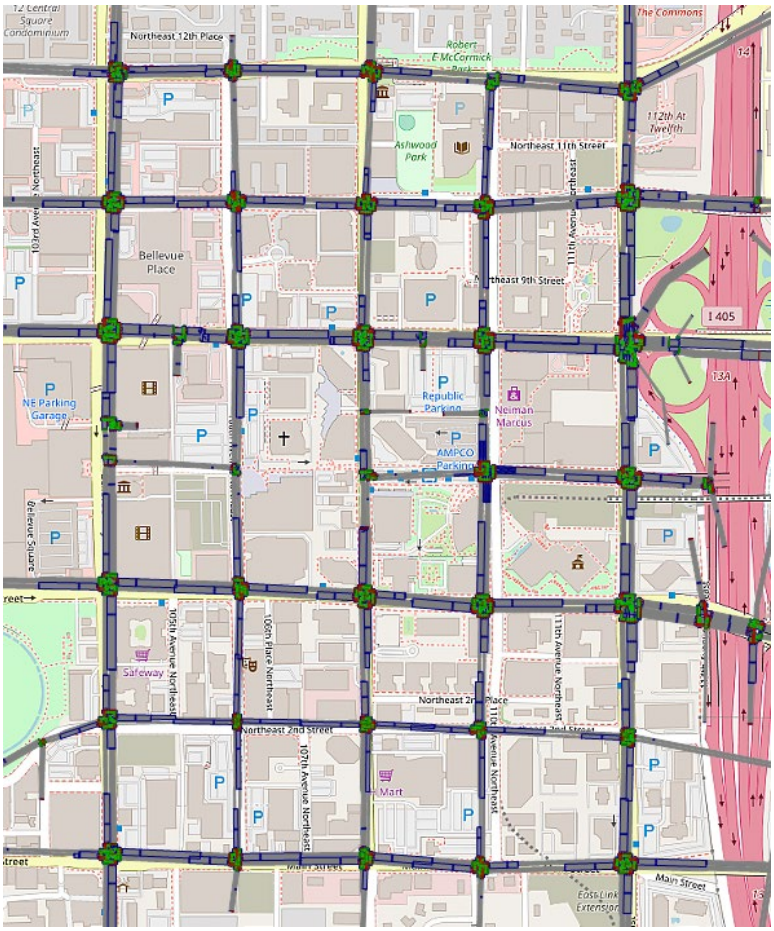


Figure 3.3 Microscopic simulation model of the studied network

4. Link Cost Estimation

Link cost estimation is the foundation of the navigation algorithm to find the path with the minimum total cost. In this study, travel time was used as the measure of travel cost. As previously mentioned, most previous studies have used macroscopic travel time functions (e.g., the BPR function) to calculate link travel time and validate the navigation algorithm. This method is effective in analyzing travel costs at the macroscopic scale, such as traffic planning and system design tasks. However, for microscopic traffic analysis (e.g., travel time on local roads and intersection wait time estimation), the BPR function fails to capture the detailed relationship between travel cost and traffic flow rate. To address such limitations, a link cost function was developed to estimate the time spent on a particular link and its downstream intersection. The link cost function describes the general relationship between travel time and traffic flow rate, and the function parameters can be derived from traffic data collected by CV devices. Numerical and simulation experiments were also conducted to validate the link cost function in various scenarios.

4.1 Base Form

This section discusses the development of the link cost function, with special emphasis on the downstream intersection delay estimation. The time spent on a particular link consists of two components: 1) the travel time on the link and 2) the wait time (delay) at the downstream intersection. The base form of the link cost function is expressed as

$$t(q) = t_0 + d(q) = \frac{l}{v_0} + d(q) \tag{4.1}$$

where $t(q)$ is the time spent on the link. $t_0 = l/v_0$ is the link travel time under the design speed (l and v_0 are the length of the link and the design speed, respectively). $d(q)$ is the delay at the downstream intersection, written as a function of the traffic flow rate, q.

While the link travel time can be directly calculated, estimation of the intersection delay is more complicated, and the details of function development are discussed in the following sections. Instead of summarizing a practical function with pre-determined parameters, this study focused on analyzing the general relationship between intersection delay and the traffic flow rate. A general form of the intersection delay function was derived from traffic flow and queueing theory, and the function parameters were derived from the actual traffic data. In this way, the link cost function utilizing data available from CVS could be uniquely determined for each individual link.

Note that in the base form, we assumed that link travel time is independent of the traffic flow rate. In the real world, however, with an increase in link traffic volume, additional delay may be caused by congestion-related events such as lane switching and shock waves. However, on local roads, congestion is not the main reason for delay, as traffic flow is primarily controlled by signalized intersections. Additionally, it is difficult to separate link congestion delay and intersection delay when travel time data are collected. Therefore, in the link cost function, congestion delay is also captured in the intersection delay function $d(q)$. As long as the function parameters are determined by total time spent on the link and the intersection, congestion delay will be included in $d(q)$, which increases with the traffic flow rate.

4.2 Intersection Delay Function

The development of an intersection delay function has been extensively studied, and various practical functions have been developed for planning purposes. An important limitation of previous functions has been that they often rely on pre-defined function parameters determined from previous field studies and lack the flexibility to handle different types of signal control plans under various traffic conditions. To overcome the limitation, this study focused on

deriving the general relationship between intersection delay and the traffic flow rate. The function parameters were derived from actual data collected at each individual link and its downstream intersection. Therefore, the link cost function will be eventually unique for each link, with its particular signal control and traffic condition.

4.2.1 General Forms for Uniform and Stochastic Arrivals

For uniform vehicle arrivals, the average vehicle delay can be explicitly calculated from equation 2.3 when the arrival flow rate is smaller than the intersection capacity, which is repeated here as

$$d = \frac{(C-g)^2}{2C}\left(1 - \frac{q}{q_s}\right)^{-1} \tag{4.2}$$

When the traffic flow rate is close to zero (i.e., $q \to 0$), the relationship between vehicle wait time and the time of arrival at the signal is as presented in figure 4.1. The average vehicle weight time equals the average height of the graph, which can be calculated as

$$d_0 = \frac{(C-g)^2}{2C} \tag{4.3}$$

This is equal to the result of equation 4.2 when $q = 0$.

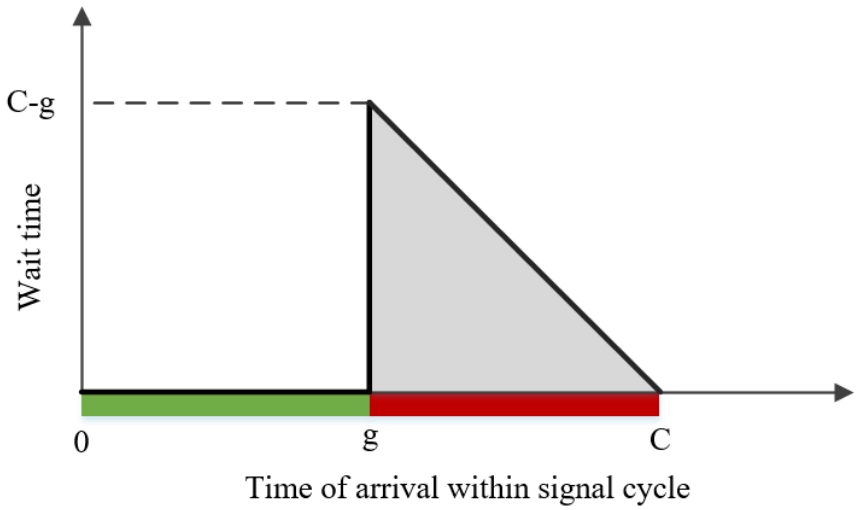


Figure 4.1 Relationship between vehicle wait time and time of arrival (fixed-time control)

For stochastic vehicle arrivals, the situation is more complicated because of occasional cycle failures even when the flow rate does not exceed the intersection's capacity. If we assume that traffic arrivals follow a Poisson distribution, then the average vehicle delay can be calculated by the Webster model (Webster, 1958) expressed as

$$d' = \frac{(C-g)^2}{2C}\left(1-\frac{q}{q_s}\right)^{-1} + \frac{q}{2q_c^2}\left(1-\frac{q}{q_c}\right)^{-1} \tag{4.4}$$

On the basis of field results, Webster further added an empirical correction term to reduce the estimation by around 10 percent (Webster, 1958). The revised function is expressed as

$$d' = \frac{(C-g)^2}{2C}\left(1-\frac{q}{q_s}\right)^{-1} + \frac{q}{2q_c^2}\left(1-\frac{q}{q_c}\right)^{-1} - 0.65\left(\frac{q_c}{q^2}\right)^{\frac{1}{3}}\left(\frac{q}{q_c}\right)^{2+\frac{g}{c}} \tag{4.5}$$

Note that in the intersection delay estimation, we use d to denote deterministic delay and d' to denote stochastic delay. On the basis of the Webster model, the stochastic delay has two major characteristics:

- The stochastic delay is greater than d_0 (i.e., the deterministic delay at $q = 0$) in all cases;
- The trend of the stochastic delay is mainly controlled by $\left(1 - \frac{q}{q_c}\right)^{-1}$.

While the first characteristic is intuitive, the second characteristic can be explained by considering the extreme condition. When the stochastic arrival flow rate approaches the intersection capacity (i.e., $q \to q_c$), the average vehicle delay will grow to infinity. This is because undischarged vehicle queues caused by occasional cycle failures are expected to carry over from cycle to cycle forever, as the signal timing plan is just enough to serve future traffic. Therefore, the average vehicle delay will keep increasing as the undischarged queue grows.

Utilizing the two characteristics concluded from previously developed intersection delay functions, we developed a general function that describes the overall trend of average vehicle delay under stochastic arrival flow rate, which is expressed as

$$d' = d_0 + a\left(1 - b\frac{q}{n \times q_c}\right)^{-1}, \quad q \in [0, nq_c) \tag{4.6}$$

where n is the number of lanes for multi-lane intersections.

The general stochastic delay function has two empirical parameters to derive from actual traffic data. Parameter a represents the scale of incremental delay. Parameter b represents the level of stochasticity in the traffic flow, and ranges between 0 and 1 when the traffic flow is between uniform and stochastic flows. Both parameters can be empirically derived from actual travel time data to ensure that the function results match field measurements. In comparison with traditional intersection delay functions, the above function is more adaptable to various scenarios with different parameter settings:

- If $a = 0$, the function equals the theoretical uniform delay function;

- If $a \neq 0, b = 0$, the function equals the uniform delay function with a fixed empirical correction factor;
- If $a > 0, b = 1$, the function approximates the theoretical stochastic delay function, with the value of a properly estimated.

4.2.2 Parameter Estimation

The general delay function form has two empirical parameters (a and b) to be derived from actual traffic data. This study estimated the delay function parameters that minimize the mean squared error (MSE) of the delay estimates. As $\{d_i\}_{i=1}^N$ and $\{q_i\}_{i=1}^N$ denote the set of average vehicle delay and traffic flow rate collected at an intersection, then the parameter estimation can be expressed in the form of a constrained optimization problem as

$$\min_{a,b} \quad \text{MSE}(a,b) = \frac{1}{N}\sum_{i=1}^{N}\left(d_i - d_0 - a\left(1 - b\frac{q_i}{n \times q_c}\right)^{-1}\right)^2 \tag{4.7}$$

$$\text{s.t.} \quad a \geq 0$$

$$b \geq 0$$

If the intersection capacity can be accurately estimated, then we should have $0 \leq \text{b} \leq 1$. However, it is difficult to estimate the exact intersection capacity because of complex traffic control rules such as shared lanes and phases by different movements. Therefore, we removed the constraint $b \leq 1$ so that the model can handle cases when we overestimate intersection capacity. As the objective function is continuous and smooth (i.e., the second derivatives exist and are continuous), the above problem can be solved by using a constrained optimization method.

After the intersection delay function is included into the base form of the link cost function, the complete function for link cost estimation is

$$t(q) = t_0 + d_0 + a\left(1 - b\frac{q}{n \times q_c}\right)^{-1} \quad (4.8)$$

On the basis of historical travel time and traffic flow rates, the delay function parameters can be derived by using data collected at each road link. Therefore, the delay function parameters will be different at each link with a specific traffic scenario. Such information will be stored in the RSUs and shared with other connected devices in the CVS.

4.3 Simulation Tests

The empirical delay function was utilized to learn the relationship between link travel time and traffic volume in a VISSIM simulation model. The objectives of simulation experiments included 1) validating the empirical delay function in the microscopic simulation environment with more complicated traffic characteristics (e.g., lane switching, acceleration/deceleration, and multi-lane intersections); and 2) developing link cost functions to be used in the CV navigation methodology.

To learn the link cost function, it was necessary to collect link travel time data under different traffic demand levels. Therefore, we changed the simulation vehicle input from 20 percent to 160 percent of the default traffic volume (i.e., vehicle counts collected during off-peak hours) to collect link travel time data under various traffic conditions. According to the simulation results, the majority of the roads were congested when the vehicle input was twice of the default value. Therefore, we did not further increase the traffic input beyond 160 percent in the simulation experiments.

The average travel time that vehicles spent on each link and its downstream intersection was collected during the simulation runs. For each vehicle input level, the simulation was repeated ten times with different random seeds. Each simulation run lasted for two simulation hours. The first simulation hour was the warm-up period, and vehicle travel time results were

only collected in the second simulation hour. In the following paragraphs, we introduce the detailed step of fitting the empirical delay function on the basis of a particular link (and its downstream intersection). Other links in the network can be analyzed with a similar procedure.

The studied link was southbound Bellevue Way NE at the intersection of NE 8th St and Bellevue Way NE. Figure 4.2 shows a screenshot of the intersection in the microscopic simulation model and the traffic count for each movement. The intersection was located in the busy commercial area in downtown Bellevue, and the four directions had nearly balanced traffic demand.

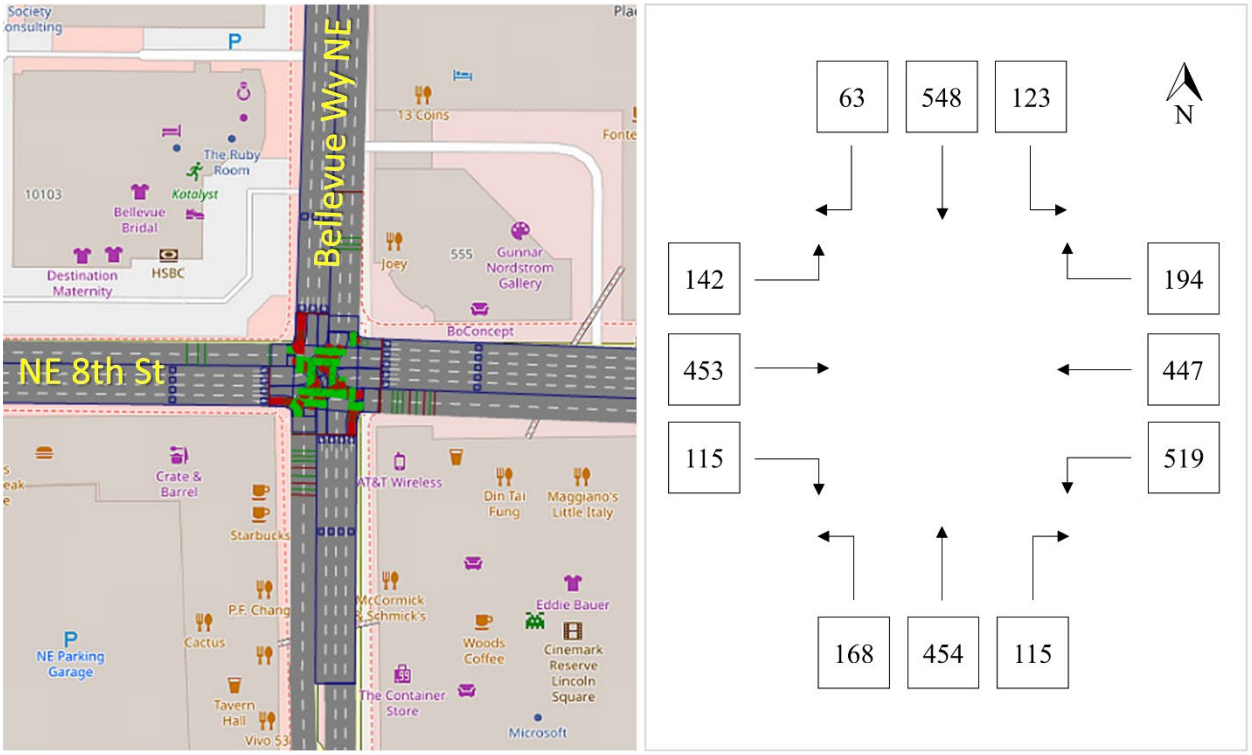


Figure 4.2 Example intersection in the microscopic simulation model

The first step was to estimate the pre-determined (or non-empirical) parameters in the link cost function. The base travel time, t_0, could be directly calculated from the link length and design speed. Given a link length of 661.4 ft and a design speed of 50 mph, we got $t_0 = 9.02$ s.

Estimation of the constant uniform delay term, d_u, was less straightforward, as the signal control was actuated. Note that from the empirical delay function, the minimum possible delay we could obtain was d_0 (i.e., the uniform delay assuming no randomness in traffic). Therefore, we set d_0 equal to a value slightly smaller than the minimum delay observed in the simulation results (e.g., $\min(d_i) - 10$ was used for the studied link). The intersection capacity q_c, again, could not be accurately estimated because of the actuated signal timing and complicated traffic control methods (e.g., shared lanes and phases by different movements). We simply assumed that the saturation headway was 2 seconds and the effective green indication ratio was 50 percent, which gave an intersection capacity of $q_c = 900$ veh/h. According to figure 4.1, we slightly underestimated the intersection capacity (as the actual capacity was around 1000 veh/h). Recall that the stochasticity parameter b can adjust to handle inaccurate capacity estimate and the assumption of $q_c = 900$ veh/h was sufficiently accurate to determine the empirical delay function.

The delay function parameters were estimated by minimizing the MSE of the function result. Figure 4.3 shows the simulation travel time results (in scatter points) and the fitted link cost function in the dashed curve. In general, the fitted link cost function accurately described the trend of link travel time results. The simulation test proved that the proposed delay function was effective at capturing the relationship between link travel time and traffic volume. Note that we used the hourly average link travel time to fit the link cost function. If the link travel time was aggregated at shorter time intervals (e.g., 15 minutes), then the travel time data would have greater variance, but the trend of the data would not be affected. When the link cost function is fit, it is recommended that the link travel time be aggregated by longer time intervals (e.g., an hour) to reduce the variance so that the trend of the link travel time can be accurately captured.

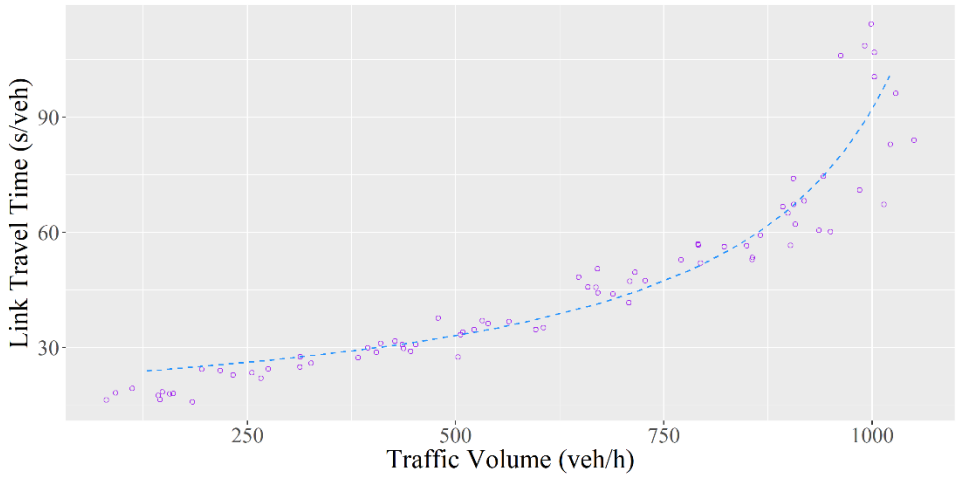


Figure 4.3 Simulated delay results and fitted delay function

The fitted link cost function was used in the navigation methodology to assess the travel times resulting from different route choice decisions. Note that a similar procedure was repeated for each link, and a unique link cost function was developed on the basis of travel time results collected at the link (and the downstream intersection) during the simulation runs.

5. Adaptive Navigation Algorithm

The vehicle navigation algorithms aim to find the optimal path from the origin to the destination that minimizes total travel cost (i.e., total travel time in this study). Depending on whether vehicles can receive real-time traffic information, different navigation algorithms can be applied to specific types of vehicles. This study defined two types of vehicles:

1. Non-connected vehicles that choose optimal routes on the basis of historical traffic data;
2. Connected vehicles that receive real-time traffic information and choose routes to minimize their individual travel cost.

The following sections discuss the specific navigation algorithm applied to each type of vehicle. Additionally, in the algorithm tests, the traffic was also considered to be a mixed flow of different types of vehicles.

5.1 Network Representation

The roadway network can be represented as a directed graph $G(V, E)$ consisting of edges and vertices, representing road links and intersections, respectively. Let $\{V\}$ denote the group of vertices and $\{E\}$ denote the group of edges. The number of vertices and edges can be expressed by $|V|$ and $|E|$, respectively. A path P is a collection of $|P|$ connected edges, expressed as

$$P = \left\{E_P^1, \dots, E_P^{|P|}\right\} \tag{5.1}$$

For a specific O-D pair, let $\mathbb{P}_{O,D}$ denote the set of all possible paths from the origin to the destination, written as

$$\mathbb{P}_{O,D} = \left\{P_{O,D}^1, \dots, P_{O,D}^{|\mathbb{P}_{O,D}|}\right\} \tag{5.2}$$

Each path $P_{O,D}^{i}$ is a unique path from the origin vertex to the destination vertex. For the specific O-D pair, the total number of possible paths is $\left|\mathbb{P}_{O,D}\right|$. The travel time (including both the link travel time and delay at the downstream intersection) associated with edge E is expressed as $t_E(q_E)$, which takes the form of the link travel time function (equation 4.8). The edge traffic volume, q_E, is a time-variant variable. If $p(q_E|T)$ expresses the relationship between edge traffic volume and time in the form of a conditional probability mass function, then the conditional expectation of edge traffic volume is

$$E(q_E|T) = \sum q_E \times p(q_E|T) \tag{5.3}$$

Note that to avoid confusion in notation, we use lower case t to represent a time period (e.g., the edge travel time) and upper case T to represent a time point (e.g., the trip start time). Equation 5.3 gives the time-variant average traffic volume, and this could be summarized from historical traffic data.

5.2 Navigation Methods

On the basis of the data availability and navigation objectives, two types of navigation methods were applied to the two vehicle types and a combination of them. The navigation methods were

1. Static navigation – find the optimal path on the basis of the static link cost
2. Dynamic UE navigation – find the optimal path, based on real-time traffic data, that minimizes total user cost.

For each navigation method, the travel time for edge E was estimated in a particular function. The following sections discuss the navigation algorithms used for each of the three types of vehicles.

5.2.1 Static Navigation

The non-connected vehicle navigation can be regarded as a static navigation problem. The travel time for all edges is pre-determined at the start time of the trip, T_0. The optimal route is the path with the minimum expected cumulative travel time. The vehicle navigation can be represented as an optimization problem expressed as

$$\min_{P \in \mathbb{P}_{O,D}} \quad \sum_{E_P^i \in P} t_{E_P^i} \left(E \left(q_{E_P^i} \middle| T_0 \right) \right) \tag{5.4}$$

Given the fixed cost for each edge, the optimal route can be found by using the classical shortest-path algorithm as described in Section 2.1. Note that the static navigation infers that the edge travel cost does not change in the optimal path searching process. However, this is not limited to vehicles using only historical traffic data. If the vehicle collects and calculates the optimal path on the basis of the edge travel time collected at the start of the trip, then this is still considered to be a static navigation problem, as the travel cost for each link does not change in the optimal path searching procedure.

5.2.2 Dynamic User Equilibrium Navigation

Static navigation aims to find the optimal path on the basis of knowledge before or at the start of the trip. However, the selected path may not continue to be "optimal" because of traffic incidents or non-recurrent congestion that cause changes in traffic during the trip. Because connected vehicles can collect real-time traffic information, they have the potential to find a more robust optimal path based on both historical traffic patterns and real-time traffic variations.

The dynamic vehicle navigation method aims to find the optimal path on the basis of estimated future travel time rather than static travel time. As the edge travel time changes with time, the downstream edge travel time depends on the expected vehicle arrival time, which is a

function of the travel times of all preceding edges starting from the origin. In other words, the algorithm predicts the future edge travel time at the time when vehicle is expected to arrive at the edge. For edge E_P^i in path P, the expected time that the vehicle will arrive at the edge, $T_{E_P^i}$, is the sum travel time from all previous edges, which is expressed as

$$T_{E_P^i} = \sum_{E_P^j \in P, j<i} t_{E_P^j} \tag{5.5}$$

Thus, dynamic navigation can be expressed in the form of an optimization problem as

$$\min_{P \in \mathbb{P}_{O,D}} \quad \sum_{E_P^i \in P} t_{E_P^i} \left(E\left(q_{E_P^i} \middle| \sum_{E_P^j \in P, j<i} t_{E_P^j} \right) \right) \tag{5.6}$$

Because the edge travel time is dependent on all previous travelled edges, calculating the edge travel time requires memorizing all previous steps in the algorithm. Therefore, a probabilistic dynamic programming framework was developed to solve the optimization problem.

The dynamic navigation algorithm aims to find the path with minimum individual user cost for each vehicle. In the equilibrium state, each vehicle selects its optimal path conditioned on the choice of all other vehicles, and no vehicle has an incentive to switch to a different path. This state is commonly known as the user equilibrium (UE) in traffic assignment. Therefore, this navigation method is referred to as Dynamic UE navigation in this study.

5.3 Adaptive Navigation Mechanism

The dynamic navigation algorithm was able to find the optimal path on the basis of expected changes in future traffic conditions. To further utilize real-time traffic information, an adaptive navigation mechanism was also developed to allow CVs to adaptively change their route choices. The adaptive routing method can be described as follows:

1. Estimate the node and link travel time on the basis of real-time traffic conditions.
2. For one single connected vehicle (or one subset of the connected vehicles), locate the current position and find its immediate downstream vertex V_i. Update its route as the shortest path from V_i to its destination V_D.
3. Update network traffic on the basis of new route choices.
4. Repeat steps 1-3 and sequentially loop through all connected vehicles.

The key procedure in the adaptive routing mechanism is that in each iteration only one CV (or one subgroup of CVs) will update its routing choice. In comparison to the case in which all CVs change routes simultaneously, the proposed mechanism requires less computational resources. Additionally, each CV will base its routing decision on previously processed CVs, as reflected by the updated network traffic conditions. The sequential updating mechanism will avoid all CVs making the same route choice, which might result in fluctuations in network traffic assignment.

Note that the sequential updating mechanism does not need extra effort in the implementation of the navigation algorithm. In real-world traffic, vehicles start their trips and receive traffic information at different times. Therefore, the navigation method will naturally be sequentially applied by different vehicles.

5.4 Probabilistic Dynamic Programming

5.4.1 Travel Time Prediction

The dynamic navigation algorithm relies on predicting the future edge travel time, which requires the estimation of future traffic volume, $E(q_E|T)$ $(T > T_0)$. As connected vehicles have access to both historical and real-time traffic data, a Bayesian method was developed to utilize both historical traffic patterns and real-time traffic measurements. Here we define a "time

period" as a length of time that periodically occurs with a similar traffic pattern (e.g., 12:00 – 1:00 PM on weekdays). For a particular edge at a specific time period, the historical and real-time traffic data used in the travel time prediction task are defined as

- Historical traffic volumes of the edge from the same time period
- Real-time traffic volume measurements from the edge from the same period.

On the basis of the two pieces of traffic information, future traffic volumes can be estimated from both historical traffic patterns and real-time traffic information. The basic idea behind Bayesian theorem is that a rational estimate procedure updates the previous belief with new information (Hoff, 2009). In the traffic volume prediction task, the historical traffic volume serves as the "previous" belief, and the most recent real-time traffic volume is considered to be the "new information." This method is based on periodically recurring traffic events and temporally correlated traffic flows. The traffic volume in a certain time period is likely to be similar, as that happened before in the same time period. Additionally, real-time traffic volume in the same time period might also suggest non-recurrent traffic incidents that should be considered in traffic flow prediction.

It is important to note that the equation for Bayesian method depends on the specific distribution of the data. Here we again consider different levels of traffic flow randomness. When the traffic flow is fully stochastic, the vehicle arrival can be approximated by using a Poisson distribution expressed as

$$q|T \sim \text{Poisson}(\lambda) \tag{5.7}$$

where $\lambda = E(q|T)$ is the expected traffic flow rate. Assuming that we have observed n_1 traffic flow rates $(q_{11}, \dots, q_{1n_1})$ in the time period T from the historical data, the joint probability of observing the data is

$$p(q_{11}, \dots, q_{1n_1} | \lambda) = \prod_{i=1}^{n_1} e^{-\lambda} \frac{\lambda^{q_{1i}}}{q_{1i}!} \propto \lambda^{\Sigma q_{1i}} e^{-n_1 \lambda} \tag{5.8}$$

Based on Bayesian theorem, the conditional probability of the Poisson distribution parameter is

$$p(\lambda | q_{11}, \dots, q_{1n_1}) = \frac{p(q_{11}, \dots, q_{1n_1} | \lambda) \times p(\lambda)}{p(q_{11}, \dots, q_{1n_1})} \propto p(\lambda) \times \lambda^{\Sigma q_{1i}} e^{-n_1 \lambda} \tag{5.9}$$

Therefore, the expected traffic flow rate, $\lambda = E(q|T)$, follows a distribution with the probability distribution function, including terms such as $\lambda^{c_1} e^{-c_2 \lambda}$. The simplest probability distribution that includes such terms is the family of Gamma distributions, and the corresponding probability distribution function is

$$p(\lambda) = \frac{b^a}{\Gamma(a)} \lambda^{a-1} e^{-b\lambda} \tag{5.10}$$

where a, b are distribution parameters and $\Gamma(\cdot)$ is the Gamma function. The mean and variance of Gamma distribution is $E(\lambda) = a/b$ and $Var(\lambda) = a/b^2$, respectively. If λ is based only on historical traffic data, when we have observed n_1 traffic flow rates, $q_{11}, \dots, q_{1n_1}$, λ follows a Gamma distribution with $a_1 = \Sigma q_{1i}$ and $b_1 = n_1$. The estimated average occupancy is

$$\lambda = E(q|T) = \frac{\Sigma q_{1i}}{n_1} \tag{5.11}$$

We use this as our previous belief of λ. In addition to the historical traffic data, if we further observe n_2 real-time traffic flow rates $q_{21}, \dots, q_{2n_2}$, then the posterior distribution of λ is

$$\begin{aligned} p(\lambda | q_{21}, \dots, q_{2n_2}) &\propto p(\lambda) \times \lambda^{\Sigma q_{2j}} e^{-n_2 \lambda} \propto \lambda^{a_1 - 1} e^{-b_1 \lambda} \times \lambda^{\Sigma q_{2j}} e^{-n_2 \lambda} \\ &\propto \lambda^{(a_1 + \Sigma q_{2j}) - 1} e^{-(b_1 + n_2)\lambda} \end{aligned} \tag{5.12}$$

This follows a amma $(a_1 + \Sigma q_{2j}, b_1 + n_2)$ distribution, and the posterior parameter estimate is

$$\lambda = E(q|T) = \frac{a_1+\Sigma q_{2j}}{b_1+n_2} = \frac{\Sigma q_{1i}+\Sigma q_{2j}}{n_1+n_2} \tag{5.13}$$

On the basis of the Bayesian theorem, the posterior estimation of expected flow rate takes the form of the weighted average of both historical and most recent data. Here n_1 and n_2 are the number of traffic flow rate observations from historical data and real-time data, but they can be generally interpreted as our "beliefs" in historical and real-time traffic. Therefore, the number of historical and real-time measurements included in the prediction can be adjusted according to our belief of the data. For example, for a road with strong traffic patterns in the past, we might include more historical data and less real-time data to generate robust traffic flow estimations. For a road with more non-recurrent traffic incidents, however, we might include more real-time traffic flow rates to ensure that real-time traffic changes are sufficiently reflected in future traffic flow prediction.

Equation 5.13 predicts the future traffic flow rate when vehicle arrival is fully stochastic. Under deterministic flow, the time-conditioned traffic volume is a constant, which can be expressed as

$$q|T \equiv E(q|T) \tag{5.14}$$

Under the deterministic flow assumption, the observed n_1 historical traffic volumes and n_2 real-time traffic volumes will all equal the time-conditioned constant traffic volume, which is expressed as

$$q_{1i} = q_{2j} \equiv E(q|T), 1 \le i \le n_1, 1 \le j \le n_2 \tag{5.15}$$

According to equation 5.15, equation 5.13 still holds for deterministic traffic flow. Therefore, the Bayesian-based traffic volume prediction function generally works when the actual traffic is a combination of deterministic and stochastic flow. On the basis of the estimated

future traffic flow rate, the future travel time can thus be predicted by using the link cost function.

5.4.2 Time-Dependent Shortest-Path Searching

For the static navigation problem, the optimal path can be found by using the traditional shortest-path algorithm. However, the dynamic UE navigation problem requires predicting future travel time on the basis of previously selected edges. The cost for each edge is not a static value but rather depends on the previous steps in the algorithm. Therefore, a revised shortest-path search method was developed based on the bottom-up dynamic programming algorithm. For the individual dynamic navigation problem described in Section 5.2.2, the dynamic shortest-path searching method is summarized as below.

For the roadway network represented as a directed graph $G(V, E)$, let V_i denote a specific vertex and E_{ij} denote the directed edge from V_i to V_j. T_i is the time of arrival at V_i, and $t_{E_{ij}}$ is the travel time associated with the edge E_{ij}. The algorithm to find the fastest route (i.e., the route with the minimum total travel time) from the origin vertex V_O to the destination vertex V_D takes the following steps:

1. Initialize the network, set the arrival time to all vertices as infinity ($T_{V_i} = +\infty, \forall i$) and the arrival time to the origin vertex as zero ($T_{V_O} = 0$).
2. Initialize an empty parent hash map P to store the parent vertex of each vertex. For example, $P_j = i$ means the V_i is the parent vertex of V_j (i.e., V_i is the immediate upstream vertex of V_j).
3. Initialize the "temporarily visited" group of vertices, including the origin vertex, and the "unvisited" group of vertices, including all other vertices. Create an empty group of "permanently visited" vertices.

4. Find the vertex V_i with the minimum arrival time in the "temporarily visited" group. For all its unvisited neighbor vertices V_j, calculate the conditional edge travel time $t_{E_{ij}}|T_{V_i}$.
5. Loop through all V_j. If we have $T_{V_i} + t_{E_{ij}}|T_{V_i} < T_{V_j}$, then update vertex V_j with the new arrival time $T_{V_j} = T_{V_i} + t_{E_{ij}}|T_{V_i}$ and update the parent hash map with $P_{V_j} = V_i$. Move V_j into the "temporarily labeled" group.
6. Move V_i into "permanently labeled" group.
7. Repeat steps 4-6 until the destination vertex V_D is in the "permanently visited" group. The shortest path can then be found by tracing the parent vertex from the destination all the way back to the origin.

In comparison with the traditional shortest-path algorithm, the revised dynamic shortest-path algorithm has the additional step of updating the edge travel time conditioned on the cumulative travel time calculated from previous steps. The conditional edge travel time can be estimated by using the Bayesian function derived in the previous section, given that CVs can access both historical and real-time traffic data.

6. VISSIM Navigation Module

One of the major objectives of the study was to test the CV-based navigation algorithm in a microscopic simulation environment. Although previous studies have developed simulation-based approaches to evaluate different dynamic traffic assignment (DTA) strategies, the majority of the studies have relied on numerical simulation models or analysis at the mesoscopic scale. This study will build the vehicle navigation algorithms in a microscopic vehicular simulation platform. Specifically, this study developed a vehicle navigation module utilizing the application programming interface (API) in VISSIM, which is a widely used commercial simulation tool for microscopic traffic modeling and analysis.

6.1 Introduction to VISSIM

VISSIM is a leading microscopic simulation software program utilized worldwide by research institutes, consulting companies, and public sector agencies (PTV, 2016a). It allows multimodal transportation modeling and operation analysis at the detailed vehicular level, and therefore, it has been deployed in numerous locations to address various issues such as transportation planning, traffic control and management, and environmental impacts.

6.1.1 Simulation Model

This study utilized the VISSIM model of downtown Bellevue provided by the City of Bellevue. A detailed introduction to the VISSIM model can be found in Chapter 3. The network geometrics and traffic characteristics were established and calibrated by the City of Bellevue and were therefore used consistently without change in this study. Specifically, the information included the following:

- Road geometrics: link length, curvature, number of lanes, etc.

- Driving behaviors: designed speed, car-following model, accelerations/decelerations, etc.
- Intersection geometry: stop signs, conflict areas, priority rules, detector locations, etc.
- Traffic signals: signal control type, signal phasing/timing, etc.

In addition to the basic simulation model, hourly traffic counts from the midday off-peak period were also collected and provided by the City of Bellevue. These data were used as the background traffic in the simulation analysis.

6.1.2 VISSIM COM API

The development of the CV-based navigation module was supported by the VISSIM Component Object Model (COM) API. COM provides data access to VISSIM objects, methods, and results, which can be adjusted or evaluated during or before/after simulation (PTV, 2016b). Additionally, COM supports a wide variety of languages (e.g., VBA, C, C++, MATLAB, and Python). Users are allowed to develop their own functions and applications that interact with COM objects to achieve different traffic control and operation purposes.

According to the COM API manual, VISSIM model objects that can be accessed in the COM API are organized in a hierarchical structure, as shown in figure 6.1. The highest-ranking object is IVissim, which must be called to access any sub-objects in the structure. INet is an important object that includes all sub-objects related to the model network (e.g., links, intersections, and vehicle input). VISSIM model parameters can also be accessed through the corresponding category (e.g., ISimulation contains simulation parameters that can be adjusted through the COM API).

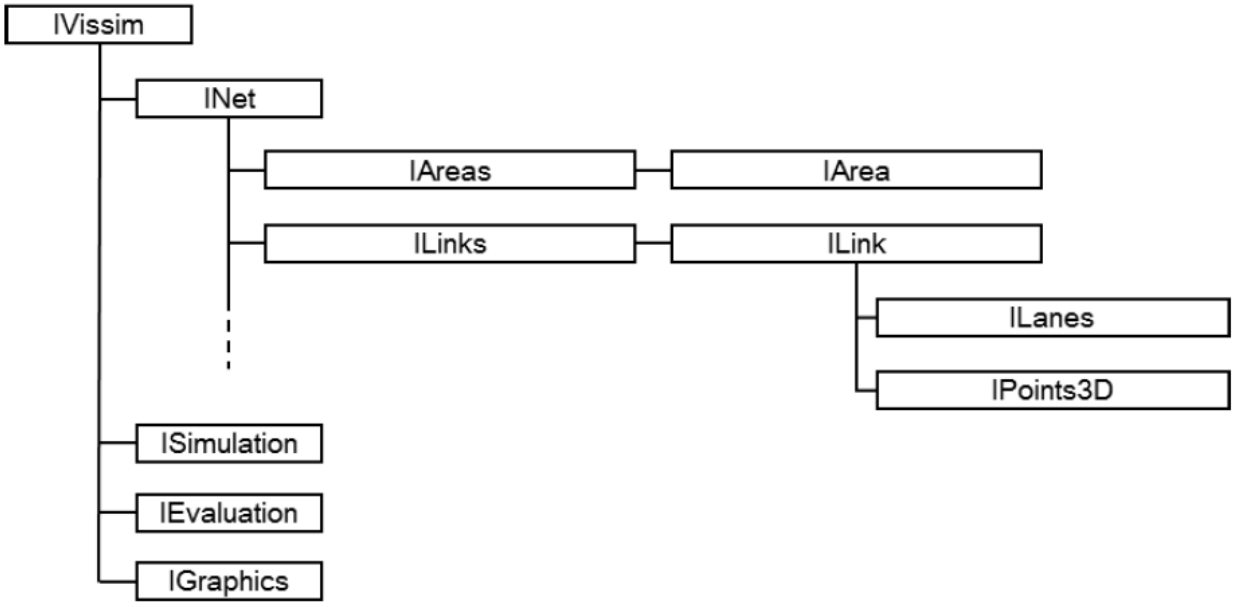


Figure 6.1 VISSIM COM object hierarchy

This study developed a Python program to achieve the CV-based navigation algorithms in the COM API. The program was integrated into the VISSIM model as a navigation module to dynamically navigate connected vehicles. During a simulation run, the navigation module would interact with a number of COM objects to achieve real-time data collection and coordinated navigation decisions.

6.2 VISSIM Objects and Parameters

This section introduces major VISSIM objects and parameters that were used in the navigation module. The majority of the VISSIM network objects and model parameters had corresponding COM objects and attributes that could be accessed in the COM API.

6.2.1 Vehicle Inputs

Vehicle inputs are placed at the start points of road links and specify traffic flows that enter the studied network from different locations. Three groups of vehicle inputs were added to the simulation model, namely background traffic, non-connected vehicles, and connected

vehicles. To consider traffic flow randomness in the simulation experiment, all vehicle inputs assumed the traffic volume type to be stochastic.

The background vehicle inputs were set up on the basis of the hourly vehicle counts provided by the City of Bellevue. The background traffic also included 2 percent of heavy goods vehicles (HGVs). Non-CV and CV vehicle inputs were added to the model at the corresponding route origins. Here we assumed that all non-CVs and CVs were passenger cars. In the simulation experiments, different levels of traffic flow rate and CV penetration were tested, and the non-CV and CV vehicle input values were adjusted accordingly before simulation runs.

6.2.2 Travel Time Measurement

Real-time traffic volume and travel time data collection is an important task for the CV navigation framework. In VISSIM, such data can be obtained by using vehicle travel time measurements. Figure 6.2 shows an example vehicle travel time measurement in VISSIM. A vehicle travel time measurement consists of one start bar and one end bar. This study collected link-based travel time, and therefore the start and end bars were placed at start points of different road links (note that the end bar of a travel time measurement was placed at the start of one downstream link). Every time a vehicle passed the start and end bar, the travel time in between was collected. In this fashion, the travel time that vehicles spent at an intersection area was included in the upstream link travel time. The average travel time and number of vehicles were then aggregated by vehicle class, and data collection interval defined in the evaluation parameters.

As illustrated in figure 6.2, one road link typically needed three travel time measurements (assuming a four-way downstream intersection) to collect left-turn, through, and right-turn traffic, respectively. The collected vehicle counts and travel time data could be analyzed

separatel, or aggregated to evaluate the overall link performance. In the studied network, the majority of the traffic signals had a shared signal phase for both left-turn and through movements. Therefore, the travel time and vehicle counts in different movements were aggregated at the link level for link performance modeling and real-time traffic data collection. By default, vehicle travel time measurements collected the total vehicle count and average travel time for all vehicle classes. However, the results could also be aggregated by vehicle class, if the corresponding VISSIM evaluation parameter was properly specified.

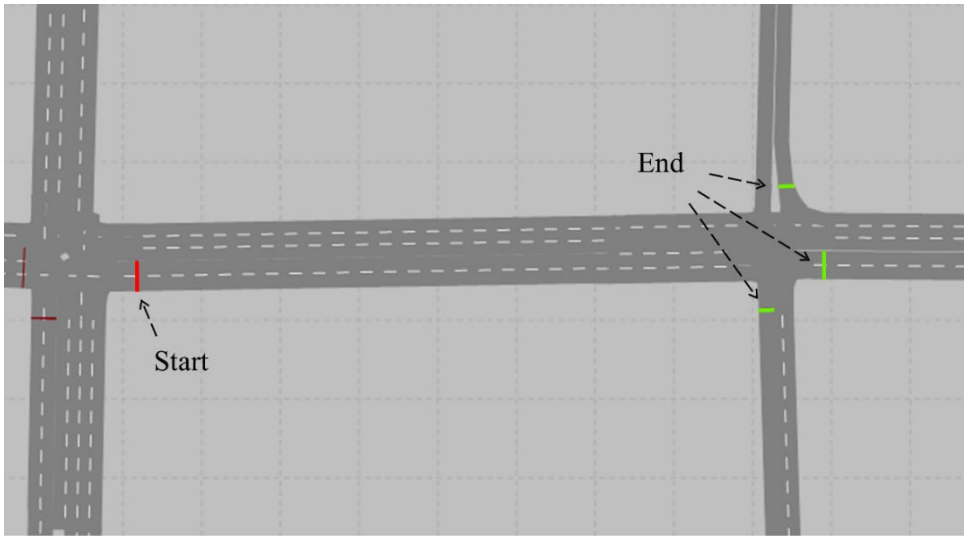


Figure 6.2 An example VISSIM vehicle travel time measurement

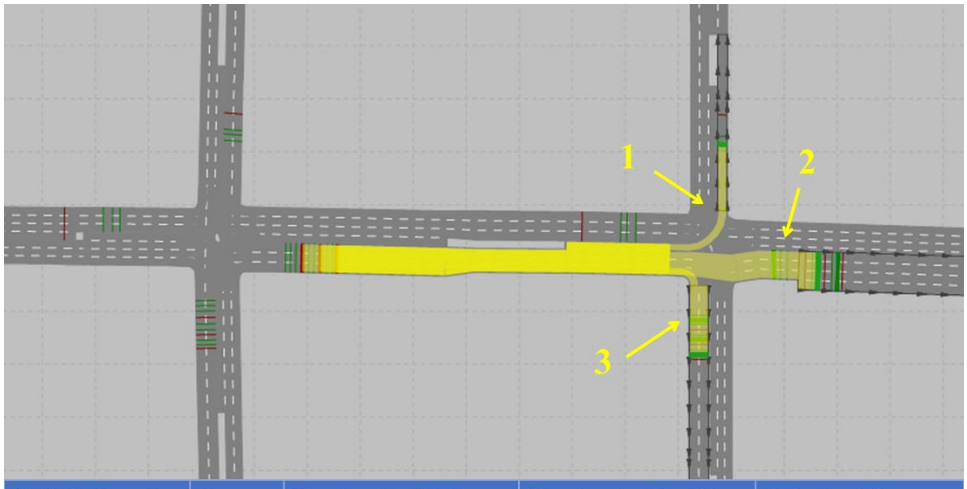
Before the navigation algorithms were tested, the vehicle travel time measurements were used to collect traffic volumes and average travel times at different links (and downstream intersections) under various levels of traffic input. The simulation results were then used to learn the relationship between link travel time and traffic flow rate (i.e., the link cost function). Details about the link cost function estimation can be found in Section 4.3.

6.2.3 Vehicle Routing Decisions

In VISSIM, vehicle movements are controlled by vehicle routing decisions. VISSIM provides two major types of vehicle routing decisions: static and dynamic vehicle routing decisions. Static vehicle routing decisions are placed at fixed locations and give the traffic flow distribution as relative flows for different movements. Dynamic vehicle routing decisions, in contrast, have a built-in DTA model to distribute the network traffic flow to a number of alternative routes. Dynamic traffic assignment is determined by using an iterative approach. The network is simulated multiple times, and drivers are allowed to choose their paths on the basis of the driving experiences in the preceding simulation. Such an approach is effective at calculating the theoretical optimal traffic assignment and providing insights for traffic planning and management. However, the built-in DTA model is not a practical method, as real-world drivers do not actually experience several different routes before selecting their path. Recall that this study developed a customized dynamic vehicle navigation methodology to approximate DTA equilibrium conditions through real-time communication in the CVS. Therefore, instead of relying on the built-in DTA model, this study used static vehicle routing decisions and dynamically changed the relative flow distribution by using the COM API functions to achieve dynamic traffic assignment results.

Figure 6.3 shows an example static vehicle routing decision in the VISSIM model. One static vehicle routing decision typically consists of one start bar (i.e., the red bar) and several end bars (i.e., the green bars) corresponding to different routes, respectively. Once a vehicle passes the start bar, the model selects one of the routes with probabilities derived from the relative flow rate. Note that because the route choice is decided for each discrete vehicle, the final traffic flow distribution will not be exactly equal to the relative flow rate. Instead, the relative flow values

give an overall share of different route choices during the simulation run. As illustrated in the figure, the static vehicle routing decisions are placed at intersections, and each route corresponds to a particular movement (e.g., left-turn, through, or right-turn movement). Note that to ensure sufficient distance for vehicles to switch lanes, the start bar of the static vehicle routing decision needs to be placed near the start of the link.



Vehicle Routing Decision	Route No	Destination Link	Destination Position (ft)	Relative Flow (veh/h)
55	1	26: 110th Avenue NE	81.7	72.0
55	2	72: NE 8th Street	46.9	1192.0
55	3	97: 110th Avenue NE	43.0	95.0

Figure 6.3 An example VISSIM static vehicle routing decision

The static vehicle routing decisions can also be set to apply only to a certain type of vehicle. Therefore, three static vehicle routing decisions were placed at each approach for each intersection: one for background traffic, one for non-CVs, and one for CVs. For background traffic, the relative flows were set equal to the vehicle counts of different movements at each intersection. For non-CVs, the relative flows were based on route choice decisions calculated from the static navigation algorithm. For CVs, the relative flows were periodically updated on the basis of dynamic navigation methods developed in the COM API.

6.2.4 Basic Parameters

In addition to network objects, the COM API also allows access to basic VISSIM environmental parameters. Basic parameters that will be adjusted in the navigation module fall into two major categories: simulation and evaluation parameters. These parameters were set as needed before simulation runs to apply different data collection plans and to test the navigation algorithms in a variety of traffic conditions.

Table 6.1 lists basic VISSIM parameters that were adjusted in the navigation module using the COM API functions. For simulation parameters, SimPeriod controlled that each simulation run lasted for two simulation hours (7,200 seconds). Specifically, the first simulation hour was generally considered to be the "warm-up" period, while the actual simulation results were collected during the second simulation hour. Random seeds were set before each simulation by using the RandSeed COM attribute. For each model input, ten different random seeds were used to obtain repeated results with variation. The NumRuns attribute can be used to repeatedly run the same simulation with different random seeds. In this study, as we needed to dynamically adjust vehicle navigation decisions in each simulation, this function was not used, and NumRuns was set to 1.0. During a simulation, the breakpoint was set by using the SimBreakAt attribute so that the simulation would pause to collect data and update vehicle routing decisions. Note that when the simulation stopped at a break point, network conditions (e.g., traffic signal states, vehicle locations and speed, etc.) held until the simulation resumed. In the simulation experiments, break points were set at 300-second intervals so that the vehicle navigation decisions were updated every 5 minutes.

Table 6.1 Basic VISSIM parameters adjusted in the navigation module

COM Attribute	Parameter Type	Parameter Description	Parameter Value
SimPeriod	Simulation	Total simulation seconds	7200
RandSeed	Simulation	Random seed	Vary
NumRuns	Simulation	Number of simulation runs	1
SimBreakAt	Simulation	Breakpoint simulation will stop at	300 intervals
VehClasses	Evaluation	Vehicle classes that travel time results will be aggregated by	Non-CV, CV, All
VehTravTmsCollectData	Evaluation	Whether to collect the vehicle travel time data	1 (true)
VehTravTmsFromTime	Evaluation	Start of the vehicle travel time data collection period	0 (from the start of the simulation)
VehTravTmsToTime	Evaluation	End of the vehicle travel time data collection period	99999 (to the end of the simulation)
VehTravTmsInterval	Evaluation	Time intervals that vehicle travel time results will be aggregated by	300

Evaluation parameters control how simulation results are collected and organized. Specifically, the navigation module requires vehicle count and travel time data collected by vehicle travel time measurements. The VehClass attribute can be specified as a list of vehicle classes by which the simulation results are aggregated. Because we analyzed detailed navigation effects on each vehicle class, the VehClass attribute was set to aggregate results by non-CVs, CVs, and all vehicles. The data collection period can be specified by the VehTravTmsFromTime and VehTravTmsToTime attributes. In this study, the traffic data were collected during the entire simulation, but only the second simulation hour data were used for result analysis. The vehicle count and travel time results can also be aggregated by time interval, as specified by the VehTravTmsInterval attribute. As the navigation module updated vehicle routing decisions on a

5-minute basis, the traffic data were also aggregated by 300-second intervals so that the most real-time data could be utilized.

In addition to the above-mentioned parameters, some other parameters (e.g., simulation speed, result format) were also specified to set up the basic simulation environment. Those parameters were not necessarily related to the navigation module framework.

6.3 Navigation Module Structure

The navigation module was built in a Python program that connected to the VISSIM COM API through the PyWin32 extension. Figure 6.4 shows the overall structure of the VISSIM navigation model. In each step of the framework, the navigation model utilized corresponding COM objects to complete the task (e.g., set parameters, collect data, update routing decisions, etc.). As non-CVs use historical data to navigate, their routing decisions were set before the start of the simulation. During the simulation, traffic data collection and CV navigation were conducted in parallel at 5-minute intervals. Note that CVs sequentially navigated and communicated route choice decisions following the coordinated navigation mechanism introduced in Section 5. The CV routing decision objects in the VISSIM model were then updated on the basis of the combined CV navigation results.

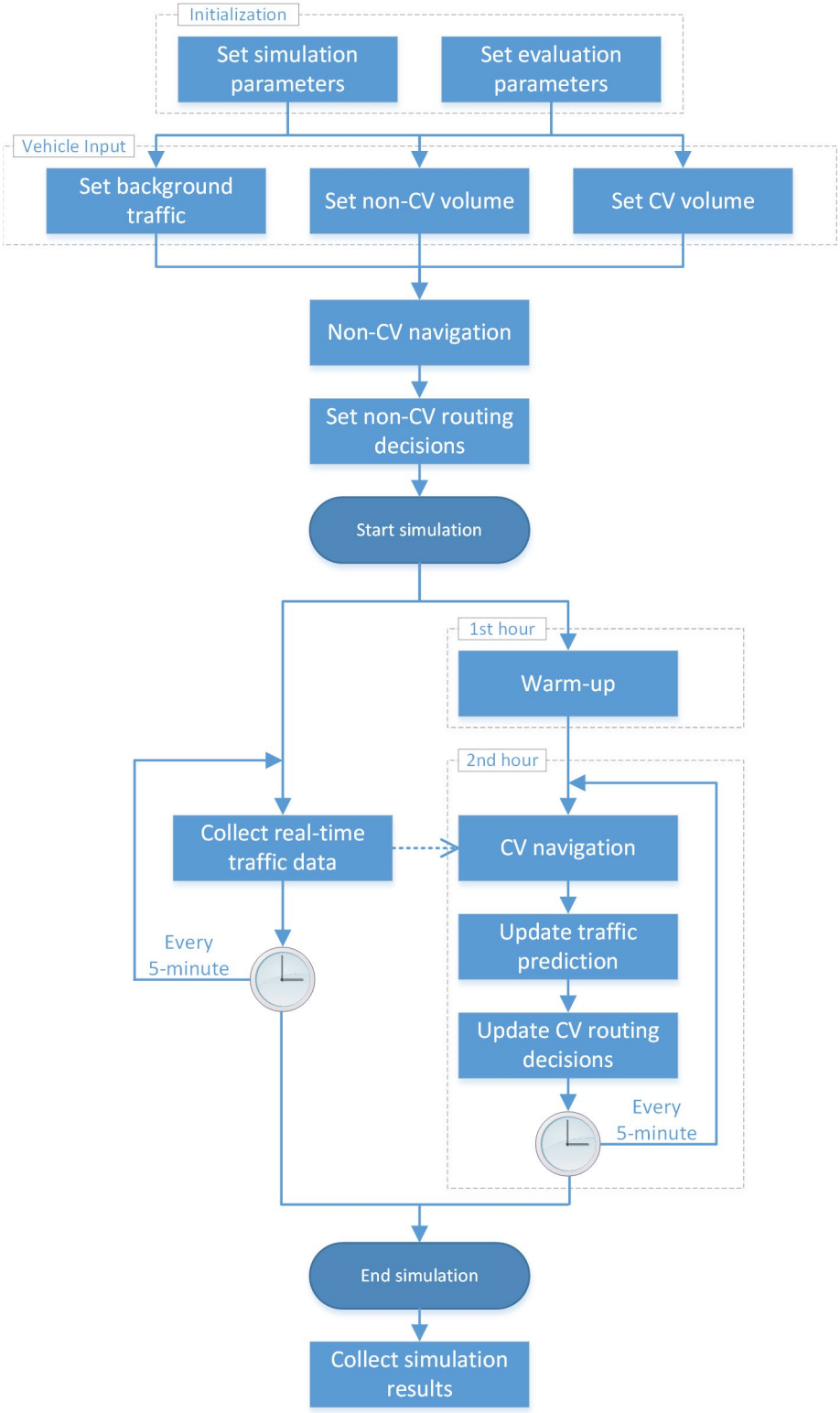


Figure 6.4 Structure of the VISSIM navigation module

Although the vehicle navigation functions could be executed in real time, the execution speed of the navigation module was mainly constrained by the COM objects. Because of the hierarchical structure of the VISSIM COM objects, all objects were sub-objects of the top-level object (i.e., IVissim). Therefore, COM objects had to be accessed or updated one at a time. For example, it was impossible to directly read the network traffic data as a table. Instead, the vehicle count and average travel time for each travel time measurement object had to be read separately. The same procedure also applied when vehicle routing decisions were updated. Therefore, simulation break points at 5-minute intervals were also established to ensure sufficient time for collecting data and updating vehicle routing decisions.

In the simulation experiments, 5 minutes (300 simulation seconds) was selected as the data collection and navigation updating interval. One of the key reasons was to balance the navigation frequency and execution speed. Because of the COM API structure, collecting real-time traffic data and updating vehicle routing decisions typically took a couple minutes for the studied network size. Additionally, aggregate traffic data (i.e., vehicle count and travel time) by 5-minute intervals also helped reduce the impacts of stochastic traffic fluctuations. If the data collection interval was too short, for example, the traffic data would have greater variance. If the data collection interval was too long (e.g., 30 minutes), in contrast, real-time traffic changes might be over-smoothed, and CVs could not change routes adaptively. Therefore, 5 minutes was the preferred interval length, long enough to smooth traffic fluctuations but short enough to still reflect real-time traffic changes.

7. Algorithm Tests

The developed CV navigation algorithms were tested in both numerical and simulation experiments. Specifically, the goals of the algorithm tests included 1) implementing the developed CV navigation algorithms in numerical calculation and microscopic simulation platform; and 2) quantifying navigation impacts under various traffic conditions and different CV penetration rates.

7.1 Experiment Design

The urban road network of downtown Bellevue was used to test the CV navigation algorithms. To evaluate the navigation results under different scenarios, we designed a number of different experiment configurations that considered various real-world traffic cases (e.g., events, non-recurrent congestions, and different CV penetration rates). Additionally, we also evaluated the navigation results in detail under different navigation objectives and algorithm parameters.

7.1.1 Traffic Input

Three types of vehicle input were added to the road network: background traffic, non-CVs, and CVs. The background traffic was set by using the actual hourly traffic count during the midday non-peak period provided by the City of Bellevue. In the experiments, we assumed that vehicles in the background traffic did not change their routes. This part of traffic could be generally considered to be drivers familiar with preferred routes (and therefore not using navigation applications in the area) or commercial vehicles with planned routes. The VISSIM model was first simulated ten times with only the background traffic to develop link cost functions and to summarize the average link traffic volumes that would serve as historical traffic data in the CV navigation algorithms.

Non-CVs and CVs were added to the road network on top of the background traffic. We assumed that this extra traffic flow started from the top-left corner and routed toward the bottom-right corner of the network, so that all road links could potentially be traveled in a candidate path. This experiment design was considered to be a non-recurrent event that generated extra traffic flow in the studied network. Non-CV and CV navigation algorithms were applied to corresponding vehicle classes to compute the optimal path for each vehicle, and we evaluated the user costs associated with the extra traffic flow. Additionally, we tested different flow levels and CV penetration rates to evaluate the navigation results in various scenarios.

7.1.2 Model Parameters

In the simulation experiments, real-time traffic data was collected and aggregated in 5-minute intervals. We used the traffic data from the three most recent time intervals (i.e., the last 15 minutes) as real-time traffic information in the travel time prediction task. Because the historical data were aggregated on an hourly basis, we used 0.25 as the Bayesian parameter to reflect the approximate traffic amount ratio. In further experiments, we also tested different Bayesian parameters, and we will discuss the impacts in some special cases (e.g., non-recurrent events).

7.2 Experiment Results

This section presents the navigation results from both the numerical and simulation experiments. Specifically, the actual path traveled by each vehicle was recorded to understand the network traffic distribution. The network travel times were also collected and summarized by vehicle type by link, in order to evaluate the navigation results.

7.2.1 Network Traffic Distribution

We first looked at network traffic distributions under different navigation algorithms. Figure 7.1 shows the distribution of the extra traffic on all road links when the extra traffic flow rate was 100 veh/h. The first subplot presents the distribution of the extra traffic when all vehicles used the static navigation algorithm (i.e., all vehicles were non-CVs). In this case, all vehicles chose the same path on the basis of the background traffic (i.e., historical traffic data). In this scenario, the extra traffic flow was equivalent to around 3 percent of the total background traffic. Therefore, the network traffic conditions were not greatly influenced; even if all extra vehicles chose the same path. However, this clearly was not the optimal traffic distribution, as multiple alternative paths existed from the origin to the destination. The second subplot presents the network traffic distributions based on the CV navigation algorithm (i.e., the DUE navigation algorithm), assuming that all extra vehicles were CVs. According to the graph, the dynamic navigation algorithm mainly utilized the two fastest paths.

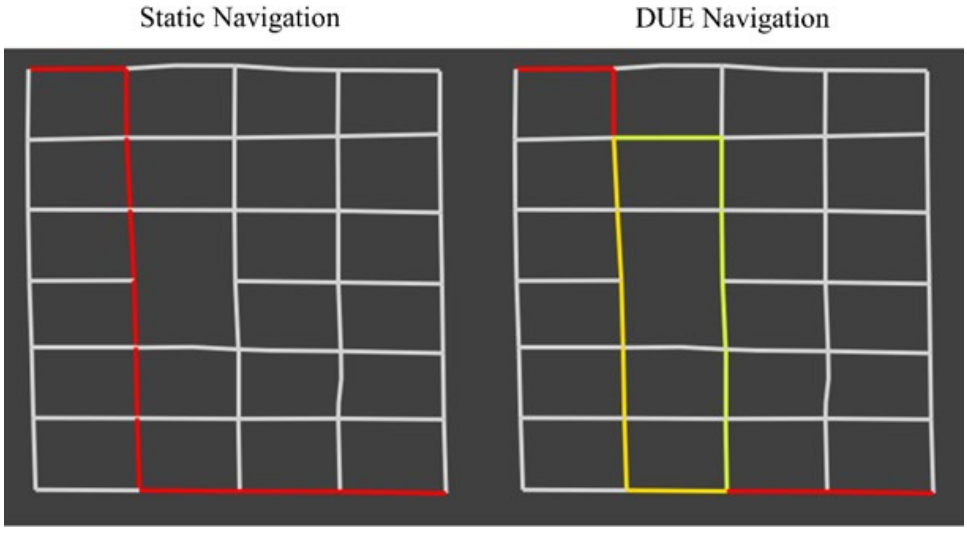


Figure 7.1 Traffic flow distribution for different navigation algorithms (100 veh/h)

Figure 7.2 through figure 7.4 show the network traffic distributions created by further increasing the extra traffic flow to 300, 500, and 800 veh/h, respectively. The static navigation results were the same, as all non-CVs still chose the same path identified by historical traffic data. The dynamic navigation algorithms utilized more roads with an increase in the CV flow rate. When the CV flow rate was 800 veh/h, for example, approximately 80 percent of the roads were traveled by some CVs, and the majority of the utilized roads received around 20 percent to 60 percent of the total extra traffic.

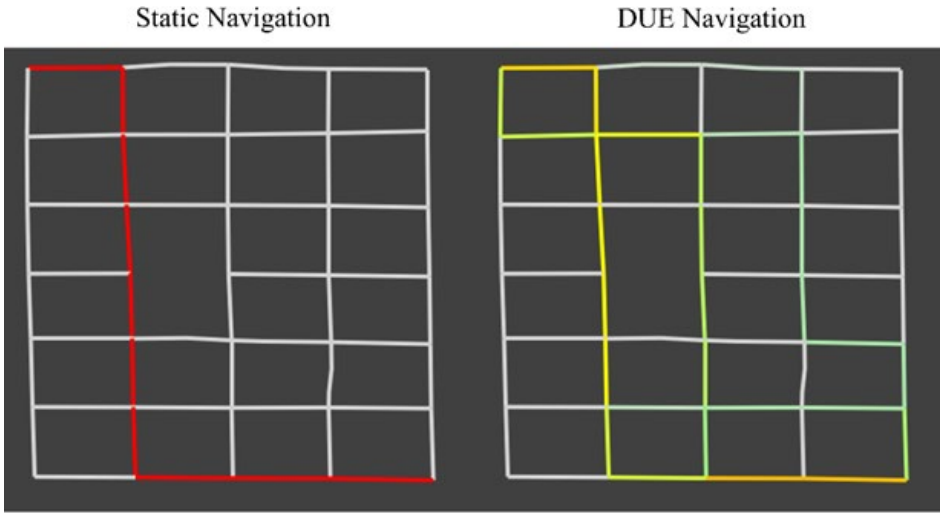


Figure 7.2 Traffic flow distribution for different navigation algorithms (300 veh/h)

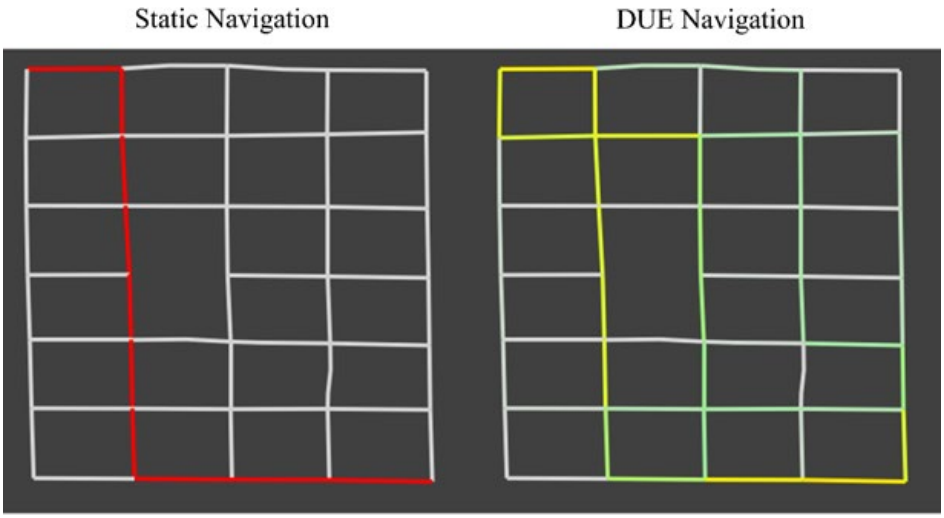


Figure 7.3 Traffic flow distribution for different navigation algorithms (500 veh/h)

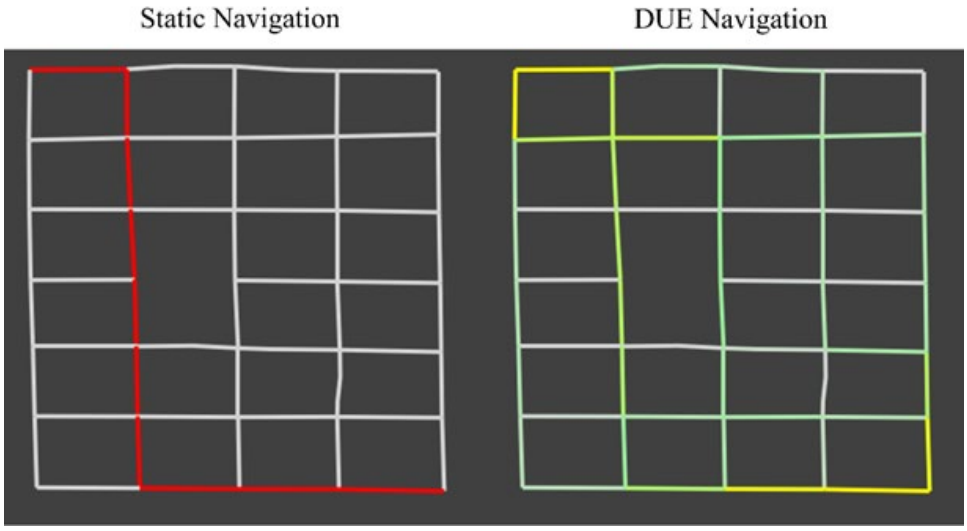


Figure 7.4 Traffic flow distribution for different navigation algorithms (800 veh/h)

To quantify the differences between network traffic distributions, we defined the path difference rate between two navigation algorithms as

$$PathDiff(1,2) = \frac{\sum_E |q_{1,E} - q_{2,E}|}{2\sum_E q_{1,E}} \quad (7.1)$$

where $q_{1,E}$ and $q_{2,E}$ are the link traffic volume under algorithms 1 and 2, respectively. The range of the difference rate was between 0 (same) and 100 percent (completely different). Note that the path difference rate would not change if we switched the places of the two algorithms (i.e., $PathDiff(1,2) = PathDiff(2,1)$). This is because the total traffic volume would be the same when comparing the two algorithms under the same traffic scenario (i.e., $\sum_E q_{1,E} = \sum_E q_{2,E}$, which both equal the extra traffic volume).

Table 7.1 summarizes the path difference rates under different levels of extra traffic flow. According to the table, differences between the static navigation and dynamic navigations increased with the traffic flow rate. This is intuitive, as more roads would be traveled when more CVs were added to the network. Recall that the static navigation algorithm assigns all traffic to

the fastest path identified by historical data. The historical fastest path still played an important role in dynamic navigation algorithms in the sense that it was traveled by around 35 percent of CVs when the CV flow rate was 800 veh/h.

Table 7.1 Path difference rate between navigation algorithms

Extra traffic flow (veh/h)	Path difference rate (static vs. dynamic, %)
100	24.2
200	38.7
300	46.2
400	50.9
500	55.5
600	58.6
700	60.8
800	62.5

It is important to note that the network traffic distributions and path difference rates were calculated from numerical experiments, which did not sufficiently consider traffic randomness in a dynamic environment. In the simulation experiments, the dynamic traffic assignment changed periodically with the real-time traffic data collection (e.g., every 5 minutes). Therefore, the actual network traffic distribution could be slightly different from the numerical results at a specific instant in time. In this section, we discuss the numerical experiment results to help understand the average traffic distribution and expected traffic distributions. Further analysis of the simulation results and discussion about result variations are presented in the following sections.

7.2.2 Impact of CV Penetration

the previous section showed that the dynamic navigation algorithms could distribute CVs on many alternative paths. However, 100 percent CV penetration will not be realistic for some time. Therefore, it was important to test the navigation algorithms under different levels of CV penetration. We further considered the extra traffic as a mixed flow of non-CVs and CVs. While

non-CVs continued to choose the optimal path on the basis of historical knowledge of traffic, CVs dynamically calculated the optimal path using the DUE navigation algorithm. Vehicle input with different CV penetrations were added to the VISSIM model, and the travel time results were collected and analyzed to quantify the impact.

Figure 7.5 shows the relationship between the average path travel time and CV penetration rates at different levels of extra traffic flow. Note that the VISSIM model was simulated ten times with different random seeds for each specific scenario (i.e., as defined by traffic flow level, CV penetration, and dynamic navigation algorithm). Because of the stochastic vehicle input, the percentage of CVs that actually entered the network in each simulation was slightly different, unless the CV penetration was 0 percent or 100 percent. In the graph, the simulation results are plotted on the basis of the detected CV penetration rate. In general, the average path travel time decreased with an increase in CV penetration rate in all scenarios, and the benefit of CVs increased as more traffic was added to the network. Under low or moderate traffic flow levels (i.e., 100 or 300 veh/h), the travel time impact of CV penetration was relatively small (but still observable) in comparison to the result variations. If the vehicles were fully connected (i.e., 100 percent CV penetration), the average path travel time with 500 veh/h traffic flow was reduced to a level similar to that under the minimum traffic flow (i.e., 100 veh/h). This indicates that dynamic navigation algorithms could efficiently assign traffic into different paths and reduce user cost to the minimum level.

When the CV penetration rate was higher than 60 percent, the system did not benefit from further increasing the CV penetration, in the sense that the marginal reduction of user cost was minimal. This was consistent with the numerical experiment results, which found that around 35 percent of CVs would still choose the historical fastest path even if the extra traffic

flow was high. If these 35 percent CVs were changed to non-CVs, their route choice decisions would still be the same, and the network traffic distribution would not be greatly influenced. That said, the mechanisms behind the similar network traffic distributions were different. For example, when the traffic is not fully connected, CVs must rely on traffic detectors to collect non-CV traffic data and then calculate the conditional optimal path. If all vehicles are CVs, they can share route choice decisions in real time without relying on traffic detectors, and dynamic traffic assignment can be achieved faster.

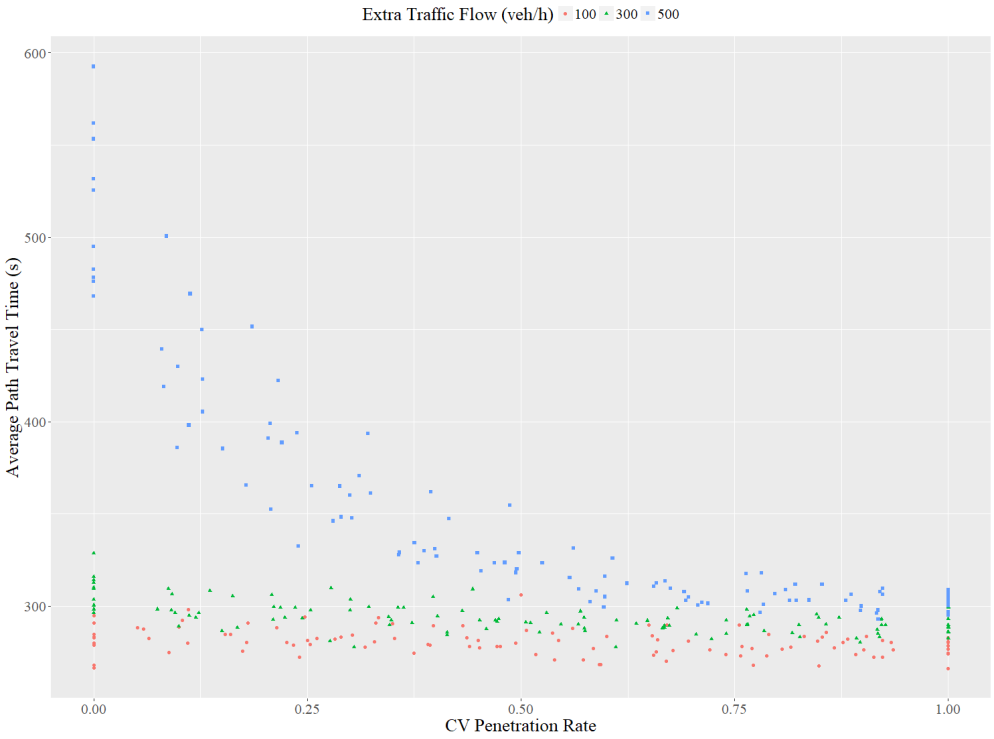


Figure 7.5 Change of user cost with CV penetration rate

8. Conclusions and Recommendations

This study developed a coordinated navigation system that can adaptively navigate CVs with DUE traffic assignment. Different from macroscopic DTA models, the developed navigation algorithms are specifically suitable for urban street networks with intersections. An empirical intersection delay function was developed on the basis of classical probabilistic intersection delay models. The delay function parameters can be empirically derived from actual traffic data collected by detectors at each intersection. On the basis of the intersection delay function, future traffic states can be predicted by combining historical and real-time traffic data. Then CVs will coordinately select different alternative paths through the real-time sharing of their routing decisions.

The developed navigation algorithms were implemented in both numerical and simulation experiments to quantify their impacts. Results showed that in comparison to the static navigation algorithm used by non-CVs, CV navigation algorithms can effectively utilize more roads and achieve a more balanced network traffic distribution. As a result, user costs can be greatly reduced. The developed CV navigation algorithms are helpful in mitigating traffic impacts caused by non-recurrent events.

The benefit of CV navigation algorithms will also increase with an increase in the CV penetration rate. However, in all experiments, the user costs could not be further reduced when the CV penetration reached around 60 percent. This indicates that traffic does not need to be fully connected to achieve the maximum benefits from CV navigation algorithms. Note that a penetration rate threshold was associated with this particular study, as around 35 to 40 percent of CVs would choose the same route as non-CVs (i.e., the fastest route identified by the static navigation algorithm) even if the CV penetration was 100 percent. The maximum beneficial CV

penetration is likely to be different in other cases with different road networks and traffic conditions, and the specific threshold value can be estimated from numerical calculation or simulation experiments.

The dynamic navigation algorithms rely on CV communications (i.e., V2I and V2V communications) to adaptively identify the optimal path on the basis of real-time traffic information and other vehicles' route choice decisions. This study did not specifically consider the quality of CV communications. In practical implementation, however, communication quality such as network latency and transmission quality would be critical to ensure the reliability and effectiveness of the CV navigation algorithms. An interesting research direction would be to quantify the communication load at different parts of the network and optimize the communication system design to ensure the quality of CV communications. Also the current research investigated only user equilibrium (UE) navigation; future work will be conducted for system optimum (SO) traffic assignment and will compare its results with those of the UE approach. Last but not least this study used only simulation data to test the proposed algorithm; more experiments should be conducted in the real world to further validate the algorithm in the future.

References

Bureau of Public Roads, 1990, Traffic assignment manual, U.S. Department of Commerce, 1964.

Chen, C.L.P., Zhou, J., Zhao, W., 2012. A real-time vehicle navigation algorithm in sensor network environments. IEEE Transactions on Intelligent Transportation Systems 13, 1657–1666. https://doi.org/10.1109/TITS.2012.2201478

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271. https://doi.org/10.1007/BF01386390

Eklund, P.W., Kirkby, S., Pollitt, S., 1996. A dynamic multi-source Dijkstra's algorithm for vehicle routing, in: 1996 Australian New Zealand Conference on Intelligent Information Systems. Proceedings. ANZIIS 96. Presented at the 1996 Australian New Zealand Conference on Intelligent Information Systems. Proceedings. ANZIIS 96, pp. 329–333. https://doi.org/10.1109/ANZIIS.1996.573976

Frank, H., 1969. Shortest paths in probabilistic graphs. Operations Research 17, 583–599. https://doi.org/10.1287/opre.17.4.583

Fu, L., 2001. An adaptive routing algorithm for in-vehicle route guidance systems with real-time information. Transportation Research Part B: Methodological 35, 749–765. https://doi.org/10.1016/S0191-2615(00)00019-9

Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics 4, 100–107. https://doi.org/10.1109/TSSC.1968.300136

Hoff, P.D., 2009. A first course in Bayesian statistical methods, Springer texts in statistics. Springer, London ; New York.

Luby, M., Ragde, P., 1989. A bidirectional shortest-path algorithm with good average-case behavior. Algorithmica 4, 551–567. https://doi.org/10.1007/BF01553908

Mahmassani, H.S., 2016. 50th anniversary invited article—autonomous vehicles and connected vehicle systems: flow and operations considerations. Transportation Science 50, 1140–1162. https://doi.org/10.1287/trsc.2016.0712

Mannering, F.L., Washburn, S.S., 2013. Principles of highway engineering and traffic analysis, 5th ed. John Wiley & Sons, Inc.

Möhring, R.H., Schilling, H., Schütz, B., Wagner, D., Willhalm, T., 2007. Partitioning graphs to speedup Dijkstra's algorithm. Journal of Experimental Algorithmics 11, 2.8. https://doi.org/10.1145/1187436.1216585

NOCoE, 2018. SPaT Challenge Overview [WWW Document]. National Operations Center of Excellence. URL https://transportationops.org/spatchallenge (accessed 1.3.19).

NTOC, 2012. National Traffic Signal Report Card: Technical Report. National Transportation Operations Coalition.

PTV, 2016a. Vissim 9 User Manual. PTV GROUP, Karlsruhe, Germany

PTV, 2016b. Vissim 9 Introduction to the COM API. PTV GROUP, Karlsruhe, Germany

TRB, 2010. Highway Capacity Manual 2010. Transportation Research Board.

USDOT, 2018. Connected Vehicle Pilot Deployment Program [WWW Document]. United States Department of Transportation. URL https://www.its.dot.gov/pilots/index.htm (accessed 2.26.18).

USDOT, 2016. Connected Vehicle Basics [WWW Document]. United States Department of Transportation. URL https://www.its.dot.gov/cv_basics/cv_basics_20qs.htm (accessed 2.26.18).

Wagner, D., Willhalm, T., 2007. Speed-up techniques for shortest-path computations, in: In Proceedings of the 24th International Symposium on Theoretical Aspects of Computer Science (Stacs'07. Springer, pp. 23–36.

Webster, F.V., 1958. Traffic signal settings. Road Research Lab Tech Papers /UK/.

Xiao, L., Lo, H.K., 2014. Adaptive Vehicle Navigation With En Route Stochastic Traffic Information. IEEE Transactions on Intelligent Transportation Systems 15, 1900–1912. https://doi.org/10.1109/TITS.2014.2303491

Bibliography

Bellman, R., 1958. On a routing problem. Quarterly of applied mathematics, 16(1), pp.87-90.

Chen, P.-A., Kempe, D., 2008. Altruism, selfishness, and spite in traffic routing, in: Proceedings of the 9th ACM Conference on Electronic Commerce - EC '08. Presented at the the 9th ACM conference, ACM Press, Chicago, Il, USA, p. 140. https://doi.org/10.1145/1386790.1386816

Çolak, S., Lima, A., González, M.C., 2016. Understanding congested travel in urban areas. Nature Communications 7, 10793. https://doi.org/10.1038/ncomms10793

Dantzig, G.B., Ramser, J.H., 1959. The Truck dispatching problem. Management Science 6, 80–91.

FCC, 1999. FCC Allocates Spectrum 5.9 GHz Range for Intelligent Transportation Systems Uses [WWW Document]. Federal Communications Commission. URL https://transition.fcc.gov/Bureaus/Engineering_Technology/News_Releases/1999/nret9006.html (accessed 1.6.19).

Gendreau, M., Guertin, F., Potvin, J.-Y., Taillard, É., 1999. Parallel Tabu search for real-time vehicle routing and dispatching. Transportation Science 33, 381–390. https://doi.org/10.1287/trsc.33.4.381

Ghiani, G., Guerriero, F., Laporte, G., Musmanno, R., 2003. Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. European Journal of Operational Research 151, 1–11. https://doi.org/10.1016/S0377-2217(02)00915-3

Groot, N., De Schutter, B., Hellendoorn, H., 2015. Toward system-optimal routing in traffic networks: a reverse Stackelberg game approach. IEEE Transactions on Intelligent Transportation Systems 16, 29–40. https://doi.org/10.1109/TITS.2014.2322312

Ichoua, S., Gendreau, M., Potvin, J.-Y., 2006. Exploiting knowledge about future demands for real-time vehicle dispatching. Transportation Science 40, 211–225. https://doi.org/10.1287/trsc.1050.0114

Kammoun, H.M., Kallel, I., Casillas, J., Abraham, A., Alimi, A.M., 2014. Adapt-Traf: An adaptive multiagent road traffic management system based on hybrid ant-hierarchical fuzzy model. Transportation Research Part C: Emerging Technologies 42, 147–167. https://doi.org/10.1016/j.trc.2014.03.003

Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L., 2013. A review of dynamic vehicle routing problems. European Journal of Operational Research 225, 1–11. https://doi.org/10.1016/j.ejor.2012.08.015

Powell, W.B., 1996. A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers. Transportation Science 30, 195–219.

SAE, 2006. Dedicated Short Range Communications (DSRC) Message Set Dictionary™. Society of Automotive Engineers. https://doi.org/10.4271/J2735_200612

Secomandi, N., 2000. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. Computers & Operations Research 27, 1201–1225. https://doi.org/10.1016/S0305-0548(99)00146-X

USDOT, 2015. CV Pilot Deployment Program - Connected Vehicle Applications [WWW Document]. United States Department of Transportation. URL https://www.its.dot.gov/pilots/cv_pilot_apps.htm (accessed 1.6.19).