Estimation and Control of Traffic Relying on Vehicular Connectivity

Nianfeng Wan
Clemson University

Follow this and additional works at: https://tigerprints.clemson.edu/all_dissertations

Recommended Citation
Wan, Nianfeng, "Estimation and Control of Traffic Relying on Vehicular Connectivity" (2016). All Dissertations. 1691.
https://tigerprints.clemson.edu/all_dissertations/1691

This Dissertation is brought to you for free and open access by the Dissertations at TigerPrints. It has been accepted for inclusion in All Dissertations by an authorized administrator of TigerPrints. For more information, please contact kokeefe@clemson.edu.
Estimation and Control of Traffic Relying on Vehicular Connectivity

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Mechanical Engineering

by
Nianfeng Wan
May 2016

Accepted by:
Dr. Ardalan Vahidi, Committee Chair
Dr. Mohammed Daqaq
Dr. John R. Wagner
Dr. Mashrur Chowdhury
Abstract

Vehicular traffic flow is essential, yet complicated to analyze. It describes the interplay among vehicles and with the infrastructure. A better understanding of traffic would benefit both individuals and the whole society in terms of improving safety, energy efficiency, and reducing environmental impacts. A large body of research exists on estimation and control of vehicular traffic in which, however, vehicles were assumed not to be able to share information due to the limits of technology. With the development of wireless communication and various sensor devices, Connected Vehicles (CV) are emerging which are able to detect, access, and share information with each other and with the infrastructure in real time. Connected Vehicle Technology (CVT) has been attracting more and more attentions from different fields.

The goal of this dissertation is to develop approaches to estimate and control vehicular traffic as well as individual vehicles relying on CVT. On one hand, CVT significantly enriches the data from individuals and the traffic, which contributes to the accuracy of traffic estimation algorithms. On the other hand, CVT enables communication and information sharing between vehicles and infrastructure, and therefore allows vehicles to achieve better control and/or coordination among themselves and with smart infrastructure.

The first part of this dissertation focused on estimation of traffic on freeways and city streets. We use data available from on road sensors and also from probe
connected vehicles and propose several novel algorithms for estimating statistical features that could represent traffic conditions.

One of the most important traffic performance measures is travel time. However it is affected by various factors, and freeways and arterials have different travel time characteristics. In this dissertation we first propose a stochastic model-based approach to freeway travel-time prediction. The approach uses the Link-Node Cell Transmission Model (LN-CTM) to model traffic and provides a probability distribution for travel time. The probability distribution is generated using a Monte Carlo simulation and an Online Expectation Maximization clustering algorithm. Results show that the approach is able to generate a reasonable multimodal distribution for travel-time.

For arterials, this dissertation presents methods for estimating statistics of travel time by utilizing sparse vehicular probe data. A public data feed from transit buses in the City of San Francisco is used. We divide each link into shorter segments, and propose iterative methods for allocating travel time statistics to each segment. Inspired by K-mean and Expectation Maximization (EM) algorithms, we iteratively update the mean and variance of travel time for each segment based on historical probe data until convergence. Based on segment travel time statistics, we then propose a method to estimate the maximum likelihood trajectory (MLT) of a probe vehicle in between two data updates on arterial roads. The results are compared to high frequency ground truth data in multiple scenarios, which demonstrate the effectiveness of the proposed approach.

The second part of this dissertation emphasize on control approaches enabled by vehicular connectivity. Estimation and prediction of surrounding vehicle behaviors and upcoming traffic makes it possible to improve driving performance. We first propose a Speed Advisory System for arterial roads, which utilizes upcoming traffic
signal information to manage ego-vehicle’s speed so as to reduce fuel consumption. An analytical solution to the optimal control problem is found, and the Singular Arc solution is further discussed. Then we use a powerful traffic micro-simulation tool Paramics, which can simulate a large number of vehicles across a complex traffic network, to analyze the impact of such system on the traffic. Multiple scenarios with different road geometries and penetration level of connected vehicles are presented and analyzed.

We also propose a Predictive Cruise Control systems in which individual vehicles can predict short term future surrounding conditions and react using Model Predictive Control so that the driving performance can be improved. The approach combines two different predictors: A Markov Chain predictor based on driving behavior data and an Expectation Maximization (EM) predictor based on historical traffic data. Predictors are statistically combined and a Model Predictive Control approach is proposed utilizing the predicting probability distribution results to obtain the optimal control actions. Multiple scenarios are presented and results show the improvements in ride comfort and fuel efficiency.
Dedication

I dedicate my dissertation to my family and friends who have helped me in my PhD research.
Acknowledgments

I would like to express my gratitude and thanks to my advisor Dr. Ardalan Vahidi. Dr. Vahidi has been a tremendous advisor during my PhD study, not only because of his invaluable instruction on research, but also his consideration, support and advice. His professional manner and passion in research demonstrates how a research scientist should behave. Comparing with myself 5 years ago, I have gained much experience on critical thinking, scientific reasoning and exploring new research areas, and have grown into a qualified research scientist. Furthermore, I have learned a lot from his academic knowledge. We have completed several challenging yet interesting projects. Without his kind encouragements and detailed guidance, I would have never achieved success through this adventure. Specially I would like to thank his careful revision on the drafts of our publications and this dissertation.

I would also like to thank all my committee members: Dr. Mohammed Daqaq, Dr. John R. Wagner and Dr. Mashrur Chowdhury. Their comments and suggestions are very helpful in completing the dissertation.

I have collaborated with California PATH, University of California Berkeley during the second year of my PhD study, under the supervision of Dr. Roberto Horowitz and Dr. Gabriel Gomes. They have provided all possible resources to support my research, and their comments and suggestions are appreciated.

Thanks also go to Dr. Andre Luckow from BMW Information and Technology
Research Center, who has supported my research through providing fundings as well as valuable insights.

During my Ph.D study, our research team has always been an energetic and supportive group. I would like to thank all group members in the team including Mr. Alireza Fayazi, Mr. Yasha Parvini, Dr. Chen Zhang, Dr. Judhajit Roy, and Mr. Angshuman Goswami. A number of projects in the dissertation are collaborated with one or more of these members. And a lot of ideas are inspired by the group discussions and suggestions.

Last but not the least, I would like to thank my family and friends’ help and support during my Ph.D studies.
# Table of Contents

Title Page ............................................................... i
Abstract ................................................................. ii
Dedication .............................................................. v
Acknowledgments ....................................................... vi
List of Tables .......................................................... x
List of Figures ........................................................ xi

1 Introduction .......................................................... 1
   1.1 Background ....................................................... 1
   1.2 Dissertation Overview ......................................... 7

2 Estimation of Travel Time Distributions for Freeways Using Monte Carlo Simulation ........................................... 8
   2.1 Literature Review ................................................. 8
   2.2 Model Description .............................................. 10
   2.3 Methodology ..................................................... 11
   2.4 Simulation Flow Process ....................................... 19
   2.5 Experimental Setup and Results ............................. 19
   2.6 Conclusion ..................................................... 23

3 Reconstructing Maximum Likelihood Trajectory of Probe Vehicles Between Sparse Updates ............................................. 25
   3.1 Literature Review ............................................... 25
   3.2 Description of the Dataset ..................................... 27
   3.3 Estimation of Segment Travel Time Statistics ............... 29
   3.4 Maximum Likelihood Trajectory Estimation ................. 34
   3.5 Results and Validation in Comparison with Ground Truth Data .................................................... 41
   3.6 Conclusions .................................................... 50
# 4 Optimal Speed Advisory For Connected Vehicles in Arterial Roads and the Impact on Mixed Traffic

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Literature Review</td>
<td>52</td>
</tr>
<tr>
<td>4.2 Optimal Control Formulation of SAS</td>
<td>56</td>
</tr>
<tr>
<td>4.3 The Simulation Environment</td>
<td>70</td>
</tr>
<tr>
<td>4.4 Micro-simulation Case Studies</td>
<td>74</td>
</tr>
<tr>
<td>4.5 Conclusion</td>
<td>81</td>
</tr>
</tbody>
</table>

# 5 Predictive Cruise Control for Autonomous Vehicles: Combining predictors of preceding vehicle’s motion

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1 introduction</td>
<td>83</td>
</tr>
<tr>
<td>5.2 Literature Review</td>
<td>85</td>
</tr>
<tr>
<td>5.3 Chance Constraints Model Predictive Control</td>
<td>87</td>
</tr>
<tr>
<td>5.4 Vehicle Trajectory Prediction Based on Historical Probe Data</td>
<td>91</td>
</tr>
<tr>
<td>5.5 Vehicle Trajectory Prediction Based on Individual Driving Data</td>
<td>94</td>
</tr>
<tr>
<td>5.6 Combination of predictors</td>
<td>95</td>
</tr>
<tr>
<td>5.7 Case Study and Results</td>
<td>99</td>
</tr>
<tr>
<td>5.8 Conclusion</td>
<td>102</td>
</tr>
</tbody>
</table>

# 6 Summary of the Dissertation

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Summary of Contributions</td>
<td>103</td>
</tr>
<tr>
<td>6.2 Dissemination of the Dissertation</td>
<td>104</td>
</tr>
</tbody>
</table>

Bibliography

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Trajectory Estimation Errors. . . . . . . . . . . . . . . . . . . . . . . 50
4.1 Values of vehicle model parameters. . . . . . . . . . . . . . . . . . . 74
# List of Figures

2.1 Cumulative Counts for a Link .................................. 12
2.2 Six Months of Mainline Flow Data, and Their Average ........... 14
2.3 Six Months of On-ramp Flow Data, and Their Average ........... 14
2.4 A Section of I-15 South ........................................ 20
2.5 The Density Contour Plot ...................................... 20
2.6 Deterministic Travel-time ...................................... 20
2.7 Travel-time Distributions and Their Components ............... 21
2.8 Travel-time Distributions and the Sample Histogram .......... 22

3.1 Aggregated plot of transit bus updates for a period of 24 hours in the city of San Francisco [25]. .......................... 28
3.2 A schematic display of segment travel times crowd-sourced from many bus passes. ................................. 30
3.3 Segment travel time and RMS errors evolution of our EM algorithm in [103] ........................................... 34
3.4 A Schematic of queuing and discharging dynamics at a traffic signal. ................................. 38
3.5 A Schematic of queuing and discharging dynamics at multiple intersections. ................................. 41
3.6 Travel time statistics for unstopped trajectories. With respect to the reference point, the Lombard intersection lies at 377 m, and Greenwich intersection lies at 480 m. ................................. 42
3.7 Histogram of position of buses stopped at Lombard intersection as captured from January to September of 2013. ................................. 43
3.8 Estimated and ground truth trajectories spanning a single intersection with no stop. ................................. 44
3.9 Estimated and ground truth trajectories spanning a single intersection with a stop. ................................. 45
3.10 Estimated and ground truth trajectories spanning multiple intersections. ................................. 46
3.11 Comparison of the maximum likelihood trajectory estimation and a naive approach. ................................. 48
3.12 Sensitivity analysis with different headways and queue ends. It is shown that the results are not highly sensitive to headways. The estimation of stop position is sensitive to the queue end. ................................. 49
| 4.1 | Schematics of the speed advisory system. | 59 |
| 4.2 | The optimal strategy alternates between periods of maximum acceleration, engine shut down (or idle), and in some situations a constant speed. | 64 |
| 4.3 | A schematic of accelerate-then-cruise case. | 68 |
| 4.4 | A schematic of glide-then-cruise case. | 69 |
| 4.5 | A screenshot of a Paramics simulation showing SAS equipped vehicles in green and conventional vehicles in white. | 71 |
| 4.6 | Headway and velocity-difference phase space as defined in [85]. | 73 |
| 4.7 | Fuel consumption of vehicles with and without speed advisory system under different traffic demand levels and different penetration levels of equipped vehicles. | 76 |
| 4.8 | Without SAS, vehicles experience more stops at each intersection. | 77 |
| 4.9 | Equipped with SAS, vehicles experience much fewer stops at each intersection and the trajectories are much smoother. | 77 |
| 4.10 | Acceleration of vehicles with and without speed advisory system under 3 different penetration levels of equipped vehicles. The traffic demand was fixed at 600 veh/h/l. Accelerations of all vehicles are plotted in thin blue and the mean acceleration is plotted in thick red. | 78 |
| 4.11 | Velocity of vehicles with and without speed advisory system under 3 different penetration levels of equipped vehicles. The traffic demand was fixed at 600 veh/h/l. Velocities of all vehicles are plotted in thin blue and the mean velocity is plotted in thick red. | 79 |
| 4.12 | Average traffic flow at four different intersections. The traffic demand was fixed at 600 veh/h/l. | 80 |
| 4.13 | Histogram of travel times for conventional and SAS equipped vehicles. The traffic demand was fixed at 600 veh/h/l. | 81 |
| 5.1 | A schematic display of segment travel times crowd-sourced from probe vehicle updates [105]. | 91 |
| 5.2 | Segment travel time and RMS error evolution of our EM algorithm in [103]. | 93 |
| 5.3 | Trajectory prediction schematic. | 94 |
| 5.4 | An example of a position sample histogram using Monte Carlo sampling. | 96 |
| 5.5 | Prediction error samples of the two predictors versus velocity. | 97 |
| 5.6 | Average prediction error of the two predictors versus velocity. | 97 |
| 5.7 | Weights of the predictors over velocity. | 98 |
| 5.8 | An example of ego vehicle trajectories comparison based on different predictors. | 100 |
| 5.9 | An example of ego vehicle trajectories computed based on different predictors. | 100 |
5.10 Average cost value of each predictor ................. 101
Chapter 1

Introduction

1.1 Background

Vehicular traffic is a complex phenomenon due to the interaction of a large number of vehicles, various road geometries and different types of drivers. Vehicular traffic has become one of the essential problems to the society, due to traffic congestion, fuel consumption and accidents. According to data sources from U.S Department of Transportation, the annual societal cost of traffic accident is 299.5 billion U.S dollars, the cost of congestion is 97.7 billion U.S dollars, and on average American drivers spend 40 hours per year in delayed traffic[26]. Insights into how to evaluate traffic conditions, what causes congestion and accidents, and how to improve vehicular mobility to reduce the cost have become critical challenges and have drawn attention from many researchers in different disciplines. A better understanding of traffic would benefit both individuals and the whole society in the sense of safety, energy efficiency, and environmental quality. Attempts to constructing mathematical theories and modeling vehicular traffic flow started back in 1920s. Over decades, a large number of traffic flow theories and modeling approaches have been developed.
One can categorize a large number of vehicular traffic research topics into two groups: estimation and control. Estimation refers to developing methods to comprehend or approximate latent current information based on observations. Also similar to estimation is prediction with the difference that in future conditions are estimated. One can also estimate in a microscopic way, such as estimating individual vehicle behavior. Estimation relies on empirical and real time data obtained from sensors such as loop detectors, cameras, historical driving behavior data storage, etc. Control is to manipulate traffic control devices, such as traffic signals, ramp metering systems, etc., or individual vehicles’ states, such as velocity, acceleration, headings, etc., to improve vehicular movement. More accurate estimation can result in better control decisions and help produce better outcomes.

In most of previous literature, it is assumed that vehicles are not able to communicate with others or infrastructure in real time. Connected Vehicle Technology (CVT) has progressed in a fast pace, in recent years, due to rapid advances in wireless communication and mobile computing technologies. Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication technologies enable vehicles to detect, access, and share information with each other in real time. In 2014, the U.S. Department of Transportation (DOT) issued advance notice of proposed rule-making to begin implementation of Vehicle-to-Vehicle communications technology in new vehicles, indicating that CVT has the ability to improve individual safety and the traffic as well [43]. On one hand, CVT significantly enriches traffic databases. Vehicles sharing their information at random locations and times can be seen as mobile sensors. Therefore it will become possible to generate more accurate estimates based on both fixed and mobile data sources. Moreover, CVT enables cooperation and control in real time through communication, which makes the traffic control strategy more efficient.

With recent development of perception technologies and computation abili-
ties, autonomous vehicle or autonomous driving in urban traffic become realistic. In 2003, the Defense Advance Research Projects Agency (DARPA) announced the Grand Challenge of developing fully autonomous vehicle. In 2007, DARPA held the famous Urban Challenge in which research groups from all over the world demonstrated their autonomous vehicles. In 2010, Google announced its self driving car project [73], and bring the attentions on how artificial intelligence would revolutionize the automotive company. Autonomous vehicles recently released by technological and automotive companies can be anticipated to be mass produced and replace human drivers in the current and next decades. Autonomous vehicles are expected to be well connected, and they require larger amount and more precise ambient information than human driver. The nature of autonomy enables them to better communicate and control. With high quality of vehicular behavior and/or traffic information, and fast decision making and control procedure, autonomous vehicles are expected to achieve better driving performance and fuel economy.

This dissertation proposes several estimation and control approaches based on advances in connected vehicle technologies, such approaches can be implemented in conventional human driving ground vehicles, they can be also adapted to fully autonomous vehicles.

Travel time is among the most important traffic performance measures. Accurate travel-time information enables drivers to understand traffic conditions, and hence to choose routes or manage their trip schedules to avoid congested road sections. Most of today’s state-of-the-art navigation systems such as Google Maps provide travel-time information to their users. From the traffic control prospective, travel-time information also helps in monitoring and controlling traffic with traffic lights, ramp metering devices, etc. [34]. Accurate prediction of travel time for a given route remains a challenging problem, as it is influenced by many different traffic and
road parameters: flow, density, speed, route length, geometry, to name a few. These parameters are obtained through various sources, which carry different kinds of uncertainties. In the first chapter of the dissertation, we present a stochastic model-based approach to freeway travel-time prediction. The approach uses the Link-Node Cell Transmission Model (LN-CTM) to model traffic and provides a probability distribution for travel time along a predetermined route. On-ramp and mainline flow profiles are collected from loop detectors, along with their uncertainties. The probability distribution is generated using Monte Carlo simulations and via an online Expectation Maximization clustering algorithm. The simulation is implemented according to a stopping criterion in order to reduce sample size requirement. A multimodal distribution for travel-time along a given route is the result. Future improvements are also discussed.

The above approach is best suited to highways where loop detectors and traffic cameras can provide the data needed to construct travel time models. In arterial streets, however, there are fewer measuring devices but more complicated environments, because of intersections, stop signs, and pedestrians. An alternative data source inside the cities are Probe Vehicular Data (PVD). Probe vehicles are those equipped with GPS units that periodically broadcast their coordinates to a backend database. Alternatively mobile phones inside a vehicle can be used to broadcast the coordinates of a vehicle. One can get travel time measurement directly from probe vehicle data. In order to get accurate estimates of the overall traffic conditions, higher penetration rate of probe vehicles and/or high probe vehicle reporting frequency are needed. Unfortunately today, the penetration rate of connected vehicles that can share their GPS coordinates is low. Moreover most probe vehicle data updates are spatiotemporally sparse. Besides, even if technically possible, high reporting frequency raises privacy concerns for the owner of such vehicles. Different methods have
been proposed in the literature to solve some of these issues [38][37]. Some other papers propose to combine probe vehicle data with loop detector data to estimate travel times [6]. Travel time is often chosen to estimate other parameters and/or describe traffic conditions when using probe data, since it is a direct observation and two successive probe updates provide one travel time observation.

The second chapter of the dissertation presents methods for estimating statistics of travel time in arterial roads by utilizing sparse vehicular probe data. We use a public data feed from transit buses in the City of San Francisco as an example data source. Sparsity of time and location updates along with frequent stops, at bus stops and traffic lights, complicates estimation of travel time for each link based on a single bus pass. Unlike most previous papers that focus on estimation of link travel times, we divide each link into shorter segments, and propose two iterative methods for allocating travel time statistics to each segment. Inspired by K-mean and Expectation Maximization (EM) algorithms, we iteratively update the mean and variance of travel time for each segment based on historical probe data. Our preliminary results show convergence to reasonable travel time patterns; for instance they clearly reveal the location of bus stops and traffic signals and statistics of delay across them. Applications of this work are in better traveler information systems and in estimation of maximum likelihood trajectory of vehicles in arterial roads, the subject of the third part of the completed work.

Due to the low reporting frequency, current probe data only provide sparse discrete samples of the whole traffic. If we can reconstruct vehicle trajectories based on these sparse updates, we can track each vehicle more accurately and furthermore understand the entire traffic better. We then propose an approach to devise a method to reconstruct vehicle trajectories between probe updates based on historical data and current traffic information, such as traffic signal timing. Travel time statistics
along the road and queue patterns at intersections are learned from historical data. The path is divided into short segments, and an Expectation Maximization (EM) algorithm is proposed for iteratively allocating travel time and delay to each segment. Then the trajectory with the maximum likelihood is generated based on travel time and delay statistics. The results are compared with high frequency ground truth data in multiple scenarios, which demonstrate the effectiveness and accuracy of the proposed approach.

The rest of the dissertation focuses on control strategies that enhance the performance of individual connected vehicles and the traffic around them. A major direction for connected vehicle is V2I communication based control strategy. A Speed Advisory System (SAS) that aids in reducing idling near traffic signals is one of the applications of CV technology. Connected Vehicles (CV) equipped with a Speed Advisory System (SAS) can obtain and utilize upcoming traffic signal information to manage their speed in advance, lower fuel consumption, and improve ride comfort by reducing idling at red lights. The motivation of our work is to demonstrate the optimal fuel minimal driving strategy. In the following chapter, we propose a SAS, which is a systematic utilization of Pontryagin Minimum Principle and Bellman’s principle of optimality, and the analytical solution is provided. After presenting this analytical solution to the fuel minimization problem, we employ a sub-optimal solution such that drivability is not sacrificed and show fuel economy still improves significantly. Moreover we further evaluate the influence of vehicles with SAS on the entire arterial traffic in micro-simulations. The results show that SAS-equipped vehicles not only improve their own fuel economy, but also benefit other conventional vehicles and the fleet fuel consumption decreases with the increment of percentage of SAS-equipped vehicles. We show that this improvement in fuel economy is achieved with a little compromise in average traffic flow and travel time.
We also propose a Predictive Cruise Control approach which utilizes the combination of prediction results to control the ego vehicle and achieve better fuel economy and driving comfort. Cruise control is one of the essential functions for both conventional vehicles and autonomous vehicles. Adaptive Cruise Control (ACC) is the next generation but it only uses instantaneous measurements. Predictive Cruise Control, however, utilizes historical information to predict surrounding vehicles behavior as well as upcoming traffic situation, which enables the ego vehicle to react in advance therefore improve the driving smoothness and reduce fuel consumption. Prediction is essential yet containing uncertainties. We proposed a Model Predictive Control with probabilistic constraints which is capable of utilizing chance constraints. We have also demonstrated that the combination of two predictors, one predictor based on individual driving behavior data, one predictor based on historical probe vehicle data, perform better than either one of them. Different scenarios show the effectiveness of our approach. Future directions are also discussed.

1.2 Dissertation Overview

The rest of the dissertation is organized as follows: Chapter 2 presents estimation of travel time distributions for freeways using Monte Carlo simulation. Chapter 3 presents an approach to reconstruct most likely vehicle trajectories in arterial roads using sparse transit bus data. Chapter 4 introduced the Speed Advisory System and the evaluation of its influence in mixed traffic conditions. Chapter 5 introduces a Predictive Cruise Control algorithm with probabilistic constraints. Chapter 6 discuss the conclusions and novel contributions of this dissertation.
Chapter 2

Estimation of Travel Time
Distributions for Freeways Using
Monte Carlo Simulation

2.1 Literature Review

Most existing methods for computing travel-times rely on data-mining from historical data. Those include methods based on linear regression [90], time series analysis [8], Kalman filtering [110] and [13], and artificial neural networks [86]. Such data-based prediction methods require large amount of traffic data with gaps and inaccuracies due to missing or bad sensors. Instead of large investments in fixing the sensor network, traffic flow models can be built as an alternative mean, to fill the data gap and for obtaining travel-time forecasts through simulation. Model-based prediction does not depend as much on real time measurements as do data-based techniques; but models can be re-calibrated on the fly when new data becomes available. Several current model-based prediction methods are based on microscopic simulation [12]
and [40], which model the behavior of each individual vehicle. As compared with macroscopic models, microscopic models are computationally more demanding and are often difficult to calibrate. In [109] and [47], the authors used macroscopic models to estimate speed of traffic and thus to predict travel time. However the estimation or prediction based on link speed is not accurate, especially when congestion happens.

Some authors propose traffic simulators which include both microscopic and macroscopic models, such as [5] and [15]. These, however, do not consider demand uncertainties in a statistical sense.

To overcome these challenges, we propose in this chapter to estimate travel times using a macroscopic model, which formulates the relationships among aggregate traffic quantities. Because the parameters and inputs of the model are influenced by various sources of uncertainty, it makes more sense to estimate a probability distribution for travel-time rather than a single deterministic value. A single travel-time sample along a route is usually not helpful, since it does not provide a sense of the reliability of the information. Instead a travel-time probability distribution has important uses in traveler information as well as in traffic control systems. To the authors’ best knowledge, there are no literature aiming at travel-time distribution prediction based on macroscopic model.

When estimating a probability distribution for travel-time, a challenge is the multidimensionality of the problem. Travel-time is affected by various factors, each of which may have a different kind of distribution (Gaussian, uniform, etc.), and small changes of the parameters may significantly alter the outcome. Because of the nonlinearity of the traffic model, real travel-time distributions often present multiple modes, and may be sensitive to the inputs. Finally there is the challenge of the finite availability of computation time and memory. In this chapter we employ Monte Carlo simulations for generating travel-time samples under demand uncertainties. A
probability distribution for travel-time is then adaptively generated, via the Online Expectation Maximization clustering method [94], as new samples become available from Monte Carlo simulations. A stopping criterion for the sampling process is also introduced in order to reduce sample size and computational requirement.

2.2 Model Description

This work uses BeATS [34] (Berkeley Advanced Transportation Simulator) as the traffic simulator. BeATS is an implementation of the Link Node Cell Transmission Model (LN-CTM), described in [80].

The LN-CTM is a macroscopic model of traffic suitable both for freeways and arterials. It is an extension of CTM [16] which simulates traffic behavior specified by volume (flow) and density. In LN-CTM, the traffic network is modeled as a directed graph. Links represent road segments and nodes are road junctions. Source links introduce traffic to the network and sink links absorb traffic. The fundamental diagram, a diagram relating densities to flows, is used to specify the parameters of each link. A split-ratio matrix at each node defines how vehicles are directed from input to output links. The required data can be obtained from the Performance Measurement Systems (PeMS): an online repository, which provides a rich archive of sensor detector data for freeways in California.

In general, the LN-CTM requires mainline and on-ramp demand profiles, calibrated fundamental diagrams and split ratio matrices as inputs. The model can be calibrated to match actual observation results[20].
2.3 Methodology

2.3.1 Travel Time Calculation

In microscopic simulation, one can track individual vehicles to estimate travel-time. Macroscopic models, because they compute only aggregate quantities, cannot provide direct estimates of travel-time for individual travelers. They are better suited, however, for estimating the probabilistic characteristics of travel-time.

We next describe the technique for calculating travel-time for a driver starting a trip at time $t_{\text{start}}$ and traveling over a route $R$.

The route $R$ is composed of a sequence of links $\{r_i\}, i = 1, 2, \ldots n$. The driver starts at the beginning of link $r_1$ at time $t = t_{\text{start}}$. The objective is to find the time $t_{\text{end}}$ when the driver will exit link $r_n$ as a function of the history of macroscopic flows and densities along the route. Then a sample of travel-time for route $R$ at time $t_{\text{start}}$ is:

$$TT(R, t_{\text{start}}) = t_{\text{end}} - t_{\text{start}}$$ (2.1)

The process can be repeated over an ensemble of simulations to obtain the distribution of $TT(R, t_{\text{start}})$. The steps below are followed to obtain the distribution of travel-time:

1. Initialization: $\rho_i(t_{\text{start}})$, the initial state of each link $i$ at time $t_{\text{start}}$, must be computed, using either a state estimator or by advancing the simulator from a previously known state. For the purpose of this work, the simulation was started with an empty initial condition at midnight and advanced deterministically to the starting time. Thus, the ensemble of runs was given a deterministic initial condition at time $t = t_{\text{start}}$. 

11
2. Take the $i$th link in the route and denote its incoming and outgoing flow with $f_{in}^i(k)$ and $f_{out}^i(k)$ at time step $k$. Then the “cumulative counts” into the link $N_{in}^i(k)$ and those out of the link $N_{out}^i(k)$ are:

$$N_{in}^i(k) = \sum_{\alpha=0}^{k} f_{in}^i(\alpha) \cdot \Delta t + N_{in}^i(0)$$ (2.2)

$$N_{out}^i(k) = \sum_{\alpha=0}^{k} f_{out}^i(\alpha) \cdot \Delta t + N_{out}^i(0)$$ (2.3)

where $\Delta t$ is the length of time step, $N_{in}^i(k)$ and $N_{out}^i(k)$ are in vehicle units.

Since the flows are non-negative, the cumulative counts are non-decreasing functions of time. They are shown in Figure 2.1 for a particular link. The travel-time in the link is the time it takes for the output flow to accumulate the total number of vehicles present in the link when the vehicle entered. Thus the travel-time $\tau$ is the solution to the following equation:

$$\rho_i(t_{in}) \cdot l_i = N_{out}^i(t_{in} + \tau) - N_{out}^i(t_{in})$$ (2.4)

where $t_{in}$ is the time when the vehicle entered the link, $\rho_i(t_{in})$ is the initial density
at time $t_{in}$, and $l_i$ is the length of the link $i$. Equation 2.4 is solved numerically by searching the $N_{out}^i(k)$ vector for the value $\rho_i(t_{in}) \cdot l_i + N_{out}^i(t_{in})$. The only subtlety that arises is that the initial time $t_{in}$ and/or the final time $t_{in} + \tau$ may not fall on the time grid. In this case we also count $N_{in}^i$ and use linear interpolation to calculate accurate $t_{in}$.

The computation of travel-time on a route is performed by computing travel times on each link of the route in sequence, and noting that the exit time for link $i$ is the entering time for link $i + 1$.

2.3.2 Monte Carlo Sampling

As mentioned before, travel-time is affected by factors such as capacity and demand. Considering that they are themselves non-Gaussian random quantities, and the system is inherently nonlinear, the travel-time estimation problem becomes analytically intractable. Therefore, in this work, the Monte Carlo method is chosen to obtain the travel-time results.

The Monte Carlo method samples randomly from a probability distribution. It approximates the distribution when it is infeasible to apply a deterministic method. The number of samples needed in Monte Carlo does not depend on the dimension of the problem, making it suitable for solving multidimensional problems. Another feature of Monte Carlo is that it is easy to estimate the order of magnitude of statistical error[57].

In this work, the uncertainties are added to the mainline and on-ramp demand profiles. The on-ramp uncertainty is assumed to be Gaussian, and the deviation is generally on the order of 5% of the demand in the morning rush hour. Mainline demands are also considered as a Gaussian distribution, and the reasonable deviation is
around 2.5%. Figures 2.2 and 2.3 show six months of loop detector readings gathered every five minutes from detectors on I-15 in California. These plots illustrate the typical variations in 5-minutes average flows.

With each simulation, the Monte Carlo method randomly samples from the demand distributions. Each simulation generates one sample travel-time. Because each simulation requires considerable computation and execution time, it is important to estimate how many samples will be needed to produce statements about the travel-time distribution with a given level of confidence. Also, it will be important to parameterize the distribution in a way that captures its important features, while using a relatively small number of parameters.
2.3.3 Clustering via the Expectation Maximization Algorithm and Bayesian Inference Criterion

The shape of the travel-time probability distributions is not known a-priori. Based on current literature\cite{56}, the travel-time distribution can be represented as a Gaussian Mixture Model (GMM). GMM represents the data as a sum of several Gaussian distributions. The probability density distribution is then:

\[
P(x|\pi, \mu, \Sigma) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i)
\]  

(2.5)

where \( P \) is the probability density function, \( \mathcal{N} \) is the Gaussian distribution, \( K \) is the number of the components or clusters, \( \mu_i \) is the mean, \( \Sigma_i \) is the covariance matrix, and \( \pi_i \) is the weight. The weights are such that,

\[
\sum_{i=1}^{K} \pi_i = 1
\]

(2.6)

We use a clustering technique to find out the values of the parameters from a group of samples. The Expectation Maximization (EM) method was chosen here for clustering the data for GMM\cite{19}. The EM method starts with a random guess of the unknown parameters, and iteratively alternates between an expectation (E) step and a maximization (M) step. The E step produces the responsibilities \( \{\gamma_i(x)\} \), \( i = 1, 2, ..., K \), where \( \gamma_i(x) \) represents the conditional probability that the data \( x \) came from the \( i \)th cluster, given the current parameters \( \{\mu_i, \Sigma_i, \pi_i\} \). That is,

\[
\gamma_i(x) = \frac{\pi_i \mathcal{N}(x|\mu_i, \Sigma_i)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i)}
\]

(2.7)

The M step updates the parameters \( \{\mu_i, \Sigma_i, \pi_i\} \) to maximize the expectation
of the log-likelihood. The parameters are updated with,

\[ \mu_i = \frac{\sum_{j=1}^{N} \gamma_i(x_j) \cdot x_j}{\sum_{j=1}^{N} \gamma_i(x_j)} \]  

(2.8)

\[ \Sigma_i = \frac{\sum_{j=1}^{N} \gamma_i(x_j) \cdot (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{N} \gamma_i(x_j)} \]  

(2.9)

\[ \pi_i = \frac{1}{N} \sum_{j=1}^{N} \gamma_i(x_j) \]  

(2.10)

where \( N \) is the number of data points.

By iterating sufficiently between the E step and the M step, the parameters can converge. However, EM is not guaranteed to converge to a global maximum of the log-likelihood function. In this work, we initiate several different random guesses to avoid getting stuck in local maxima[76].

Another important point is that the number of clusters in the distribution is unknown. This work uses a Bayesian Inference Criterion (BIC) to estimate the optimal number of clusters[91]. The BIC criterion can be represented as:

\[ BIC = -\ln P(D|\mu, \Sigma) + \frac{KQ + 1}{2} \ln N \]  

(2.11)

where \( \ln P(D|\mu, \Sigma) \) is the log-likelihood function, \( D \) represents the samples, \( Q \) is the number degrees of freedom (here since the travel-time has one degree of freedom, \( Q = 1 \)), \( K \) is the number of clusters, and \( N \) is the sample size. The optimal cluster number would generate the maximum BIC value. With different initial conditions, BIC may converge to different optimal numbers of clusters. In this work, we choose
the most frequent result as the optimal one.

2.3.4 Online EM Algorithm

Statistically, in Monte Carlo sampling, more samples provide more accurate results. However, the computation time of the simulation has a linear relation with the number of samples. The more it samples, the longer time it requires. If the prediction time is too long, the traffic condition may significantly change, and the “delayed” prediction is less reliable. Moreover, for the method to be applicable to real-time systems, it must be capable of handling streaming data. That is, given a new data packet (30 samples, for example), the clustering method should be able to use that to update a running estimate. It stops requesting new data only if the results meet a stopping criterion. The advantage of this “data stream” structure is that it minimizes the number of simulations as well as the amount of memory needed to store the samples. Both of these are essential requirements for travel advisory as well as traffic management systems.

Since the target distribution is assumed to be GMM, the Online Expectation Maximization (Online EM) method[94] is suitable for clustering the data. In this work, the Online EM method applies EM only to newly arrived data rather than to the whole historical data. And the incremental GMM estimation algorithm merges Gaussian components that are statistically equivalent, and maintains other components.

The W statistic test is used for equality of covariance. Let the newly coming samples $x_i$ with $i = 1, 2, .., n$ have a covariance matrix $\Sigma_x$, and a given target covariance matrix $\Sigma_0$. The null hypothesis is $\Sigma_x = \Sigma_0$. Define $L_0$ as a lower triangular matrix by Cholesky decomposition of $\Sigma_0$, that is, $\Sigma_0 = L_0 L_0^T$. Let $y_i = L_0^{-1} x_i$, $i = 1, 2, .., n$.
1, 2, ..., n, then the W statistic is represented as:

\[
W = \frac{1}{d} \text{tr}[(S_y - I)^2] - \frac{d}{n} \left[ \frac{1}{d} \text{tr}(S_y) \right]^2 + \frac{d}{n}
\]  

(2.12)

Where \( S_y \) is covariance of \( y_i \), \( d \) is the dimension, \( n \) is the sample size, and \( \text{tr}(\cdot) \) is the trace of the matrix. From [60], \( \frac{nWd}{2} \) has \( \chi^2 \) distribution, that is:

\[
\frac{nWd}{2} \sim \chi_{d(d+1)/2}^2
\]  

(2.13)

Once we set a significance value for the \( \chi^2 \) distribution, we can decide whether the test has passed or failed.

The Hotellings \( T^2 \) statistic is used for equality of mean. Let the newly coming samples \( x_i, i = 1, 2, ..., n \) have a mean \( \mu_x \), and a given target mean \( \mu_0 \). The \( T^2 \) is defined as:

\[
T^2 = n(\mu_x - \mu_0)^T S^{-1}(\mu_x - \mu_0)
\]  

(2.14)

Where \( S \) is covariance of \( x_i \). From [42], \( \frac{n-d}{d(n-1)} T^2 \) has \( F \) distribution, that is:

\[
\frac{n-d}{d(n-1)} T^2 \sim F_{d,n-d}
\]  

(2.15)

Once we set a significance value for the \( F \) distribution, we can decide whether the test has passed or failed.

Since it is essential to stop the Monte Carlo simulation properly to avoid too many samples, several rules are added:

1. When a cluster can be merged with more than one other clusters, the one with the highest weight is chosen;
2. A cluster is eliminated whenever its weight falls below a threshold;
3. The clustering algorithm is stopped if a certain number of iterations pass without new clusters being created.

2.4 Simulation Flow Process

In summary, the process of generating a distribution of travel-time from stochastic simulation is as follows:
1. The simulator advances to the given starting time.
2. (Monte Carlo step) The simulator applies uncertainties to the model, and runs a certain number of times to get travel-time samples. Each travel-time is calculated through the method mentioned at the start of this section.
3. (Online EM step) The simulator clusters the incoming samples, and calculates the parameters using the EM algorithm.
4. (Merging or Maintaining step) The simulator merges qualified new clusters to the old ones, and maintains the rest.
5. (Eliminating step) If the clusters have less weight than a threshold, the simulator merges them into the nearest cluster.
6. If there is no more new cluster in several steps, the simulation is stopped and returns the parameters, otherwise step 2-6 are repeated.

2.5 Experimental Setup and Results

A section of I-15 southbound was used to test the algorithm. The stretch is located between Escondido and San Diego in California. Figure 2.4 shows the section in Google Maps. The section contains more than 120 nodes and 100 links. A route
with 9 consecutive links was created. The total length of the route is 1.47 miles. It contains two on-ramps and no off-ramps. The demand profiles and the split ratio data
Figure 2.7: Travel-time Distributions and Their Components

was obtained from PeMs for Monday, January 7, 2013.

Figure 2.5 shows the density contour plot of one simulation sample. The vertical axis is the time, the horizontal axis is the spatial dimension, and the color represents the amount of density on each link. The stretch over which travel-time was computed is highlighted. The contour plot shows that on the route the congestion begins around 7:15 AM and ends at about 9:45 AM, which illustrates the Monday morning rush hour on I-15. With reasonable uncertainties, the boundary of the congestion changes and the travel-time changes too.

The travel-time curve resulting form a simulation with mean values of demands is shown in Figure 2.6.

The significance level for the Online EM algorithm was set to 0.05. The threshold for eliminating cluster was set to 0.05, and the number of steps for convergence to 3. Two starting times were selected. The first one is 8:00 AM. At this time, the route is heavily congested. Figure 2.7a shows the results. Although the distribution seems to be a Gaussian distribution, the Online EM clustering algorithm found two clusters. The GMM distribution result is more accurate. In this heavily congested example, the travel-time distribution is unimodal. The second starting time is 7:20 AM. At this time, the route is on the edge of congestion. With variable uncertainties,
(a) The first case (8:00 AM)  
(b) The second case (7:20 AM)

Figure 2.8: Travel-time Distributions and the Sample Histogram

in some cases it is congested while in others it is in freeflow. Since vehicles cannot travel faster than freeflow speed, the minimum travel-time is the freeflow speed travel-time, which is 69 seconds. Figure 2.7b shows that there are two well separated modes in the distribution. In the first mode, the travel-time stays around freeflow speed travel-time, which indicates that there is no congestion or only a small number of links in the route are congested while others are in freeflow. In the second mode, more links become congested. In that case the travel-time distribution represents multiple modes.

By using the On-line EM algorithm, the clusters are eliminated, merged, or maintained and after several steps, the clusters number and parameters become stable. In the first case, the simulation stopped with 170 samples. In the second one, 290 samples were needed. Figure 2.8 compares the travel-time distribution prediction results with the histogram of 1000 travel-time samples, which more precisely capture the shape of the distribution. The comparison shows good agreement, suggesting that the Online EM algorithm and the stopping criterion work well and require fewer samples.

Travel-time distributions can be used by drivers and traffic managers to make
more informed decisions about expected traffic patterns. For example, from Figure 2.7, drivers could expect with a high degree of certainty to take between 470 seconds and 490 seconds to travel the given route if they start at 8:00 AM. On the other hand, they will be aware that at 7:20 AM the situation is less reliable and a wider range of outcomes are possible.

2.6 Conclusion

This work introduced a model-based approach for predicting probability distribution of travel-times for freeways. We used the BeATS simulator, an implementation of the Link-Node Cell Transmission Model, to model vehicular traffic flow and to estimate link travel times for a given set of demand profiles. Uncertainty in demand was handled by Monte Carlo sampling and an Online Expectation Maximization algorithm that estimated a Gaussian mixture probability distribution for travel times. Simulations with data from a freeway in California showed that the method could provide a robust estimate of travel time probability distribution with a relatively small number of simulations.

The proposed approach is not only suitable for freeways, but also applicable to arterial roads. Travel time for arterial roads is expected to have a multi-modal distribution due to the stop-and-go pattern induced by traffic signals at intersections. Therefore optimizing the sampling process becomes more critical for arterial roads and is the subject of future work.

Furthermore, the mainline and on-ramp demand is assumed to have Gaussian distribution in this work. Also we assume that the demands themselves are not correlated. In reality it is not always the case. In our future work, demands patterns will be investigate more accurately, and auto-correlation between demands will be
taken into account, and the modeling method will be improved if necessary.

Our model-based travel-time estimator runs open-loop and therefore is sensitive to accuracy of its parameters. Combining model predictions and sparse estimates of travel times from probe vehicle data, in a closed-loop estimator, can generate more accurate estimates of travel-time distribution, and will be investigated in our future work.
Chapter 3

Reconstructing Maximum Likelihood Trajectory of Probe Vehicles Between Sparse Updates

3.1 Literature Review

Nowadays the technologies of wireless communication and cloud storage enable collection of probe data more efficiently. But existing probe data sets are spatiotemporally sparse. Higher penetration rate of probe vehicles and higher reporting frequencies are needed for accurate traffic estimation or vehicle control purposes, but are unlikely in the near future. High reporting frequency also raises privacy concerns [38, 39]. Moreover, current available high frequency vehicle trajectory data usually contains large measurement errors and other errors as well. Researchers propose several methods to refine trajectories [79] [88]. On the other hand, due to their low reporting frequency, current probe data, in raw form, can provide only a very incomplete picture of traffic on the road. But if vehicle trajectories between two sparse
updates could be effectively reconstructed, additional virtual data points are generated that perhaps help more accurate evaluation of road traffic. One application is shown in [68] where the authors use high frequency data to conduct traffic shockwave analysis. In this chapter we propose a method for reconstruction of vehicle trajectories between probe updates based on historical data and current traffic information such as signal timing.

There are a number of papers that address vehicular trajectory reconstruction [44, 75, 82, 97, 28]. For example in [44], the authors proposed a method for short term prediction of a vehicle trajectory based on the usage of nearby vehicles information. In [97] trajectories are reconstructed based on the variational formulation of kinematic waves, and results have been tested with NGSIM data and microsimulation data. In [28], the authors focus on finding the most likely driving mode sequences (deceleration, idle, acceleration, and cruise), to estimate a vehicle’s trajectory. The approach in [97] is macroscopic while [28] employ a microscopic-based approach.

In this chapter we employ a different probabilistic microscopic-based approach, and use segment travel time statistics to reconstruct vehicle trajectories. Using historic probe data, we first estimate the travel time statistics across short road segments and also queue patterns at intersections. We then employ a maximum likelihood approach to generate the most likely trajectory of a probe vehicle between consecutive GPS updates.

Because travel time is an important measure of traffic conditions, a number of papers have focused on travel time estimation or prediction for freeways [102] and arterials [36, 96, 31, 112]. An example of a probabilistic approach to travel time estimation can be found in [35]. In [14] it is shown that travel time can be used in estimating vehicle trajectories.

Most of these existing papers estimate link travel times; the limitation is the
implicit assumption of uniform distribution of travel time along an entire link. However, travel time along an arterial road may not be uniformly distributed, and for instance is higher near signalized intersections. To capture this variability, in [103], we proposed to divide each link to short segments of equal length and estimated statistics of travel time for each segment by using an Expectation Maximization algorithm.

In this chapter we build on our earlier results reported in [103, 104] to estimate the most likely trajectory of a probe vehicle between two consecutive updates. Presence of signalized intersections between the two updates complicates the problem and is influenced by factors such as queue size, signal timing and phase [67, 29]. In this paper we present solutions for when the updates span a single or multiple intersections. We evaluate the proposed algorithms using sparse updates from transit buses in the city of San Francisco. We demonstrate the effectiveness of the proposed method in comparison with high frequency data obtained from ground truth measurements of those same buses.

The rest of the chapter is organized as follows: Section 3.2 describes the bus data feed. Section 3.3 explains estimation of segmental travel time statistics. Section 3.4 outlines our proposed method for estimating the most likely trajectory of a vehicle including the cases where the vehicle comes to a stop at an intersection queue. Section 3.5 presents the results as compared to ground truth measurements followed by conclusions in Section 3.6.

### 3.2 Description of the Dataset

In this chapter, we use a public data feed of transit buses in the city of San Francisco. The data contains GPS time stamp, longitude and latitude, velocity,
heading and several other attributes of transit buses and is provided by NextBus. NextBus provides real-time passenger information for over 135 transit agencies and organizations in North America [81]. The data can be queried in almost real-time in eXtensible Markup Language (XML) interface using URLs with parameters specified in the query string.

Figure 3.1 shows aggregated GPS updates from all buses in the city of San Francisco within a twenty-four hour period. As shown in the figure the coverage includes most major streets. Also the fact that buses traverse each route regularly is an advantage of using them as probe vehicles. However, these updates are sparse; at every 200 meters or 90 seconds whichever comes first. Moreover, buses stop not only at intersections but also at bus stops which complicates the trajectory estimation problem. The bus data was among the very few publicly available data feeds that we could find and therefore was used to verify the effectiveness of our proposed algorithms.
3.3 Estimation of Segment Travel Time Statistics

In order to reconstruct the most likely path of a vehicle between its two updates, we first estimate the statistics of travel time for each road segment relying on historical probe data. Two successive updates of a probe vehicle provides a travel time observation. As mentioned before, most existing papers offer methods for estimating “link travel time”, which is the travel time between two adjacent intersections. Their implicit assumption is that travel time is uniformly distributed along a link, which is not true in most cases. Moreover, probe vehicle updates normally occur at random positions and times and not necessarily at the two ends of a link. To address these aforementioned issues, we propose to divide a whole path into short segments, and to allocate a travel time to each segment based on probe data.

Using the haversine formula, the reported longitude and latitude coordinates are converted to a linear distance measured from an arbitrary reference point at the upstream end. Let’s denote each of such segments by \( x_i, i = 1, 2, ..., N \). All segments have the same length with the resolution we choose. The travel time across each segment is denoted by \( TT_{x_i} \) which is a random variable. Every observation of this random variable is denoted by \( tt_{j}^{i} \), where \( j \) is the probe vehicle (here a bus) index. As shown in Figure 5.1, there is a travel time realization for each bus pass crossing a segment.

Assume there are \( M \) bus passes along the route indexed each by \( j \in \{1, 2, ..., M\} \). Each consecutive pair of updates by bus \( j \) provides a travel time observation denoted by \( tt_{[x_{a_1}, x_{a_n}]}^{j} \) from the beginning of segment \( x_{a_1} \) to the end of segment \( x_{a_n} \), where \( a_i \in \{1, 2, ..., N\} \). The interval \( [x_{a_1}, x_{a_n}] \) denotes the segments that lie between the two consecutive updates. If an update is not right on the segment boundary we assign it to the closest one. The error is acceptable as long as segments are short. The
sumination of travel time allocated to each segment must equal the total observed travel time, i.e.

\[ \sum_{i=a_1}^{a_n} tt^j_i = a_n \sum_{i=a_1}^{a_n} tt^j_i \] (3.1)

To generate estimates of travel time for each segment, \( tt^j_{x_1} \), we decompose the observed travel times between covered segments. We propose an Expectation Maximization (EM) algorithm to iteratively decompose each observed travel time between segments and subsequently calculate the statistics of travel time for each segment. We assume that each segment travel time has a Gaussian probability density function\(^1\). We also assume the conditional independence of segment travel times, that is, under the same traffic condition in the same time interval, a segment travel time is independent from all other segments\(^2\).

\(^1\)We are aware that the Gaussian assumption may cause problems such as it may result in negative travel time. Fortunately, the probability is extremely low. Under the resolution we choose, the mid-link mean segment travel time is 0.5 s, the average variance is around 0.1, therefore the chance of allocating a negative travel time (outside of 5 sigma) is 0.0001%. Segment travel time at intersections are higher and have less chance to cause such a problem.

\(^2\)The independence assumption is for simplification. A large amount of ground truth data is needed to verify how strong this assumption is. We only had access to 15 valid high frequency ground truth data traces for three different days, therefore we cannot conclude independence nor we can infer dependence. This is the weakness of this approach, although the results show good precision. More verifications are needed when more ground truth data is available.
Under these assumptions, after allocating travel times to each segment, we can calculate its probability. Then we can readjust the allocation by maximizing its likelihood function, and update the mean and variance for each segment travel time.

At the very first step, segment travel times are initialized evenly, that is, for a pair of GPS updates starting at \( x_{a_1} \) and ending at \( x_{a_n} \), we have

\[
\begin{align*}
t_{t_x}^j &= \frac{1}{a_n - a_1 + 1}t_{[x_{a_1},x_{a_n}]}^j
\end{align*}
\]  

(3.2)

for \( i \in \{a_1, a_2, \ldots, a_n\} \). The initialization is repeated for all observations.

Suppose \( K_{x_i} \leq M \) is the number of overlapping observations at segment \( x_i \), then for each segment \( x_i \), we have \( K_{x_i} \) travel time realizations \( t_{t_x}^k \), \( k \in \{1, 2, \ldots, K_{x_i}\} \). Next we iterate between the following E step and M Step till the segment travel times converge.

**E Step:** For each segment \( x_i \), we calculate the mean \( \mu_{x_i} \) and standard deviation \( \sigma_{x_i} \) of allocated travel times. Having these two parameters, with the assumption that segment travel times are normally distributed, the probability density for \( t_{t_x}^k \) can be calculated as:

\[
p(t_{t_x}^k | \mu_{x_i}, \sigma_{x_i}) = \frac{1}{\sqrt{2\pi\sigma_{x_i}}} \cdot e^{-\frac{(t_{t_x}^k - \mu_{x_i})^2}{2\sigma_{x_i}^2}}
\]

(3.3)

This is also the likelihood function of \( t_{t_x}^k \) given \( \mu_{x_i} \) and \( \sigma_{x_i} \). We often use the log-likelihood function for computational convenience, that is

\[
\log[p(t_{t_x}^k | \mu_{x_i}, \sigma_{x_i})] = -\frac{(t_{t_x}^k - \mu_{x_i})^2}{2\sigma_{x_i}^2} - \log(\sqrt{2\pi\sigma_{x_i}})
\]

(3.4)

**M Step:** In this step, for each update pair (observation), segment travel times \( t_{t_x}^k \) are reallocated such that the log likelihood function is maximized. Suppose an update pair
starts at segment \( x_{a_1} \) and ends at \( x_{a_n} \), its log likelihood function can be represented as:

\[
\log[p(tt^k_X|\mu_X, \Sigma_X)] = \sum_{i=a_1}^{a_n} \log[p(tt^k_{x_i}|\mu_{x_i}, \sigma_{x_i})] \quad (3.5)
\]

where \( X = [x_{a_1}, x_{a_2}, ..., x_{a_n}]^T \), \( \mu_X = [\mu_{x_{a_1}}, ..., \mu_{x_{a_n}}]^T \), and \( \Sigma_X = \begin{bmatrix}
\sigma^2_{x_{a_1}} & 0 & \ldots & 0 \\
0 & \sigma^2_{x_{a_2}} & \ldots & 0 \\
0 & 0 & \ldots & \sigma^2_{x_{a_n}}
\end{bmatrix} \).

Equation (5.12) holds under the assumption that segment travel times are independent from each other.

In the M step, the parameters \( \mu_{x_i} \) and \( \sigma_{x_i} \) are considered to be constants, therefore the terms containing only them can be ignored when maximizing the log-likelihood function with respect to allocated travel times. The problem then can be formulated as

\[
\arg \max_{tt^k_{x_i}} \left( \sum_{i=a_1}^{a_n} \log[p(tt^k_{x_i}|\mu_{x_i}, \sigma_{x_i})] \right) = \arg \min_{tt^k_{x_i}} \left( \sum_{i=a_1}^{a_n} \frac{(tt^k_{x_i} - \mu_{x_i})^2}{2\sigma^2_{x_i}} \right) = \arg \min_{tt^k_{x_i}} \left( \sum_{i=a_1}^{a_n} \frac{(tt^k_{x_i} - 2tt^k_{x_i}\mu_{x_i})^2}{2\sigma^2_{x_i}} \right)
\quad (3.6)
\]

subject to the equality constraint imposed by Equation (5.10). This is a Constrained Quadratic Programming (CQP) problem. We can rewrite the problem in the standard
form as,

$$\arg \min_y J = \frac{1}{2} y^T Q y + c^T y$$

s.t. \( \sum_{i=a_1}^{a_n} tt_{x_i}^k = tt_{[x_{a_1}, x_{a_n}]}^k \) (3.7)

where

$$y = \begin{bmatrix} tt_{x_{a_1}}^k \\ tt_{x_{a_2}}^k \\ \vdots \\ tt_{x_{a_n}}^k \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \end{bmatrix}, \quad \text{and} \quad c = \begin{bmatrix} \mu_{x_{a_1}} \\ \frac{\sigma^2_{x_{a_1}}}{\sigma^2_{x_{a_2}}} \\ \mu_{x_{a_2}} \\ \frac{\sigma^2_{x_{a_2}}}{\sigma^2_{x_{a_n}}} \\ \vdots \\ \mu_{x_{a_n}} \end{bmatrix}.$$

The solution to this CQP problem is the maximum likelihood segment travel time allocation for each observation. Note that the M Step is run separately for each pair of GPS updates (observation). After reallocation, the method goes back to the E step to update mean and variance for each segment based on all observations. Iterations are stopped upon convergence of the algorithm, when the difference between consecutive iterations is below a threshold. We have implemented this method in our previous work [103] for all buses (with or without stops). The segment travel time evolution and RMS (Root Mean Square) errors are shown in Figure 5.2. The positions of bus stops and intersections are also shown in vertical dashed and dash-dot lines. It is shown that our EM algorithm converges fast. With the initial guess, the RMS error of average segment travel time between iterations starts at 0.2m, and drops to 10^{-3}m at the 60th iteration. And a reasonable pattern is shown, in which segment travel times increase before bus stops and intersections.
Figure 3.3: Segment travel time and RMS errors evolution of our EM algorithm in [103].

### 3.4 Maximum Likelihood Trajectory Estimation

With statistical information about segment travel time, we can proceed to estimating the most likely trajectory between two consecutive probe updates. The problem is complicated by stops and idling at a red lights or at bus stops. Stopping durations and positions are different for each bus, due to their different arrival time, queue size at intersections, and variable time spent at bus stops. Therefore, if we treat all the data similarly when estimating travel time statistics, stop durations and positions will be averaged, and uncertainties will be large around intersections and bus stops. To better capture the differences, we propose to cluster the trajectories into those with stops and those without, and to deal with them separately.

#### 3.4.1 Estimation of Trajectories without Stops

In this chapter, we label the trajectory between two consecutive updates as “unstopped” if:
1. There is no intersection between the updates and the average velocity is above a threshold (i.e. 25 mph).

2. There is an intersection between the updates and either the time stamps of the former update and the latter update are within the same green light phase, or the average velocity is larger than a threshold (i.e. 25 mph).

Note that traffic signal timings are assumed to be available; in [25] our group demonstrated that fixed timings of traffic signals can be estimated from the Nextbus sparse updates. In [25], bus position and velocity updates along with their timestamps are used as the inputs for estimating signal timings. This was done by reconstructing the kinematics of bus movement across an intersection and the details can be found in [25].

We first select historical trajectories without stops and apply the method in Section 3.3 to estimate travel time statistics corresponding to unstopped passes. Next, given two successive updates of an “unstopped” trajectory, we reconstruct the trajectory according to the following two steps:

Step 1: Given two new updates, the time stamps of the former and latter updates are denoted by $T(x_{a_1})$ and $T(x_{a_n})$ respectively. We use the time of the former update and mean segment travel times to predict (or estimate) the time of the latter update, denoted by $\hat{T}(x_{a_n})$, as follows,

$$\hat{T}(x_{a_n}) = T(x_{a_1}) + \sum_{i=a_1}^{a_n} \mu_{x_i}$$

where $\mu_{x_i}$ is the mean segment travel time and can obtained as explained in Section 3.3. We define

$$\text{Delay}_{[x_{a_1}, x_{a_n}]} = T(x_{a_n}) - \hat{T}(x_{a_n})$$
as the delay in between the update pair, where a positive delay indicates a slower than average trip, and a negative delay indicates the reverse.  

Step 2: We use travel time statistics for each segment in the interval $[x_{a_1}, x_{a_n}]$ and allocate the $\text{Delay}_{[x_{a_1}, x_{a_n}]}$ to each segment using a maximum likelihood approach. Similar to Equation (5.10), we have

$$\text{Delay}_{[x_{a_1}, x_{a_n}]} = \sum_{i=a_1}^{a_n} \Delta tt_{x_i} \quad (3.10)$$

where $\Delta tt_{x_i}$ is the deviation of travel time in segment $x_i$ from its mean value $\mu_{x_i}$. The corresponding likelihood function for this “adjusted” time $\Delta tt_{x_i}$ can be obtained based on Equation (5.11) and according to:

$$p(\Delta tt_{x_i} | 0, \sigma_{x_i}) = \frac{1}{\sqrt{2\pi\sigma_{x_i}}} \cdot e^{-\frac{(\Delta tt_{x_i})^2}{2\sigma_{x_i}^2}} \quad (3.11)$$

where $\sigma_{x_i}^2$ is the segment variance as calculated in Section 3.3. To obtain the maximum likelihood trajectory we need to maximize a likelihood function similar to that of Equation (3.6) subject to the equality constraint in Equation (3.10). This leads to another CQP problem:

$$\arg\min_J = \frac{1}{2} y^T Q y + c^T y \quad (3.12)$$

s.t. $\sum_{i=a_1}^{a_n} \Delta tt_{x_i} = \text{Delay}_{[x_{a_1}, x_{a_n}]}$

---

3Note that the usage of the word “delay” in traffic context is often referring to the difference between an actual travel time and the free flow travel time. In our proposed method we are comparing an actual travel time to the mean historic travel time. For the lack of a better word, we refer to this time difference as delay. This is delay with respect to mean historic observations and can assume positive or negative values.
where
\[
\begin{bmatrix}
\Delta t_{xa_1} \\
\Delta t_{xa_2} \\
. \\
. \\
\Delta t_{xan}
\end{bmatrix},
\begin{bmatrix}
\frac{1}{\sigma_{xa_1}^2} & 0 & \ldots & 0 \\
0 & \frac{1}{\sigma_{xa_2}^2} & \ldots & 0 \\
0 & 0 & \ldots & \frac{1}{\sigma_{xan}^2}
\end{bmatrix},
\text{and } c = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix}.
\]

The solution to this problem are the travel time adjustments for each segment within the given pair of updates.\(^4\) After these adjustments, we have the most likely segment travel times between two update points, and hence the most likely trajectory.

### 3.4.2 Estimation of Trajectories with a Single Intersection Stop

A vehicle stops at an intersection if there is a queue in front and/or the traffic signal is in its red phase. The stop time and position are functions of queuing and discharging dynamics. Shock wave theory has been widely used to describe the queue dynamics [17, 92]. Figure 3.4 shows a model for the dynamics of a queue at a red light. Here the point \(C\) is the furthest point the queue end can reach. As a traffic signal turns red, the queue starts building up linearly over time as shown by the line \(AC\). Upon start of the next green, the vehicles start leaving the queue in a pace determined by the slope of line \(BC\). The slope of \(AC\) is determined by the upstream

\(^4\)Note that we solve two similar CQPs in this paper, but with different goals. The goal of the first CQP is to find mean segment travel times and their variances. The inputs are historical probe bus data, and the outputs are mean segment travel time (in general and not for individual pairs). In the second CQP the goal is to find the segment travel time for a specific pair of probe bus updates. The inputs are two consecutive data updates from the same bus. Solving the latter problem requires the result of the first problem, however, decomposing delays only occurs in the latter problem. Although both problems happened to be a CQP, they have different purposes. For example, in the first problem, the decomposition can only be positive since the variable is travel time, while in the second problem, the decomposition can be either positive or negative since the variable is delay.
traffic flow, and the slope of $BC$ is a function of the downstream traffic condition and average acceleration of departing vehicles. In this chapter, we assume that the queue discharge rate (slope of $BC$) is a constant determined by the mean headway and vehicle length [46]. Furthermore, we assume that the furthest queue end remains constant within a certain time period of day, which we estimate from historical data for different times of day. With these we can estimate the location of point $C$ in Figure 3.4, and calculate the upstream flow (slope of $AC$). The sensitivity of this approach to the assumed queue length and discharge rate is discussed later in this paper when discussing the results.

In this section, we focus only on the update pairs that span one intersection. As shown in Figure 3.4, a stopped update pair has potentially many candidate trajectories. Two example candidate trajectories are shown in a dotted line and a dash-dotted line respectively. Points $S_1$ and $S_2$ show when and where the bus joins and leaves the queue respectively. Once the stop position, denoted by $x_{S_1}$, is determined, the stop duration is determined (here $x_{S_1} = x_{S_2}$). In other words, the time coordinates of $S_1$ and $S_2$, denoted by $T(x_{S_1})$ and $T(x_{S_2})$, can be obtained. The MLT of the update pair is composed of three parts: the MLT from the former update to
point $S_1$, the idling trajectory from $S_1$ to $S_2$, and the MLT from point $S_2$ to the latter update. The vehicle is assumed to be stationary between $S_1$ and $S_2$, and not in the other two intervals. In order to obtain the MLT, the following CQP is formulated and solved:

$$\arg\min_y J = \frac{1}{2} y^T Q y + c^T y$$

s.t. $\sum_{i=a_1}^{S_1} \Delta t_{t_i} = \text{Delay}_{[x_{a_1}, x_{S_1}]}$ \hspace{1cm} (3.13)

$$\sum_{i=S_2}^{a_n} \Delta t_{t_i} = \text{Delay}_{[x_{S_2}, x_{a_n}]}$$

where

$$\text{Delay}_{[x_{a_1}, x_{S_1}]} = T(x_{S_1}) - \hat{T}(x_{S_1})$$ \hspace{1cm} (3.14)

$$\text{Delay}_{[x_{S_2}, x_{a_n}]} = T(x_{a_n}) - \hat{T}(x_{a_n})$$ \hspace{1cm} (3.15)

Given different stop positions in the queue, different MLTs can be obtained that have different maximum likelihood function values. We try all possible stop position choices, that is, from intersection position $A$ to the furthest queue end $C$, and then choose the one with the maximum likelihood function value as the final
3.4.3 Estimation of Trajectories with Multiple Stops

If there are multiple intersections between two successive updates, we need to determine multiple stop positions and durations. Figure 3.5 shows an example in which a bus experiences stops at two adjacent intersections. Here we utilize a recursive approach to solve the problem. To estimate the trajectory between the two updates, we first try all possible stop positions for the first intersection. Under the assumption of a fixed queue pattern, each candidate stop position determines $S_1$ and $S_2$. Then the problem is to find the MLT connecting point “former” to point $S_1$ and point $S_2$ to the “latter” update. In order to find the MLT from point $S_2$ to the “latter” update, we invoke the estimation method itself again, that is, we try all possible stop positions at the second intersection and find the one which results in the maximum likely trajectory. In other words, for each candidate stop position at the first intersection, we need to solve another set of MLT estimation problems for stopped vehicles at the second intersection. The trajectory with the maximum likelihood of all possible trajectories is then chosen as the final result. Note that the number of possible stop positions is determined by the furthest queue end at the intersection and the resolution we choose. Suppose we have $n$ candidates for the first intersection and $m$ for the second; the computation time complexity is $O(n \times m)$. With the increment of number of intersections, the computation time complexity grows exponentially. However, estimating a trajectory crossing many intersections is not meaningful in general since the uncertainty would dominate the result. We found that with a small number of intersections, our approach is computationally manageable.
Figure 3.5: A Schematic of queuing and discharging dynamics at multiple intersections.

3.5 Results and Validation in Comparison with Ground Truth Data

To validate our approach, we chose a portion of the southbound path on Van Ness Avenue in the city of San Francisco, spanning two adjacent intersections: Lom- bard street and Greenwich street. Historical data was accumulated, using the sparse Nextbus feed, of bus routes 47 and 49 from January to September of 2013. The hourly segment travel time statistics for different time periods of day were then calculated. On Van Ness avenue the signals are pre-timed, and the SPaT information is gathered manually\(^5\). However signals have different offset times for different times of day. Our approach has taken these offsets into account when obtaining segment travel time statistics. And because of clock drift of traffic light we also used a crowdsourcing approach to estimate green initiations in near real-time based on latest bus passes [25]. There are other sources of uncertainty beyond traffic signals, such as slow traffic, that introduce errors in our estimations. If the signals were actuated, current approach is

\(^5\)In many cities the potential exists to obtain signal timings in real-time from the traffic control center. Alternatively in our group we have shown that timing of fixed time signals can be inferred from sparse vehicular probe data [25]
Figure 3.6: Travel time statistics for unstopped trajectories. With respect to the reference point, the Lombard intersection lies at 377 m, and Greenwich intersection lies at 480 m.

expected to have had larger errors.

In order to evaluate the accuracy of the proposed MLT estimator, a set of high frequency ground truth data was gathered on November 3rd, 4th, and 5th, 2014. Because high frequency data was not directly available from San Francisco transit agency or other sources, we collected this data using a GPS receiver and while physically riding on multiple buses on Van Ness avenue. We used a high-sensitivity, 12-parallel-channel, USB-connected Garmin GPS 18x receiver that recorded its GPS coordinates at a frequency of 1 Hz. To ensure the best possible signal reception, we tried to place the receiver as close as possible to a window, depending on the bus occupancy level. Route 47 typically used standard buses with a length of 40 feet while route 49 buses were articulated with a length of 60 feet. To ensure consistency in GPS recordings, we did our best to place the recorder at almost the same distance from the front of a bus in every ride. After three days of recording, 15 valid high frequency GPS tracks had been recorded.
Figure 3.7: Histogram of position of buses stopped at Lombard intersection as captured from January to September of 2013.

We first estimate the statistics of segment travel times using only the unstopped trajectories of the training data. The unstopped trajectories are selected based on the two steps outlined in Section 3.4.1 and using a lower velocity threshold of 6.5 m/s. Each segment length is chosen to be 5 m. Figure 3.6 shows the mean travel time allocation and its standard deviation after convergence of the EM algorithm for the update pairs that qualify the “no stop” conditions.

The maximum queue size at each intersection is also learned from training data. For example, Figure 3.7 shows a position histogram of buses with zero velocity at Lombard intersection over a time span of a few months, and for three different times of day. From this histogram an estimate for the furthest queue end during the morning rush hour is 80 meters and is what we use in this chapter for estimating trajectories in the specific hour of 8:00-9:00 AM. To calculate the discharging rate of the queue (slope of line BC in Figure 3.4), we assumed a headway of 1.4 seconds and average vehicle length of 5.5 meters [46], resulting in the queue discharge rate of 3.9 m/s. Here we assume that all intersections have the same discharging rate.

To estimate the spatiotemporal location of a queue the timing of the traffic signal is also needed as seen in Figure 3.4. The signals at Lombard and Greenwich intersections are fixed time and their baseline timing was known to us from the city...
Figure 3.8: Estimated and ground truth trajectories spanning a single intersection with no stop.

timing cards. The starting time of a green phase was estimated once using ground truth measurements and then cyclically mapped forward in time as explained in [25]. To correct for potential clock drift of the traffic signal, the green initiation time could be periodically corroborated/adjusted by a crowd-sourcing algorithm that estimates initiation of a green phase based on probe data in real-time. Details and performance of such an algorithm are described in [25].

Given travel time statistics for each segment, traffic signal timings, and historical queue patterns, we calculated the most likely trajectory between two distant updates. The results were compared against high resolution ground truth recordings. Figure 3.8 shows this comparison for 4 unstopped trajectories spanning a single intersection. The estimations are close to ground truth readings for the most part.
Figure 3.9: Estimated and ground truth trajectories spanning a single intersection with a stop.

Note that there is an unidentified short slow downs happening at 440 m in 3.8c and 3.8d, and at 300 m in 3.8d. These unusual mid-link delays could not be captured in historical data, therefore the estimator did not locate the delay and it just distributed it between segments.
Figure 3.10: Estimated and ground truth trajectories spanning multiple intersections.

In the second scenario we estimate stopped trajectories that span a single intersection. Figure 3.9 shows six trajectories spanning Lombard intersection. It can be seen that our approach finds the stop position in queue and stop duration with little error in most cases. Note that our estimator also takes acceleration and deceleration constraints into account. This helps accuracy of the estimator right before and right after a stop position.

The third scenario considers trajectories spanning multiple intersections. Figure 3.10 shows buses crossing Lombard and Greenwich intersections with one or two stops. Again the estimator is able to capture the actual trajectory well in most instances. Even in 3.10d where the bus experiences stops at each intersection, the estimator is able to detect relatively accurately the position in queue at each intersec-
tion. To emphasize, the results illustrate that our approach has convincing estimation performance even when intersections lie in between the updates.

To further demonstrate the effectiveness of our approach, a benchmark (naive) approach is proposed here. The naive approach assumes that signal timings are known and vehicles travel at average free flow speed (e.g. 7.5m/s). It uses shock-wave theory to determine the stop position and duration in queues. The queue length is assumed to be the historical average. Figure 3.11 compares the results in different scenarios. It can be seen that although our approach has a similar performance when estimating moving vehicle trajectories, it is more accurate in estimating stop locations and stop duration at intersections.

To quantitatively evaluate the estimation results, we calculate the Mean Absolute Error (MAE) between the estimated and the ground truth trajectories. The MAE is defined as:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |e_i| = \frac{1}{n} \sum_{i=1}^{n} |\hat{x}_i - x_i|$$

(3.16)

where $\hat{x}_i$ is the estimated position and $x_i$ is the actual position at time step $i$.

We list MAEs for each path in Table 3.1. The stop position in queue is also important for stopped buses and is included in Table 3.1.

The estimation accuracy is affected by various factors. Obviously the likelihood of errors increases with a longer distance between two updates. Moreover, the uncertainty increases with larger number of stops. The time period of day also influences the results, since during off peak hours the queue size may have more cycle variability, and the assumption of a fixed queue pattern does not hold. We assumed the traffic signal timing clock time is accurately predictable, but there can be slow drifts in a signal clock that would skew our results. The ground truth GPS data
Figure 3.11: Comparison of the maximum likelihood trajectory estimation and a naive approach.

itself contains errors. For example, 3.9d and 3.9e show that the ground truth position decreases over some time intervals, indicating that the bus traveled backwards, which is highly improbable and could be due to missing or erroneous GPS updates.

One may have the concern that we only use average headways and queue patterns in our approach, however in reality they are not constant, are affected by traffic congestion levels and vary with time. We are aware that such assumptions introduce errors. However our approach relies only on historical probe data, which does not contain more information. We have also performed a sensitivity analysis on the queue end and headways. Given two updates, we use different queue end and
headways to estimate the trajectory and calculate the errors in stop position and the mean absolute errors. The results are shown in Figure 3.12.

The computation time is another factor to be evaluated. The computer we used had I5-2310 4 cores 2.9MHz CPU, 8Gb memory, 1Tb Hard Disk and Windows 7 Home Premium 64 bit operation system. For the mean segment travel time CQP, in which the link length is 800 meters, containing 9 intersections or bus stops, using 6 months probe bus data, the overall computational time is 40 minutes which can be done off line. For each pair of updates, the delay decomposition CQP computational time is from 5 to 40 seconds, depending on the length between the updates and whether there are signalized intersections in between or not. These numbers show that our approach has the potential to be applied in real time. For example in a connected vehicle scheme, a cloud server can solve the problem within seconds. Vehicles connected to the server can receive the trajectory estimation results before

![Figure 3.12](image-url)
<table>
<thead>
<tr>
<th>Pass ID</th>
<th>MAE(m)</th>
<th># of Stops</th>
<th>Error-in-Queue(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.78</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>2</td>
<td>3.50</td>
<td>1</td>
<td>1.86</td>
</tr>
<tr>
<td>3</td>
<td>24.06</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>4</td>
<td>6.62</td>
<td>1</td>
<td>6.14</td>
</tr>
<tr>
<td>5</td>
<td>3.81</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>6</td>
<td>6.53</td>
<td>1</td>
<td>2.34</td>
</tr>
<tr>
<td>7</td>
<td>2.49</td>
<td>0</td>
<td>2.05</td>
</tr>
<tr>
<td>8</td>
<td>6.54</td>
<td>1</td>
<td>4.33</td>
</tr>
<tr>
<td>9</td>
<td>7.57</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>10</td>
<td>7.44</td>
<td>1</td>
<td>0.61</td>
</tr>
<tr>
<td>11</td>
<td>3.61</td>
<td>1</td>
<td>3.11</td>
</tr>
<tr>
<td>12</td>
<td>5.93</td>
<td>1</td>
<td>3.40</td>
</tr>
<tr>
<td>13</td>
<td>6.60</td>
<td>1</td>
<td>8.00</td>
</tr>
<tr>
<td>14</td>
<td>3.40</td>
<td>1</td>
<td>0.64</td>
</tr>
<tr>
<td>15</td>
<td>11.25</td>
<td>2</td>
<td>5.25, 2.48</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>6.94</strong></td>
<td><strong>3.36</strong></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: Trajectory Estimation Errors.

3.6 Conclusions

This chapter presented a new method for reconstructing the trajectory of vehicles between sparse updates using a maximum likelihood approach. We introduced an iterative method for estimating travel time statistics for short segments of an arterial road using only sparse updates from probe vehicles. Using the same data feed we could estimate historical queue patterns at intersections. Relying on travel time and queue statistics and prior knowledge of signal timings, we described an algorithm that effectively estimates the most likely trajectory of vehicles in between sparse position updates. Moreover the estimated trajectories spanning an intersection could capture the stop-and-go pattern induced by traffic signals and the most likely position of the vehicle in the queue.

The results are consistent with ground truth readings and are quite promising.
given uncertainties arising from traffic signal drift, variability in queue size, inconsistencies in GPS recordings, and the limitation posed by the assumptions we have made when constructing the estimator.
Chapter 4

Optimal Speed Advisory For Connected Vehicles in Arterial Roads and the Impact on Mixed Traffic

4.1 Literature Review

A large body of research has been done on developing driving strategies that improves fuel economy. For example in [62] the authors proposed a fuel optimized operating strategy, they concluded that the optimal operating strategy is periodic because of the S-shaped engine fueling rate. In [22], the authors propose a headway control algorithm to reduce fuel. In [32] and [33], the authors propose look-ahead control algorithms which take upcoming road topography into account to reduce fuel consumption. These approaches assume that vehicles are not communicating among each other. With CV technologies, new approaches are emerging rapidly. In [7],
the authors propose energy saving strategies for plug-in hybrid electric vehicles using upcoming traffic signal timing and headway information.

A Speed Advisory System (SAS) that aids in reducing idling near traffic signals is one of the applications of CV technology, which has been proposed by our research group [2], [1], [69] and various other researchers across the world [58], [71], [89], [106]. Vehicles equipped with a Speed Advisory System (SAS) can utilize upcoming traffic signal information predictively and manage their speed in advance to reduce idling at red lights. SAS relies on vehicle connectivity to obtain traffic Signal Phase and Timing (SPaT). The technology for transmitting traffic signal information to subscribing vehicles has been demonstrated in several research projects [58], [107] including in projects by our group [70]. The SPaT information may be directly transmitted to vehicles within range using Dedicated Short Range Communications (DSRC) technology [58] or may become available by the traffic control center through cellular and Wi-Fi networks as shown in [70]. Alternative means of inferring SPaT information via on-board cameras [58] and via crowd-sourcing [25] have also been proposed.

Motion planning or trajectory planning problems can be formulated as optimal control problems [53]. Past research has formulated the speed advisory problem as optimal control problems and obtained the optimal speed trajectory. In [1], [48] and [49] Model Predictive Control (MPC) approaches have been used to obtain near optimal trajectories while considering traffic signals. In [30] the authors propose to obtain speed trajectories considering queue pattern and signal timings. In [69], [51], [52], and [83], the authors propose Dynamic Programming (DP) approaches to solve the optimal control problem. Unfortunately, these methods are costly in terms of CPU and memory use and often cannot be executed in real time. In [84], the authors use a linearized model of a vehicle’s longitudinal dynamics and solved the fuel
minimization problem analytically with given boundary conditions. This analytical method is computationally less expensive and by which the approach proposed in our paper is inspired. The solution we propose maintains the nonlinearities in vehicle dynamics, relaxes the boundary conditions to the minimum required information, and solves the optimal control problem relying on Pontryagin’s Minimum Principle (PMP) and kinematic constraints. We show that the optimal solution requires switching between maximum engine torque (boost) and engine shut-down (glide) and occasionally includes a period of constant speed (sustain). Similar conclusions can be found in [63],[65]. We then argue that the resulting speed profile, while fuel optimal, is uncomfortable to drivers and may also be disruptive to surrounding traffic. We then resort to modified suboptimal speed profiles and still show improvement in fuel economy as a result of avoiding red lights.

Equipped with analytical solutions, we then evaluate the impact on fuel economy in mixed traffic conditions. SAS can significantly reduce energy consumption of individual vehicles and improves their ride comfort, yet it decreases the average speed of equipped vehicles and increases their travel time. It is not difficult to analyze the effect on each equipped vehicle itself [1],[69],[89],[9],[72]. However, the SAS technology is unlikely to be implemented in every vehicle in the near future. Therefore it is essential to evaluate the influence of equipped vehicles on other vehicles in mixed traffic flow. There are many papers aiming at evaluating the impact of adaptive cruise control in mixed traffic [54],[45], or the impact of vehicle-infrastructure cooperation [24]. However, to the authors’ best knowledge, there are few papers discussing the impact of SAS on mixed traffic for multiple intersections. For instance in [52] and in [108] the authors evaluate the influence of eco-driving or eco-speed control only on vehicles equipped with such a system, or only on the surrounding traffic. Another example in [77] only discuss impacts of eco-driving on ego-vehicle’s pollutant emission
and fuel consumption. In [9] the authors only discuss impacts of speed advisory to driving behavior. In this paper, we evaluate the influence of SAS-equipped vehicles on each other as well as on conventional vehicles. Moreover, in this chapter we evaluate the system when traveling across multiple intersections.

It is currently prohibitively difficult to do in the field experiments of a large number of connected vehicles in mixed traffic. Therefore it is necessary to choose a simulation tool to conduct simulations under different traffic situations. In this chapter we use the microscopic traffic simulation tool *Paramics*. Paramics is able to simulate a large number of vehicles in a complex traffic network. Moreover, it is easy to set percentages of different types of vehicles and adjust traffic demands. A direct result is measurement of instantaneous speed and acceleration values that influence driving comfort as well as fuel consumption. Vehicles equipped with SAS aim to avoid sharp braking and/or stopping at traffic signals, which improves their fuel efficiency. We will use a fuel consumption model to calculate the fuel economy of each vehicle based on its velocity and acceleration profile. From velocity trajectories we also evaluate travel time of each vehicle. In Paramics it is possible to install virtual sensors to also measure traffic flow. We use these virtual measurements to evaluate the side effects of the proposed SAS technology.

The rest of this chapter is organized as follows: Section 4.2 presents the optimal control framework for obtaining analytical solutions for the optimal speed trajectory of individual vehicles. Section 4.3 introduces the simulation environment, parameters, and various test setups. The results are summarized in Section 4.4 followed by conclusions in Section 4.5.
4.2 Optimal Control Formulation of SAS

In SAS the goal is to calculate a reference speed profile based on ego vehicle’s position, speed, and upcoming Signal Phase and Timing (SPaT) such that if the vehicle follows this speed trajectory, it consumes the least amount of fuel. Often the vehicle is guided to pass the intersection when the light is green. This is due to the fact that stop-and-go motion requires more energy than cruising and should be reduced as much as possible. In this section we formulate this problem as an optimal control problem and present an analytical solution for it.

4.2.1 The Vehicle Model

The vehicle longitudinal dynamics is needed and can be written as [100]:

\[ m \dot{v} = \frac{T_e}{r_g} - \frac{1}{2} \rho_a A C_D v^2 - mg(\mu \cos \theta + \sin \theta) - F_b \]  

(4.1)

where \( m \) is mass of the vehicle and includes powertrain inertial effects, \( v \) is forward velocity, \( T_e \) is the engine torque at the flywheel, \( F_b \) is the braking force generated at the tire contact point with the road. The parameter \( r_g \) is the wheel radius divided by total gear ratio. Air density is \( \rho_a \), \( A \) is vehicle front area, \( C_D \) is aerodynamic drag coefficient, and \( \mu \) is the rolling friction coefficient. The road slope angle is \( \theta \) and in this chapter it is assumed to be a constant over time to simplify the analytical derivations. We also assume the gear ratio and therefore \( r_g \) remain constant; this assumption enables us to generate analytical solutions and is valid during a large part of city cruising when the vehicle is in a fixed gear.\(^1\)

\(^1\)One can consider the gear choice as an extra degree of freedom for energy optimization which introduces a discrete optimization variable and almost certainly requires a numerical optimization procedure. Alternatively and more realistically one can rely on the existing gear shift logic of a vehicle and augment it either in the longitudinal dynamics model in Eq. (4.1) or incorporate it in
The longitudinal dynamics in (4.2) can be rewritten in the following form after dividing both sides by $m$:

\[ \dot{v} = u_e - C_1 v^2 - C_2 - u_b \] (4.2)

where $u_e = \frac{T_e}{mg}$ is the vehicle acceleration contributed by engine torque, $u_b = \frac{F_b}{m}$ is the vehicle deceleration due to braking force, and $C_1 = \frac{1}{2} \rho a A C_D m$ and $C_2 = g(\mu \cos \theta + \sin \theta)$ are constants, where we assumed that road grade is time-invariant.

The fuel minimization problem can be cast as an optimal control problem, in which the vehicle’s position $x$ and velocity $v$ are the dynamic states and the control inputs are $u_e$ and $u_b$. Note that the braking force should be zero while the engine torque is positive, and vice versa. And we intuitively know that the minimum fuel solution will avoid braking when possible. Therefore we can separate the engine engaged case from the braking applied case, and discuss them respectively.

Based on Equation (4.2) and when braking is not applied, we have the following state-space equations:

\[
\begin{cases}
\dot{x} = v \\
\dot{v} = u_e - C_1 v^2 - C_2
\end{cases}
\] (4.3)

The upper bound on control input $u_e^{max}$ corresponds to the maximum engine torque $T_e^{max}$ and the lower bound $u_e^{min} = 0$ corresponds to zero engine torque (engine shut-off). Note that maximum engine torque depends on the operating point of the engine. However since $u_e^{max}$ is the maximum traction force at wheel, we assume that a fixed maximum value is always achievable by selection of gear and engine torque. The upper bound on the velocity is the maximum speed limit of the road and the lower the fuel economy model. In either case, the problem will not lend itself to an analytical solution such as the one presented in this paper and a numerical solution may be unavoidable.
bound is such that the vehicle does not significantly block the traffic.

Besides the vehicle dynamics model, a fuel consumption estimation model is also needed. In this chapter, the fuel consumption estimation model is adopted from [50], where the authors sampled sufficient data from a passenger size vehicle and fit into third order polynomial curves that approximate the relation between fuel consumption rate and velocity and acceleration. In this model the fuel consumption rate $\dot{m}_f$, during positive acceleration and constant speed cruising is estimated by:

$$
\dot{m}_f = \alpha_0 + \alpha_1 v + \alpha_2 v^2 + \alpha_3 v^3 + (\beta_0 + \beta_1 v + \beta_2 v^2)a
$$

where $a$ is the vehicle acceleration and $\alpha_i$ and $\beta_i$ are model parameters. The parameter values is presented later in Table 4.1. Note that this fuel consumption model assumes that the road gradient is zero. When the acceleration is negative, we assume that the engine is idling and consuming a minimal constant fuel rate $\alpha_0$.\(^2\) To summarize, we employ the following model for fuel consumption rate:

$$
\dot{m}_f = \begin{cases} 
\alpha_0 + \alpha_1 v + \alpha_2 v^2 + \alpha_3 v^3 + (\beta_0 + \beta_1 v + \beta_2 v^2)a, & a \geq 0 \\
\alpha_0, & a < 0 
\end{cases}
$$

(4.4)

### 4.2.2 Fuel Consumption Minimization

The vehicles equipped with SAS can uni-directionally receive the relevant signal information via DSRC. Alternatively equipped vehicles may subscribe to each upcoming light via a cellular network and receive updated SPaT information from the signal; a technology that our group has successfully demonstrated recently [70].

\(^2\)Later in this chapter we show that the optimal strategy is to either run the engine at maximum torque, zero torque, such that the vehicle is at a constant speed. Therefore under these conditions, negative vehicle acceleration can only correspond to engine idle/shut-off which justifies the assumption of minimal constant fuel rate.
The distance between the light and current vehicle can be estimated from the vehicle’s GPS coordinates and traffic light information. Through V2I communication, the information such as current phase, time left in the current phase, phase duration and cycle length of the upcoming traffic signal also become available. Therefore, several time windows that the vehicle can pass at green can be calculated as illustrated schematically in Figure 4.1. In this paper it is assumed that the vehicle calculates the first feasible time to pass during green, given the road speed limits. In the future it may be possible for intelligent traffic signals to reserve and allocate a time to each subscribing vehicle to pass during green [3].

Given a start time $t_0$ and a target time $t_f$ to arrive at the signal, our objective is to find the fuel minimal velocity profile and corresponding control inputs for the vehicle. In other words the goal is to find the control input that minimizes the following cost function:

$$J = \int_{t_0}^{t_f} \dot{m}_f dt$$ (4.5)

subject to the state dynamics in Eq. (4.3) and the control input constraint that $0 \leq u_e \leq u_e^{max}$. This is a calculus of variation problem and one can obtain necessary conditions for optimality in the form of ordinary differential equations. In control
theory these necessary conditions for optimality are described by Pontryagin’s Minimum Principle (PMP) [55]. According to PMP, first the Hamiltonian is constructed as the sum of the integrand in (4.5) and the right side of the state equations in (4.3) multiplied by Lagrange multipliers λ_1(t), also referred to as costates:

\[
H = \dot{m}_f + \lambda_1(t)v + \lambda_2(t)(u_e(t) - C_1v^2 - C_2)
\]  

(4.6)

PMP states that the optimal input trajectory \( u_e^*(t) \) and the corresponding optimal state and co-state trajectories must minimize the Hamiltonian. Combining Equations (4.3), (4.4), and (4.6), we observe that in our problem the Hamiltonian is an affine function of the control input \( u_e \); therefore the partial derivative,

\[
H_u = \frac{\partial H}{\partial u_e} = \beta_0 + \beta_1v + \beta_2v^2 + \lambda_2
\]  

(4.7)

does not contain the control input term \( u_e \). This indicates that the optimal control switches between control boundaries, 0 and \( u_{e}^{max} \), depending on the sign of \( H_u \). If \( H_u = 0 \) for an interval in time, the system is said to be on a singular arc and the control will assume a value between the upper and lower constraints; in this paper we refer to it as \( u_e^{sing} \). In other words,

\[
u_e^* = \begin{cases} 
    u_{e}^{max} & \text{if } (\beta_0 + \beta_1v + \beta_2v^2 + \lambda_2) < 0 \\
    0 & \text{if } (\beta_0 + \beta_1v + \beta_2v^2 + \lambda_2) > 0 \\
    u_{e}^{sing} & \text{if } (\beta_0 + \beta_1v + \beta_2v^2 + \lambda_2) = 0
\end{cases}
\]

(4.8)

which is referred to as bang-singular-bang control\(^3\). The PMP necessary conditions

\(^3\)The bang-bang (pulse and glide) fuel optimal result is corroborated by the findings in recent publications, for instance in [64]. In [64] the authors use a nonlinear engine fuel consumption map and explain that periodic switching between high (but not the maximum) engine torque and zero engine torque is more fuel economical than operating the engine at medium torque levels. Our fuel consumption model, adopted from a recently published paper [50] is obtained by regression
also describe the dynamics of the co-states as $\dot{\lambda}_1 = -H_x$ and $\dot{\lambda}_2 = -H_v$, therefore:

$$
\begin{align*}
\dot{\lambda}_1 &= -\frac{\partial H}{\partial x} = 0 \\
\dot{\lambda}_2 &= -\frac{\partial H}{\partial v} = -[\alpha_1 + 2\alpha_2 v + 3\alpha_3 v^2 + (\beta_1 + 2\beta_2 v)(u_e - C_1 v^2 - C_2) \\
&+ (\beta_0 + \beta_1 v + \beta_2 v^2)(-2C_1 v) + \lambda_1 - 2C_1 \lambda_2 v]
\end{align*}
$$

(4.9)

In general the dynamics of co-states must be tracked because the optimal control input will depend on them. State dynamics in Eq. (4.3), the co-state dynamics in Eq. (4.9) together with Eq. (4.8) form the necessary conditions for optimality. To solve the 4 differential equations in (4.3) and (4.8), which are in general coupled, 4 boundary conditions are needed. Here $x(t_0)$, $v(t_0)$, and $x(t_f)$ are specified. We can either constrain the velocity at $t_f$ which provides an additional boundary condition or leave it open; in the latter case the value of $\lambda_2(t_f)$ will be fixed and known according to the theory of optimal control [55]. This is a two-point boundary value problem, because the boundary conditions are split at both ends, and in general is very hard to solve analytically. Fortunately the special structure of the problem in this paper (bang-singular-bang control and mild coupling in dynamics) allows us to obtain an analytical solution as discussed next.

and relates fuel consumption rate to vehicles speed and acceleration. As a result this model has embedded in it not only the engine fuel consumption map but also the vehicles gear shift logic. The relationship between fuel rate and acceleration input of the engine in this model is linear. Therefore, with this model, the analytically calculated optimal solution is proven to switch between maximum and zero engine acceleration input at the wheel. Note that maximum engine input at the wheel does not necessarily map to maximum engine torque due to the embedded gear shift logic.

Note also that our results are also limited by its affine approximation. It is however assuring to see that a similar bang-bang optimal structure was found in [64] based on a nonlinear engine map.
4.2.3 Optimal Strategy on the Singular Interval

As shown in Equation (4.8), $H_u = \partial H / \partial u$ vanishes when,

$$H_u = \beta_0 + \beta_1 v + \beta_2 v^2 + \lambda_2 = 0 \quad (4.10)$$

This relationship could only hold for a point in time, upon which the control switches between its maximum and minimum. But if this relationship holds for an interval of time, we say we have a singular interval and the optimal solution may move along a singular arc. Because the control $u_e$ does not appear in (4.10), we cannot directly determine its optimal value on the singular arc.

However, since (4.10) must hold for an interval of time on a singular arc, its time derivatives must vanish [10],

$$\frac{d^n}{dt^n} \left( \frac{\partial H}{\partial u_e} \right) = 0 \quad (4.11)$$

and it is guaranteed that upon taking time derivatives repeatedly, the control input will finally emerge and therefore can be solved for. The proof can be found in standard optimal control texts such as [10].

Taking the first time derivative of Equation (4.10) and substituting from state and co-state dynamics we get,

$$\frac{d}{dt} \frac{\partial H}{\partial u_e} = \beta_1 \dot{v} + 2\beta_2 v \ddot{v} + \dot{\lambda}_2 = 0 \Rightarrow$$

$$\beta_1(u_e - C_1 v^2 - C_2) + 2\beta_2 v(u_e - C_1 v^2 - C_2)$$

$$- \{ \alpha_1 + 2\alpha_2 v + 3\alpha_3 v^2$$

$$\quad + (\beta_1 + 2\beta_2 v)(u_e - C_1 v^2 - C_2)$$

$$\quad + (\beta_0 + \beta_1 v + \beta_2 v^2)(-2C_1 v) + \lambda_1 - 2C_1 \lambda_2 v \} = 0$$
Substituting for $\lambda_2$ from Eqn. (4.10) reduces the above equation to,

$$\alpha_1 + 2\alpha_2 v + 3\alpha_3 v^2 + \lambda_1 = 0 \quad (4.12)$$

which provides a relationship between the constant co-state $\lambda_1$ and $v(t)$; but it is not still clear if this holds only at a singular point or on a singular arc. Note that $u_e$ has not appeared in (4.12), thus we calculate the second time derivative. Since $\lambda_1$ is known to be a constant, we obtain

$$\frac{d^2}{dt^2} \frac{\partial H}{\partial u_e} = 2\alpha_2 \dot{v} + 6\alpha_3 v \dot{v} = 0 \Rightarrow (2\alpha_2 + 6\alpha_3 v)(u_e - C_1 v^2 - C_2) = 0 \quad (4.13)$$

The control input has finally appeared and when it is equal to $C_1 v^2 + C_2$ the second time derivative vanishes. If this condition holds we can also conclude that $\dot{v} = 0$ and the velocity is a constant $v_c$. Therefore the optimal control candidate on the singular arc is a constant and equal to:

$$v_e^{\text{sing}} = C_1 v_c^2 + C_2. \quad (4.14)$$

But the minimizing solution should satisfy another necessary condition referred to as Kelley’s condition [10] which is,

$$(-1) \frac{\partial}{\partial u_e} \left[ \frac{d^2}{dt^2} \left( \frac{\partial H}{\partial u_e} \right) \right] \geq 0 \quad (4.15)$$

thus requiring,

$$2\alpha_2 + 6\alpha_3 v \leq 0 \quad \Rightarrow \quad v \leq -\frac{\alpha_2}{3\alpha_3} \quad (4.16)$$
which provides a bound on speed, above which the singular arc will not be a part of the optimal solution. In this problem and with the values listed in Table 4.1 the optimal solution could follow a singular arc if $v \leq 4.14 \text{ m/s or } 15 \text{ km/h}$. This is quite a low speed and therefore singular arcs of constant speed may only occasionally happen and in general the optimal solution will be of bang-bang form.

### 4.2.4 Analytical Solution for Minimum Fuel Control

As shown in Eq. (4.8), the optimal engine input alters between its two extreme values if the inequality conditions hold. When the equality condition is satisfied for an extended interval, the system is said to be moving on a singular arc, and the optimal control input could vary between $u_{e}^{\text{max}}$ and zero. Further analysis is needed to verify the existence of a singular arc and to obtain the corresponding optimal control input $u_{e}^{\text{sing}}(t)$. We present the analysis in Section 4.2.3.

The conclusion in Section 4.2.3 is that a singular arc could exist (albeit occasionally and at very low speeds). On a singular arc the optimal velocity and optimal control input are constants, we refer to the constant velocity by $v_{c}$, and consequently the optimal control input will be $u_{e}^{\text{sing}} = C_{1}v_{c}^2 + C_{2}$. The optimal control is said to have a Bang-Singular-Bang form as shown in Figure 4.2.

Figure 4.2 shows that the optimal velocity trajectory consists in general of...
three segments:

1. \( t_0 \leq t \leq t_1 \)

   In the first segment where the inequality conditions in (4.8) hold, the optimal torque is either the maximum or the minimum. Since the initial states \((x(t_0), v(t_0))\) are specified, the initial values of costates \((\lambda_1(t_0), \lambda_2(t_0))\) are open, according to the theory of calculus of variations. A relationship between \( t_1 \) and \( v_c \) can be obtained by solving,

   \[
   t_1 - t_0 = \int_{v(t_0)}^{v_c} \frac{dv}{u_e - C_1v^2 - C_2}
   \]

   Also \( d_1 \) is the area under the velocity curve and therefore can be analytically related to \( v_c \).

2. \( t_1 \leq t \leq t_2 \)

   This period exists only if a singular arc is determined to be a part of the optimal solution. We have shown in Section 4.2.3 that the velocity will be constant on a singular arc, therefore \( v(t_1) = v(t_2) = v_c \) and \( d_2 = (t_2 - t_1)v_c \). The costates will also be constants as shown in Equations (4.10) and (4.12) in Section 4.2.3.

3. \( t_2 \leq t \leq t_f \)

   In the third segment, the inequality conditions hold again and the optimal input is either at its maximum or the minimum. We have the following relationship,

   \[
   t_f - t_2 = \int_{v_c}^{v(t_f)} \frac{dv}{u_e - C_1v^2 - C_2'}
   \]

   The final position \( x(t_f) \) is fixed, which is the position of the intersection. Therefore \( \lambda_1(t_f) \) is free. The final velocity \( v(t_f) \), on the other hand, can be either
open or fixed. If \( v(t_f) \) is open, \( \lambda_2(t_f) \) should be fixed to zero according to calculus of variation theory. Given \( \lambda_2(t_f) = 0 \) the ordinary differential equation in Equation (4.9) can be solved backward in time to find \( \lambda_2(t_2) \) which is related to \( v_c \) via Equation (4.10). This could be a tedious process and will need a numerical approach, since in Eq. (4.9) \( \lambda_2 \) and \( v \) are coupled. Alternatively if \( v(t_f) \) is specified or constrained, one only needs to integrate the velocity equation and Eq. (4.9) does not need to be solved. This simplifies the process and allows an analytical solution to the problem. The distance \( d_3 \) is the area under the velocity curve and therefore can be analytically related to \( v_c \).

Finally the total distance traveled is fixed and known,

\[
d_1 + d_2 + d_3 = d = x(t_f) - x(t_0)
\]  

which provides an additional relationship between the unknown variables \( v_c, t_1, \) and \( t_2 \) and allows us to calculate the optimal velocity trajectory analytically.

As shown in Section 4.2.3, only at very low speeds the constant velocity portion in segment 2 will be a part of the optimal trajectory. In the majority of scenarios the fuel optimal trajectory will be bang-bang and consists only of periods of maximum torque and glide (segments 1 and 3). Implementing this optimal solution in a real vehicle is impractical because it is both uncomfortable to the occupants and potentially disruptive to traffic. In an effort to resolve these issues, we propose to choose \( v(t_f) = v_c \). By doing so, the constant speed interval extends from \( t_1 \) to \( t_f \) and the third segment can be avoided. In other words the vehicle either accelerates with maximum torque or decelerates with engine off to a constant speed and cruises past the light. We note that this choice will be sub-optimal in most scenarios, but is practical and implementable in real world conditions. By reduced idling at red lights
the strategy still improves the fuel efficiency as demonstrated by the results of this paper.

The solution then reduces to the two following cases:

Case 1: The vehicle accelerates with maximum engine torque to reach a constant speed at time \( t_1 \) as depicted in Figure 4.3, therefore

\[
t - t_0 = \int_{v(t_0)}^{v(t)} \frac{dv}{u_{e_{\text{max}}}^2 - C_1 v^2 - C_2} \quad t_0 \leq t \leq t_1
\]  

(4.18)

The solution is

\[
v(t) = Q_1 \cdot \frac{Q_2 e^{2Q_1C_1(t-t_0)} - 1}{Q_2 e^{2Q_1C_1(t-t_0)} + 1} \quad t_0 \leq t \leq t_1
\]  

(4.19)

where \( Q_1 = \sqrt{\frac{u_{e_{\text{max}}}^2 - C_2}{C_1}} \), \( Q_2 = \left| \frac{Q_1 + v(t_0)}{Q_1 - v(t_0)} \right| \). At time \( t_1 \), the velocity reaches the constant value \( v(t_1) \), which satisfies Equation (4.19). The velocity trajectory should also satisfy the distance condition,

\[
d = x_{t_f} - x_{t_0} = \int_{t_0}^{t_1} v(t)dt + (t_f - t_1)v(t_1)
\]  

(4.20)

The solution is

\[
d = -Q_1(t_1 - t_0) + \frac{1}{C_1} \ln \left[ \frac{Q_2 e^{2Q_1C_1(t_1-t_0)} + 1}{Q_2 + 1} \right] + (t_f - t_1)v(t_1)
\]  

(4.21)

which will be used to solve for the only remaining unknown variable \( t_1 \).

Case 2: The engine is turned off or idles with minimum torque and the vehicle decelerates to a constant speed as shown schematically in Figure 4.4. There are situations when gliding alone cannot delay the vehicle enough for a green arrival. In these situations the lowest needed braking force is used to meet the boundary conditions. After
introducing braking input, the vehicle longitudinal dynamics becomes:

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= -C_1 v^2 - C_2 - u_b
\end{align*}
\] (4.22)

where \( u_b \) is deceleration due to braking. Integrating the velocity equation we get:

\[
v(t) = Q_3 \cdot \tan[Q_4 - C_1 Q_3 (t - t_0)]
\] (4.23)

where \( Q_3 = \sqrt{\frac{C_2 + u_b}{C_1}} \), \( Q_4 = \arctan\left(\frac{v(t_0)}{Q_3}\right) \). Integrating the velocity similar to (4.20), we obtain

\[
d = \frac{\ln[\sec(-Q_4)]}{C_1} - \frac{\ln[\sec(C_1 Q_3 t_1 - C_1 Q_3 t_0 - Q_4)]}{C_1} + (t_f - t_1)v(t_1)
\] (4.24)

Combining Equations (4.23) and (4.24), there are two unknown variables \( u_b \) and \( t_1 \).

For the purpose of fuel minimization, we choose the solution which gives the largest \( t_1 \) which maximizes the length of interval \([t_0, t_1]\) during which the engine is off or idling and consuming minimum fuel.

With these analytical solutions at hand, the procedure of the Speed Advisory Algorithm that we implement in our micro-simulations can be summarized as follows:
Step 1: At each time step $t_0$, obtain the current state $(x(t_0), v(t_0))$, determine the next traffic light and obtain the position of the next upcoming light $x_f$.

Step 2: Assume that the vehicle accelerates to the maximum velocity and maintains the speed till it reaches the intersection. Obtain the time $t_f$; this is the earliest possible time the vehicle can arrive at the intersection.

Step 3: Determine whether $t_f$ is in a green phase. If it is in a red phase, increase $t_f$ to the beginning of the next green phase.

Step 4: Given the current vehicle speed, determine if the vehicle needs to accelerate or decelerate to arrive at position $x_f$ at time $t_f$. Then use the corresponding analytical solution in Equations (4.21) or (4.24) to determine the velocity trajectory.

Step 5: Go back to Step 1 for next time step.

Note that the optimization problem is solved at every time step, so even when the vehicle motion is impeded by surrounding traffic, a new (sub)optimal velocity trajectory is obtained based on the latest position with respect to the traffic signal.

One may raise concern on the choice of $t_f$. In our algorithm the arrival time is set to be the earliest time the vehicle can pass the intersection, which is fixed. One may argue that choosing a later arrival time may achieve better fuel economy. While we acknowledge it, trading off travel time for one vehicle will influence other vehicles as well. Therefore we fix this condition and solve the optimal control problem based
It is essential to analyze the impact of this speed advisory system, both on SAS-equipped vehicles and on other conventional vehicles in mixed traffic conditions with many vehicles. This is achieved in a microsimulation environment as described in the next section. We are interested not only in a single intersection case but also in scenarios with a series of intersections. With multiple traffic signals ahead, the optimal solution changes frequently and cannot be calculated in real time when using computationally demanding algorithms such as dynamic programming. In the following sections, we will demonstrate the advantages of the analytical SAS approach we proposed in this paper.

4.3 The Simulation Environment

Simulations are conducted in Quadstone Paramics\textsuperscript{TM}, version V6.9.3. Paramics is a microscopic traffic simulation software that has been widely used in academic research and commercial fields. It is capable of simulating a large number of vehicles in a wide road network and allows users to program their own functions for different types of vehicles.

The major modules used in this paper are Modeller and Programmer. Modeller is the module for traffic network model creation, simulation animation, and results storage and display. All network attributes such as road geometry, number of lanes, traffic signal timings, origins and destinations, traffic demands, vehicle types and their percentages are set in the Modeller. Programmer allows users to implement their own algorithms. Many Application Programing Interface (API) functions can be used, such as QPO (overriding existing functions), QPX (extending existing functions), QPS (setting values to variables) and QPG (getting values from variables). All APIs
are written in C language.

In this paper, SAS is implemented in a number of vehicles by overriding their *leadspeed* and *followspeed* functions in Paramics default car following model. Our overriding function calculates and returns the reference speed at each simulation step such that idling at red is reduced. If the advised speed cannot be achieved due to other constraints, the vehicle will simply follow the speed generated by the Paramics default car-following model. After each run, all parameters such as time stamps, 3D coordinates, speed and acceleration trajectories can be stored. Figure 4.5 shows a screenshot of a running simulation where conventional vehicles are in white and SAS-equipped vehicles are in green.

![Figure 4.5: A screenshot of a Paramics simulation showing SAS equipped vehicles in green and conventional vehicles in white.](image)

Next we describe the standard car following model in Paramics used by conventional vehicles and then the lane change model adopted by all vehicles is explained.

### 4.3.1 The Car Following Model in Paramics

Paramics uses a standard car following model but enables editing several of its parameters. Each vehicle in Paramics has a target headway $h$. The default mean
value for target headway is 1 second, however this is adjustable by the user. The value of headway of each vehicle is also affected by certain parameters such as vehicle type, heading direction, aggressiveness, and reaction time. A vehicle can operate in 3 modes: acceleration, cruising, and braking.

When the the distance between the ego vehicle and the preceding vehicle exceeds a certain distance threshold, the vehicle will be in acceleration mode; the acceleration is set to be the maximum acceleration, here 2.5 \( m/s^2 \), till the ego vehicle reaches the road speed limit. When the ego vehicle detects that the preceding vehicle is braking, its acceleration is limited to a maximum of 1 \( m/s^2 \) for safety purposes.

When the the distance between the ego vehicle and the preceding vehicle falls below the distance threshold, the ego vehicle will be in cruising mode. In the cruising mode, the ego vehicle tries to maintain a desired gap with the preceding vehicle, represented by \( s_1 \) and calculated as:

\[
\begin{align*}
    s_1 &= h \Delta v \\
    \text{where } h &\text{ is the target headway and } \Delta v \text{ is the speed difference between the ego and the preceding vehicle. In aggressive driving setting, a shorter desired gap } s_2, \text{ is defined and followed:}
\end{align*}
\]

\[
\begin{align*}
    s_2 &= \frac{s_1^2}{d} < s_1
\end{align*}
\]

where \( d \) is the current distance between the vehicles. The acceleration needed to reach the desired separation is calculated as a function of the headway and velocity difference. The headway and velocity-difference phase space is divided into several regions as shown in Figure 4.6, and the acceleration in each region is calculated as follows:
where $k_1 = 1.0 \text{ s}^{-1}$ and $k_2 = 1.0 \text{ s}^{-2}$ are constants. More details of this policy can be found in [85] and [21].

4.3.2 The Lane Changing Model in Paramics

Lane changing in Paramics is determined by a gap-acceptance policy, and is based on the gaps with respect to preceding and the following vehicles in adjacent lanes. Lane changing is only allowed if these gaps would be sufficiently large. Suppose $Veh_0$ is the vehicle that is aiming to change lane, and $Veh_1$ and $Veh_2$ are vehicles in the target lane that are in front and behind of the position $Veh_0$ would occupy. A lane change maneuver will happen if the following gap condition is satisfied for a period.
of time, typically 3-6 seconds:

\[
g_i > (h_0 + \frac{\Delta v_i}{D_i} + h)v_i \quad i = 1, 2
\]  \hspace{1cm} (4.27)

Here \( g_i \) is the gap with respect to \( \text{Veh}_i \), \( h_0 \) is the minimum headway required for safety purposes, \( h \) is the target headway, \( D_i \) is the maximum deceleration of \( \text{Veh}_i \), and \( \Delta v_i \) is the relative speed with respect to \( \text{Veh}_i \).

### 4.4 Micro-simulation Case Studies

We create a number of simulation scenarios, and evaluate the impact of SAS at different congestion levels and with various penetration rates of connected vehicles. In all simulations the parameters of the vehicle longitudinal dynamics and fuel economy models are those shown in Table 4.1.

Table 4.1: Values of vehicle model parameters.

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Value</th>
<th>Unit</th>
<th>Coefficient</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m )</td>
<td>1200</td>
<td>kg</td>
<td>( A )</td>
<td>0.25</td>
<td>( m^2 )</td>
</tr>
<tr>
<td>( C_D )</td>
<td>0.35</td>
<td>–</td>
<td>( g )</td>
<td>9.8</td>
<td>( m/s^2 )</td>
</tr>
<tr>
<td>( \rho_a )</td>
<td>1.184</td>
<td>kg/m^3</td>
<td>( \mu )</td>
<td>0.015</td>
<td>–</td>
</tr>
<tr>
<td>( u^\text{max}_e )</td>
<td>2.5</td>
<td>m/s^2</td>
<td>( u^\text{max}_b )</td>
<td>2.9</td>
<td>m/s^2</td>
</tr>
<tr>
<td>( v^\text{max} )</td>
<td>80</td>
<td>km/h</td>
<td>( v^\text{min} )</td>
<td>10</td>
<td>km/h</td>
</tr>
<tr>
<td>( \alpha_0 )</td>
<td>0.1569</td>
<td>mL/s</td>
<td>( \beta_0 )</td>
<td>0.07224</td>
<td>mL/s</td>
</tr>
<tr>
<td>( \alpha_1 )</td>
<td>( 2.45 \times 10^{-2} )</td>
<td>mL/m</td>
<td>( \beta_1 )</td>
<td>( 9.681 \times 10^{-2} )</td>
<td>mL/s^2</td>
</tr>
<tr>
<td>( \alpha_2 )</td>
<td>( -7.415 \times 10^{-4} )</td>
<td>mL/s/m^3</td>
<td>( \beta_2 )</td>
<td>( 1.075 \times 10^{-3} )</td>
<td>mL/s^3</td>
</tr>
<tr>
<td>( \alpha_3 )</td>
<td>( 5.975 \times 10^{-5} )</td>
<td>mL/s/m^3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 4.4.1 Simulation Scenarios

The simulation is run in an urban corridor network. The path contains four signalized intersections. All signal timing plans are fixed and not coordinated with
others. The main street has links with three or four lanes with a total length of 2.203 kilometers (1.367 miles). The speed limit of each link is 80 km/h (49.7 mph). Conventional vehicles do not have prior access to traffic signal information and always try to reach the maximum road speed limit unless affected by nearby vehicles or traffic signals. For SAS-equipped vehicles, the minimum cruise speed is set to 2.78 m/s (10 km/h), and the maximum acceleration and deceleration are set to $2.5 \text{ m/s}^2$ and $2.9 \text{ m/s}^2$ respectively which are reasonable for passenger vehicles. Three different traffic demand levels (300, 600, and 900 vehicles per hour per lane) and seven different percentages (100%, 90%, 70%, 50%, 30%, 10%, 0) of SAS-equipped vehicles are considered; therefore 21 simulations are conducted. Note that the results are influenced by Paramics vehicle generating model and its parameters, therefore implementing our algorithm in other simulator such as Aimsun, SUMO may provide different results.

Five main parameters of vehicles or traffic are to be statistically analyzed. We evaluate fuel consumption, acceleration, velocity, total travel time, and traffic flow. The simulation results and relational behaviors are shown next.

### 4.4.2 Simulation Results

Figure 4.7 compares the average individual vehicle fuel consumption for different traffic demands and percentages of SAS-equipped vehicles. It can be seen that the fuel consumption of vehicles with and without SAS are well separated, SAS-equipped vehicles (three bottom curves) consume much less fuel than the ones without SAS (three top curves). This is due to fewer stops and closer to optimal operation of the engine. Another very interesting trend seen in Figure 4.7 is that with the increment of the percentage of SAS-equipped vehicles, conventional vehicles consume less fuel. In other words, SAS-equipped vehicles have a positive impact on the energy
efficiency of the entire mix of vehicles. With the increment of vehicles equipped with SAS, other conventional vehicles are more likely be blocked by a slower vehicle with SAS. By their simple car following strategy, such conventional vehicles may reduce the chance of stopping at intersections as well.

![Figure 4.7: Fuel consumption of vehicles with and without speed advisory system under different traffic demand levels and different penetration levels of equipped vehicles.](image)

The results for SAS-equipped vehicles indicate that as the traffic demand is at the demand level (900 veh/h/l), the fuel consumption of SAS-equipped vehicles is high and increases with penetration rates. It can be explained that at congested levels, slow downs and stops become more likely for all vehicles and SAS vehicles can not avoid them. It was unexpected to observe that 600 veh/h/l (mild congestion) resulted in better fuel economy than 300 veh/h/l (low congestion) for SAS vehicles. One explanation could be that in the mild congestion scenario, the vehicles are more likely to be affected by other vehicles and go at lower and more fuel efficient speeds.

In order to compare the effects of SAS, we further plot the position trajectories of multiple vehicles under same traffic conditions. Figure 4.8 shows trajectories from
Figure 4.8: Without SAS, vehicles experience more stops at each intersection.

Figure 4.9: Equipped with SAS, vehicles experience much fewer stops at each intersection and the trajectories are much smoother.

vehicles without SAS and Figure 4.9 shows trajectories from vehicles with SAS. We can clearly see that vehicles with SAS successfully avoid stops at intersections and the trajectories are much smoother.

Figure 4.10 shows acceleration trajectories of all vehicles and highlights the mean acceleration over position for different penetration levels and under the same traffic demand 600 veh/h/l. Plots on the left column are of conventional vehicles, while the right column belongs to SAS-equipped vehicles. The percentage of SAS-equipped vehicles increases from top to bottom. Comparing between columns, it can be seen that the mean acceleration of SAS-equipped vehicles is significantly less than the ones without SAS. Comparing among rows, it is shown that, for vehicles
Figure 4.10: Acceleration of vehicles with and without speed advisory system under 3 different penetration levels of equipped vehicles. The traffic demand was fixed at 600 veh/h/l. Accelerations of all vehicles are plotted in thin blue and the mean acceleration is plotted in thick red.

With SAS, the mean acceleration does not change much. The mean acceleration of conventional vehicles without SAS on the other hand, decreases with the increment of SAS-equipped vehicles.

Note that the maximum and minimum acceleration values are sometimes above the upper limit or below the lower limit used in optimal control calculations and listed in Table 4.1. This is due to the fact that vehicles are interacting and when in conflict with other vehicles, a vehicle may resort to higher acceleration or lower deceleration to avoid a collision.

Figure 4.11 shows velocity trajectories of all vehicles and highlights the mean velocity for different penetration levels and under the same traffic demand 600 veh/h/l. From the left column, it can be seen that the mean velocity of conventional vehicles
Figure 4.11: Velocity of vehicles with and without speed advisory system under 3 different penetration levels of equipped vehicles. The traffic demand was fixed at 600 veh/h/l. Velocities of all vehicles are plotted in thin blue and the mean velocity is plotted in thick red.

drops at four signalized intersections. In the right column, the velocity curves are more smooth indicating that SAS-equipped vehicles react to traffic signals in advance. Similar to the acceleration plots, the mean velocity of conventional vehicles has higher fluctuations than that of SAS-equipped vehicles. And with more SAS-equipped vehicles, the mean velocity of conventional vehicles drops. These acceleration and velocity trajectories can best explain the trends observed in fuel consumption in Figure 4.7.

We have thus shown that SAS-equipped vehicles improve the overall energy efficiency in mixed traffic by harmonizing the motion of conventional vehicles. This comes at a cost to overall traffic flow and vehicles’ average speed. With the increment of vehicles equipped with SAS, the cumulative counts, which can be considered as average traffic flows, drop slightly. To evaluate whether such a sacrifice is acceptable,
we cumulate vehicle counts for each intersection at different SAS penetration rates. The vehicle counts are gathered through virtual loop detectors embedded at each simulated intersection, and the data is stored every 1 seconds. The traffic demand is also fixed to 600 veh/h/l. Note that at the first intersection the traffic is saturated therefore the vehicle count are similar. The remaining three intersections are not congested. Comparative plots of average traffic flow for four intersections and three SAS penetration levels are shown Fig 4.12. According to these plots SAS has caused 2%-8% decrease in average traffic flow. For individual vehicles this impact is best shown by observing their travel times. Figure 4.13 shows travel time histograms of conventional and equipped vehicles under different SAS penetrations and when traffic demand is 600 veh/h/l. While travel time distribution of SAS-equipped vehicles remains almost the same, that of conventional vehicles moves to the right. The mean travel time of conventional vehicles increases about 45 seconds, or 16%. This could be an acceptable trade-off given the benefits of much increased energy efficiency and speed harmonization.

Figure 4.12: Average traffic flow at four different intersections. The traffic demand was fixed at 600 veh/h/l.
Figure 4.13: Histogram of travel times for conventional and SAS equipped vehicles. The traffic demand was fixed at 600 veh/h/l.

4.5 Conclusion

We have shown in this chapter, via 21 carefully arranged microsimulation case studies, that connected vehicles equipped with a speed advisory system have the potential to decrease their fuel consumption significantly by reducing idling at red lights. More interestingly we showed that even at relatively low penetration levels, the SAS equipped vehicles have a harmonizing effect on the motion of conventional vehicles, thus contributing to better energy efficiency of vehicles without a speed advisory system. The trade-off is a slight increase in travel times.

We formulated the speed advisory system as an optimal control problem and obtained the general structure of fuel optimal solution analytically. We demonstrated that the minimum fuel driving strategy is bang-bang in which the vehicle switches between acceleration with maximal engine torque and gliding with the engine turned off or idling until it arrives at a green light. In between these two extremes, sometimes the optimal trajectory includes periods of constant speed, but only at very low
velocities. We determined that a bang-bang solution, while most energy efficient for each vehicle, would cause discomfort to the vehicle occupants and could disrupt the traffic flow. To prevent a jerky ride, we constrained the velocity to a smoother profile which was still guided by our optimal results. This solution, while suboptimal, is implementable and still increases the energy efficiency by reducing idling at traffic signals.
Chapter 5

Predictive Cruise Control for Autonomous Vehicles: Combining predictors of preceding vehicle’s motion.

5.1 introduction

Autonomous vehicle functions or full vehicle autonomy could enable much safer, more comfortable, and fuel efficient driving. By relying on precision sensing and tight control they have the potential to outperform human drivers. For instance, they can act more smoothly in maintaining a safe distance to a preceding vehicle and as a result improve comfort and energy efficiency of the vehicle. The Adaptive Cruise Control (ACC) function that is available on many production vehicles today, does exactly the same thing. It uses a radar to detect the relative speed and position of the preceding vehicle and maintains a safe following gap. However, to the best knowledge
of the author, ACC systems in production vehicles or car following in autonomous cars, relies on instantaneous sensory information about the relative motion of the preceding vehicle and therefore engage reactively rather than proactively.

With proliferation of connected vehicles and connected road infrastructure systems, large datasets are becoming available of surrounding traffic, individual driving patterns, etc. Such information enables autonomous functions that act more proactively by predicting, over a short horizon, the motion of neighboring cars including the preceding vehicle. For example, an autonomous car following function can switch to its “cautious mode when it determines, from recent sensory data, that a preceding vehicle is being aggressively driven. Or in anticipation of a recurrent bottleneck, the car following function can smoothly slow down instead of a reactive and sudden deceleration.

In this chapter we present results on a Predictive Cruise Control system that takes into account probabilistic information about the motion of the preceding vehicle. A similar concept was first introduced in our group and was published in [111]. We build on this previous work and propose an enhanced approach for predicting the most likely trajectory of preceding vehicle. We propose to combine two predictors that take into account individual driving habits and traffic conditions: a Markov chain method to predict individual driving behavior, and an Expectation Maximization method to predict the trajectory of the front vehicle based on historical traffic patterns. Similar to [111] we solve the autonomous car following problem using a chance constrained model predictive control framework where inter-vehicle gap constraints are imposed probabilistically. We compare the performance of the proposed approach to methods that use only instantaneous sensory information in a number of simulated scenarios.

The rest of the chapter is organized as follows: First a quick overview of relevant research is given in Section 5.2. In Section 5.3 an MPC approach with chance
5.2 Literature Review

Adaptive Cruise Control (ACC) is now available on many vehicle models. A large body of research on ACC exists [99]. A vehicle equipped with ACC can detect the preceding vehicle’s position and velocity, and maintain a safe desired distance or headway. However, current ACC systems rely on instantaneous sensor information. Although the reaction time of an onboard computer is shorter than a human drivers, the motion based on instantaneous information can still fluctuate, especially in complex traffic situations. One way to solve this problem is to resort to Cooperative Adaptive Cruise Control (CACC) [78] [101] [87]. CACC allows vehicles to communicate with each other and improve their collective performance. However, CACC requires vehicle connectivity, some level of autonomy, and a complex coordination scheme. In the near future, it is unlikely to replace all road vehicles with these capabilities, and the traffic will remain mixed. With the development of on-board data acquisition devices, on the other hand, more and more historical data is stored and becomes available to every participant in traffic. Patterns can be learned from those datasets, which enable more precise short term prediction of each vehicle’s behavior (human driven and autonomous vehicles) as well as traffic. Such predictions can be fed into an optimal control system to control the ego vehicle’s motion.

There is a large body of literature on predicting an individual vehicle’s behavior. One direction is to predict based upon static driver characteristics. For example, in
[59], the authors predict driving aggressiveness based on personality, age, and power of the vehicle. Another way is to learn from historical driving data. Markov chain is a powerful tool for short term prediction. The fundamental property of Markov chain is that the next state value only depends on the current state and is independent from any previous state. With sufficient training data, one can obtain a state transition matrix and use it to predict future states. In [66], the authors use a Markov model to predict driver’s intended actions. In [27], the authors use a Markov chain to predict lane changing intention of a neighboring vehicle. Our group has proposed a Markov chain Monte Carlo method to sample from the transition matrix to predict the preceding vehicle’s trajectory [111]. Similar ideas can also be seen in [74] in which the authors further take road grade into consideration as a component of their Markov model state.

Moreover, a vehicle’s motion is also affected by the surrounding traffic. The general traffic pattern is recurrent [93] thus predictable. On the other hand, traffic is not evenly distributed along the road; it is reasonable to expect high density and delay around intersections and faster traffic at mid-links [103]. Considering the recurrent pattern of upcoming traffic, one can use historical traffic data to predict traffic conditions for different sections of a road. For example, our group has proposed a method which only uses sparse probe vehicle data to estimate the most likely trajectory of a vehicle [105].

To the authors best knowledge, there is a gap in the literature when it comes to taking into account both individual driving behavior and traffic patterns. One main contribution of this dissertation is prediction of the preceding vehicle’s trajectory based on both, the vehicles driving pattern and the upcoming traffic. The challenge is how to combine the predictors statistically based on the available training dataset. A number of papers address this kind of problem such as [4][98][61][41].
Having the prediction results, different control schemes can be used for ego vehicle car following. Model Predictive Control (MPC) can be a very effective method [18] because of its anticipatory nature. Such an MPC-based car following model uses predictions of the motion of the preceding vehicle and considers safety gap constraints, to determine the optimal trajectory of ego vehicle. Note that the predictions need not be deterministic. In fact, a prediction represented by a probability distribution could be used to handle partial and imperfect information of the motion of preceding vehicle. This in turn would introduce probabilistic constraints to be enforced. In this work, we adopt a chance constrained MPC approach to handle probabilistic predictions and constraints [11]. The details are provided next.

5.3 Chance Constraints Model Predictive Control

5.3.1 Problem statement

The predictive cruise control problem describes the situation in which the ego vehicle is following the preceding vehicle. The ego vehicle detects the preceding vehicle and tries to maintain a desired distance, which is often a function of two vehicles relative velocity, that is:

\[ D_{\text{desired}} = D_{\text{min}} + Tv \]  

(5.1)

where \( D_{\text{desired}} \) is the desired distance between the ego vehicle and the preceding vehicle, \( D_{\text{min}} \) is the minimum required safety distance, \( T \) is the reaction time (also known as headway), and \( v \) is the speed of the ego vehicle.
5.3.2 Vehicle Kinematic Model

The kinematics of a vehicle longitudinal motion is modeled, along with a first order lag between the acceleration command input and the vehicle’s acceleration. The state space model is described by:

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= a \\
\dot{a} &= -\frac{1}{\tau}a + \frac{1}{\tau}u
\end{align*}
\]

where \( u \) is the acceleration command input and \( \tau \) is a time constant. The state vector can be written as \( Z = [x, v, a]^T \). After discretization, the state space model becomes:

\[
Z_d(k+1) = A_dZ_d(k) + B_du_d
\]

where \( A_d \) and \( B_d \) are discretized state space matrices.

5.3.3 Predictive cruise control problem formulation

We propose to use a Model Predictive Control (MPC) approach for controlling the speed in predictive cruise control mode. Given a cost function \( J \), MPC solves a finite moving horizon optimization problem at each time step and only applies the current time step control input to the system. At the next step the horizon is moved forward by one step and the process is repeated. Let \( Z_d(i+k|k) \) denote the \( i \)th step prediction at time step \( k \). With a horizon of \( N_c \) time steps, a sequence \( U = [u(k|k), u(1+k|k), ..., u(N_c - 1 + k|k)]^T \) is found which minimize the cost function. But only the current time step control input \( u(k|k) \) is applied. As the system moves one time step forward, another finite horizon optimization problem needs to
be solved, with the current states as initial conditions. States and control inputs have constraints.

In this paper, the cost function is defined as:

\[
J = \sum_{i=0}^{N_c-1} a^2(i + k|k) + q(r(N_c + k|k) - x(N_c + k|k) - T \dot{x}(N_c + k|k) - D_{min})
\]

(5.4)

where \(r\) and \(x\) denote the position of the preceding and ego vehicles respectively, \(q\) is a penalizing coefficient. Penalizing acceleration in the cost function is meant to improve ride comfort. The last term penalizes the gap between the two vehicles at the end of control horizon and encourages maintaining a safe and reasonable distance between the two vehicles. It has been shown that this cost function also reduces fuel consumption [111]. Hard constraints on vehicles states and on the following distance must be enforced at each step in time. For example, at any time, the acceleration should be between its maximum and minimum values, that is:

\[
a_{min} \leq a(i + k|k) \leq a_{max}
\]

(5.5)

The following distance between two vehicles must also be constrained:

\[
D_{min} \leq r(i + k|k) - x(i + k|k) - T \dot{x}(i + k|k) \leq D_{max}
\]

(5.6)

where \(D_{min}\) is the minimum allowable gap for safety reasons and \(D_{max}\) prevents large gaps that reduce road capacity.

If the preceding vehicle’s trajectory is known, this becomes a standard linear MPC problem, and can be efficiently solved as a quadratic program. However, at time step \(k\), \(r(i+k|k)\) is not available and must be predicted. We propose two probabilistic
prediction methods based on different two data sources in Sections 5.4 and 5.5.

We can define the constraints that are satisfied with a given probability. Given a probability $\alpha$ (e.g. $\alpha = 95\%$), we can state the minimum distance constraint in (5.6) as a chance constraint:

$$Pr(x(i + k|k) + T\dot{x}(i + k|k) \leq r(i + k|k) - D_{\text{min}}) \leq 1 - \alpha$$  \hspace{1cm} (5.7)

which means that the chance of violating the constraint should be less than $1 - \alpha$. Denoting by $r^{1-\alpha}$ the position where the cumulative distribution function value is equal to $1 - \alpha$. Then the probabilistic constraint can be converted to a deterministic constraint:

$$x(i + k|k) + T\dot{x}(i + k|k) \leq r^{1-\alpha} - D_{\text{min}}$$  \hspace{1cm} (5.8)

Similarly, we can obtain the maximum distance constraint at a given probability $\beta$, therefore

$$r^{\beta} - D_{\text{max}} \leq x(i + k|k) + T\dot{x}(i + k|k) \leq r^{1-\alpha} - D_{\text{min}}$$  \hspace{1cm} (5.9)

With this transformation of the probabilistic constraints to deterministic ones, we end up with a standard MPC problem. The challenge is obtaining a probability distribution for the position of the preceding vehicle at future steps in time which is the focus of the next Section.
5.4 Vehicle Trajectory Prediction Based on Historical Probe Data

In this section we propose to use vehicle probe data to estimate travel time statistics across different segments of the road and to use these probabilistic estimates to predict the motion of a preceding vehicle in traffic. This prediction method is built on our previous work in [103] and [105], in which we proposed an approach to reconstruct the most likely trajectory along an arterial road between two consecutive probe data updates. The example data source we use is a sparse data feed from transit buses in the city of San Francisco. Each bus in San Francisco has a GPS device which reports its location, speed, and time stamp every 200 meters or 90 seconds whichever comes first. The position and time of an update is at random. In [103], we proposed to divide a road into smaller segments to find out each segment travel time statistics, as shown in Fig 5.1.

The travel time for each segment is denoted by $tt^j_{x_i}$ for the $j$th bus at the $x_i$th segment. For each bus, two consecutive updates provide a travel time observation, which is denoted by $tt^j_{[x_{a_1}, x_{a_n}]}$ from the beginning segment $x_{a_1}$ to the end segment $x_{a_n}$.
The sum of each segment travel time should equal the total observed travel time, i.e.

$$tt^j_{[x_{a1}, x_{an}]} = \sum_{i=a_1}^{a_n} tt^j_{x_i} \quad (5.10)$$

The approach is to allocate the total travel time into each segment such that its likelihood is maximized. We proposed an Expectation Maximization (EM) approach to iteratively allocate the observed travel time into each segment and subsequently calculate the statistics of travel time for each segment. We are assuming that the travel time for each segment has a Gaussian probability density function. Therefore, after each allocation we can calculate its likelihood. And we reallocate the segment travel time to maximize this likelihood. After initialization, the EM steps can be represented as:

**E step**: Having allocated travel times from different buses, for each segment $x_i$, we calculate the mean $\mu_{x_i}$ and variance $\sigma_{x_i}$. Then the probability density for $tt^k_{x_i}$ can be calculated as:

$$p(tt^k_{x_i} | \mu_{x_i}, \sigma_{x_i}) = \frac{1}{\sqrt{2 \pi \sigma_{x_i}}} \cdot e^{-\frac{(tt^k_{x_i} - \mu_{x_i})^2}{2 \sigma_{x_i}^2}} \quad (5.11)$$

**M step**: In this step, we go back to each travel time observation and reallocate the segment travel time such that the likelihood function is maximized. Suppose an update pair starts at segment $x_{a_1}$ and ends at $x_{a_n}$; its log likelihood function can be represented by:

$$log[p(tt^k_X | \mu_X, \Sigma_X)] = \sum_{i=a_1}^{a_n} log[p(tt^k_{x_i} | \mu_{x_i}, \sigma_{x_i})] \quad (5.12)$$

where $X = [x_{a_1}, x_{a_2}, ..., x_{a_n}]^T$, $\mu_X = [\mu_{x_{a_1}}, ..., \mu_{x_{a_n}}]^T$. 

92
and \( \Sigma_X = \begin{bmatrix} \sigma_{x_1}^2 & 0 & \ldots & 0 \\ 0 & \sigma_{x_2}^2 & \ldots & 0 \\ 0 & 0 & \ldots & \sigma_{x_n}^2 \end{bmatrix} \).

This problem is in fact a constrained linear quadratic programming problem. The solution to this problem is the maximum likelihood segment travel time allocation for each observation. In Figure 5.2 we show the segment travel time evolution and RMS (Root Mean Square) errors through iterations.

Note that the travel time statistics is sensitive to the time period of day. We choose historical data within certain time periods to obtain the results. Furthermore, queue patterns can also be learned and once traffic signal timings become available, stop position and duration can be estimated as shown with more detail in [105].

Having obtained the mean segment travel times along the path of interest, and given a start time, we can predict a trajectory by assuming that the vehicle is following the mean travel time for each segment. If a lower and upper probability thresholds are set, the upper and the lower trajectory boundaries can be obtained. A prediction of the position probability distribution at the end of the horizon can

Figure 5.2: Segment travel time and RMS error evolution of our EM algorithm in [103].
then be obtained. An example is shown in Fig 5.3. The predictor requires a starting update \((T(x_{a_1}), x_{a_1})\) as the input.

5.5 Vehicle Trajectory Prediction Based on Individual Driving Data

In [74] and in our previous work [111], a Markov chain based approach is used to predict the future position of a car. The approach uses historical driving data to train a Markov model. By counting historical occurrences, probabilities are assigned to the transition between two states (i.e. preceding vehicle velocity) from consecutive time steps. The assumption is that the next step’s state only depends on the current step’s state and is independent from the previous steps. The element \(p(i, j)\) in the transition matrix represents the probability of reaching the state \(j\) in the next step, given the current state \(i\). Previous work in our group [111] only included velocity as the state, but more precise predictions can be made if acceleration is also included as a state. However, including acceleration will increase the dimension of
the transition matrix by one, and requires a much larger and richer dataset. To
circumvent this challenge we define the Markov state as $S_i = [v_i, \ acc_i]^T$, where $v_i$
is the discretized velocity and $acc_i$ is a categorical variable describing whether the
vehicle is accelerating, decelerating, or cruising. That is, $v_i \in \{0, 1, \ldots, 15\} m/s$ and
$acc_i \in \{accelerating, decelerating, cruising\}$. In the training procedure, we label the
training data by the three acceleration categories. With these newly defined states,
the Markov chain requires three times as much data as in [111] to be trained with,
but does much better in predicting future states. The transition matrix, denoted by
$P_{N_v}$ is a $N_v \times N_v$ matrix, where $N_v$ is the number of discretized states.

With the current state and the transition matrix, the position distribution
within a next finite horizon can be obtained using a Monte Carlo sampling technique.
Through Monte Carlo sampling, a sequence of state realizations can be generated. For
each velocity sample sequence the position at the $i$th future step can be calculated:

$$r(n + k|k) = r(k) + \frac{1}{2} \sum_{i=0}^{n-1} (v(i + k|k) + v(i + 1 + k|k))\Delta t(i)$$

(5.13)

for which the probability is known. With sufficient number of samples [95], a dis-
tribution can be estimated from the generated histogram, assuming that a Gaussian
distribution. Figure 5.4 shows an example of histogram of the preceding vehicle po-
position at the end of a 10 step horizon.

5.6 Combination of predictors

In reality it is common that one predictor outperforms the others in a certain
situation while worse in other situations. Combination of multiple predictors can
improve the prediction accuracy as shown in the literature [61] and [41]. In previous
sections we have proposed two different predictors: predictor 1 is built on historical probe vehicles data, and predictor 2 is derived from individual driving behavior data. The outputs of these two predictors are position distributions, both of which are assumed to be Gaussian. However, they are based on different data sources, and their estimation conditions and assumptions are different.

One way to combine Gaussian distributions is to form a Gaussian Mixture Model (GMM) [102]. The probability density function of a GMM can be represented by:

$$P(x|\pi, \mu, \Sigma) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i)$$ (5.14)

where $\mathcal{N}$ represents a Gaussian distribution component, $K$ is the number of components, $\mu_i$, $\Sigma_i$, and $\pi_i$ are the mean, the covariance matrix, and the weight of the $i$th component. The weights are such that,
We propose to assign the weight assigned to each predictor based on its performance. When a predictor is performing better, a higher weight should be assigned to it in the GMM. Intuitively, predictor 1 is expected to predict better the stop positions and durations close to an intersection, and when the traffic is heavy. Predictor 2 is expected to perform better at mid-links since the vehicle is less affected by other
Figure 5.7: Weights of the predictors over velocity.

vehicles and its behavior is more likely to be dominated by individual driver style. To test this hypothesis, we evaluate the prediction errors against the ground truth trajectory for each predictor separately. Here we select 7 ground truth trajectories as training data. In the future when more trajectories are collected as training data, the training results could improve. Figure 5.5 and Figure 5.6 show the samples and the average prediction errors versus velocity. It could be concluded from these figures that, on average, predictor 1 performs better than predictor 2 when the velocity is close to zero. Therefore, we assign higher weights to predictor 1 at low speeds. At higher speeds, predictor 2 performs better than predictor 1 and receives a larger weight. Suppose at position $x_i$, the average error of predictor 1 and predictor 2 are denoting by $e_1$ and $e_2$ respectively; we calculate their weights $\pi_1$ and $\pi_2$ by:

$$
\pi_1 = \frac{|e_2|}{|e_1| + |e_2|},
$$

$$
\pi_2 = \frac{|e_1|}{|e_1| + |e_2|}
$$

The assigned weights are shown in Figure 5.7. The average prediction errors
shown in Figure 5.6 are consistent with our hypothesis. Note that the errors of Predictor 1 are relatively large which may be due to the fact that currently we only have ground truth data from only 15 vehicle trajectories. The lack of training data may explain the prediction inaccuracy especially when accelerating and decelerating.

In the next simulation study we will implement the proposed combined predictor to obtain probabilistic predictions of the motion of a preceding vehicle and solve a chance constrained MPC to find the best trajectory of the ego vehicle. To demonstrate the effectiveness of the proposed scheme, we use a classic Proportional Derivative (PD) control car following model as a benchmark to compare with. All coefficients of this model are adopted from [23]. Also a best case scenario is simulated with the assumption that the preceding vehicle’s trajectory is predetermined; a standard MPC is solved to determine the best trajectory that can be theoretically achieved.

5.7 Case Study and Results

The scenarios are designed under the following simulated situation: an autonomous vehicle is following a preceding bus on Van Ness Avenue in San Francisco. The path consists of multiple intersections, of which traffic signal timings are available. The ego autonomous vehicle predicts the preceding bus’s trajectory by using the combination of two predictors discussed in previous sections. The training data for Predictor 1 is from 6 months of aggregated historical probe bus data, while training data for Predictor 2 is from multiple high frequency GPS ground truth data gathered on the same route in San Francisco as reported in [105].

To evaluate the effectiveness of the approach, multiple high frequency trajectory ground truth data are selected as preceding vehicle’s trajectories and simulations
are conducted in different scenarios. At any time, the ego vehicle predicts the preceding vehicle’s trajectory within a finite horizon and solves an MPC problem with chance constraints.

The first scenario is shown in Figure 5.8, where the preceding vehicle experienced a stop and joined the queue close to the intersection. The results show that the ego vehicle trajectory based on the combined prediction results is similar to the one based on Predictor 2, and is closer to the trajectory based on MPC with perfect information. However, at the end of the trajectory, the combination based trajectory
passes the probe data based trajectory.

The second scenario is shown in Figure 5.9, where the vehicle experienced a stop and joined the queue away from the intersection. The results are similar to the previous case. Note that the strategy is minimizing the sum of accelerations while maintaining a reasonable distance with respect to the preceding vehicle. Although the trajectory based on MPC with perfect information is further away from the preceding vehicle, it is the best trajectory to follow.

Figure 5.10 shows the comparison of average costs, that are being minimized, of each predictor based on 6 simulation cases. The first column is the theoretical best case scenario. First, we can conclude that using prediction results, the ego vehicle can achieve much better performance compared with a classic PD control strategy. Furthermore, we can see that the cost is reduced by combining the two predictors, which demonstrates the benefits of assigning different weights to these predictors based on the situation.

Note that the cost in Equation (5.4) does not have a physical meaning. However, it reflects the improvement in driving comfort (and perhaps fuel consumption).
5.8 Conclusion

In this chapter, a Predictive Cruise Control approach is proposed. The approach takes the position distribution prediction of the preceding vehicle as an input, and minimizes a car following cost function with probabilistic inter-vehicle constraints in a receding horizon manner. Two predictors are introduced: Predictor 1 is based on historical probe vehicle data and reflects the effect of recurrent traffic conditions, and Predictor 2 is based on individual driver behavior data. The distributions of predictors are combined as a Gaussian Mixture Model based on their prediction performance in training data sets. Simulation results show that the combined predictor achieves reduced car-following costs.
Chapter 6

Summary of the Dissertation

6.1 Summary of Contributions

The main contributions of this dissertation can be divided into estimation and control of autonomous/connected vehicles. Machine learning and control algorithms are designed and evaluated. Detailed contributions for each chapter are listed:

In chapter 2, a travel time distribution estimation method for freeway is proposed.

1. The approach uses the Link-Node Cell Transmission Model (LN-CTM) to model traffic and provides a probability distribution for travel time.

2. The probability distribution is generated using Monte Carlo simulation and the Online Expectation Maximization clustering algorithm.

3. Results show that the approach is able to generate an accurate multimodal distribution for travel time.

In chapter 3, a Speed Advisory System is proposed and its influence to the mixed traffic is analyzed. The contributions are:
1. Speed Advisory System (SAS) is proposed in which vehicles can utilize upcoming traffic signal information for reducing idling at red lights.

2. We formulate the problem and obtain the analytical solutions for fuel minimal driving strategy.

3. We show that SAS equipped vehicles improve energy efficiency of conventional vehicles as well.

In chapter 4, an approach of reconstructing trajectories between probe vehicle data updates is proposed. The contributions are:

1. We propose to divide the path into smaller segments, and an EM algorithm is used to allocate travel time to each segment.

2. We allocate travel time observations and delays to each segment such that the likelihood is maximized.

3. Our approach can estimate stop position and duration at intersection accurately using the shock wave theory.

4. We gathered ground truth data to compare with the results which demonstrates the effectiveness of our approach.

## 6.2 Dissemination of the Dissertation

### Journals


Conferences


Bibliography


[70] Grant Mahler, Andreas Winckler, Alireza Fayazi, Martin Filusch, and Ardalan Vahidi. System architecture and experimental results on 4g/lte communication of traffic signal state to connected vehicles for improving energy efficiency on arterial roads. (in review) Intelligent Transportation Systems, IEEE Transactions on, 2015.


