8-2018

Connected and Autonomous Vehicles Applications Development and Evaluation for Transportation Cyber-Physical Systems

Md Mizanur Rahman
Clemson University, mizanbuet@clemson.edu

CONNECTED AND AUTONOMOUS VEHICLES APPLICATIONS DEVELOPMENT AND EVALUATION FOR TRANSPORTATION CYBER-PHYSICAL SYSTEMS

A Dissertation
Presented to
the Graduate School of
Clemson University

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

by
Md Mizanur Rahman
August 2018

Accepted by:
Dr. Mashrur Chowdhury, Committee Chair
Dr. Jim Martin, Co-advisor
Dr. Amy Apon
Dr. Hsein Juang

ABSTRACT

Cyber-Physical Systems (CPS) seamlessly integrate computation, networking and physical devices. A Connected and Autonomous Vehicle (CAV) system in which each vehicle can wirelessly communicate and share data with other vehicles or infrastructures (e.g., traffic signal, roadside unit), requires a Transportation Cyber-Physical System (TCPS) for improving safety and mobility, and reducing greenhouse gas emissions. Unfortunately, a typical TCPS with a centralized computing service cannot support real-time CAV applications due to the often unpredictable network latency, high data loss rate and expensive communication bandwidth, especially in a mobile network, such as a CAV environment. Edge computing, a new concept for the CPS, distributes the resources for communication, computation, control, and storage at different edges of the systems. TCPS with edge computing strategy forms an edge-centric TCPS. This edge-centric TCPS system can reduce data loss and data delivery delay, and fulfill the high bandwidth requirements.

Within the edge-centric TCPS, Vehicle-to-X (V2X) communication, along with the in-vehicle sensors, provides a 360-degree view for CAVs that enables autonomous vehicles' operation beyond the sensor range. The addition of wireless connectivity would improve the operational efficiency of CAVs by providing real-time roadway information, such as traffic signal phasing and timing information, downstream traffic incident alerts, and predicting future traffic queue information. In addition, temporal variation of roadway traffic can be captured by sharing Basic Safety Messages (BSMs) from each vehicle through the communication between vehicles as well as with roadside infrastructures (e.g., traffic signal, roadside unit) and traffic management centers. In the early days of CAVs,

data will be collected only from a limited number of CAVs due to a low CAV penetration rate and not from other non-connected vehicles. This will result in noise in the traffic data because of low penetration rate of CAVs. This lack of data combined with the data loss rate in the wireless CAV environment makes it challenging to predict traffic behavior, which is dynamic over time. To address this challenge, it is important to develop and evaluate a machine learning technique to capture stochastic variation in traffic patterns over time.

This dissertation focuses on the development and evaluation of various connected and autonomous vehicles applications in an edge-centric TCPS. It includes adaptive queue prediction, traffic data prediction, dynamic routing and Cooperative Adaptive Cruise Control (CACC) applications. An adaptive queue prediction algorithm is described in Chapter 2 for predicting real-time traffic queue status in an edge-centric TCPS. Chapter 3 presents noise reduction models to reduce the noise from the traffic data generated from the BSMs at different penetration of CAVs and evaluate the performance of the Long Short-Term Memory (LSTM) prediction model for predicting traffic data using the resulting filtered data set. The development and evaluation of a dynamic routing application in a CV environment is detailed in Chapter 4 to reduce incident recovery time and increase safety on a freeway. The development of an evaluation framework is detailed in Chapter 5 to evaluate car-following models for CACC controller design in terms of vehicle dynamics and string stability to ensure user acceptance is detailed in Chapter 5.

Innovative methods presented in this dissertation were proven to be providing positive improvements in transportation mobility. These research will lead to the real-

world deployment of these applications in an edge-centric TCPS as the dissertation focuses on the edge-centric TCPS deployment strategy. In addition, as multiple CAV applications as presented in this dissertation can be supported simultaneously by the same TCPS, public investments will only include infrastructure investments, such as investments in roadside infrastructure and back-end computing infrastructure. These connected and autonomous vehicle applications can potentially provide significant economic benefits compared to its cost.

DEDICATION

To my beloved parents, my wife and my family for whom my whole life pertains.

ACKNOWLEDGMENTS

To enlist the entire number of persons to whom I am in debt for, might not be possible. Therefore, in a few words, I would like to say thank you to those who have made a mark on me, and this research in particular.

I would like to thank my advisor, Professor Mashrur Chowdhury, for his faith in me. His sheer brilliance, sound technical insights and constant support were highly inspiring. His insight and consultation have guided me to face the challenges throughout my graduate student life. I feel privileged to know him and to be a part of his research group.

I would like to acknowledge my co-advisor, Dr. Jim Martin, for his guidance and support in the connected vehicles research of this dissertation. I would also like to acknowledge my committee members, Dr. Amy Apon and Dr. Hsein Juang for the support and time in reviewing the dissertation.

Without the support and encouragement of my family, and in particular my parents Md Sirajul Islam and Roshida Khatun, this work would not have been possible. My beloved wife has always been my motivation to push forward. I also would like to mention my friends back in Bangladesh, the United States, and Clemson, without whom, this journey would have been difficult.

I would like to conclude by acknowledging the South Carolina Department of Transportation, BMW, the National Science Foundation, and University Transportation Center at Clemson University for their financial support during the entire research period.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

List of Figures (Continued) Page

List of Figures (Continued) Page

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

Cyber-Physical Systems (CPS) seamlessly integrate computation, networking and physical devices. Specifically, a Connected and Autonomous Vehicle (CAV) system in which each vehicle can wirelessly communicate and share data with other vehicles or infrastructures (e.g., traffic signal, roadside unit), requires a Transportation Cyber-Physical System (TCPS) for improving safety and mobility, and reducing greenhouse gas emissions.

Unfortunately, a TCPS with a centralized computing service cannot support real-time CAV applications due to the often unpredictable communication network latency, and high data loss rate and expensive bandwidth, a difficulty that is particularly characteristic of mobile networks, such within the CAV environment (NSF, 2017). Edge computing, however, is a new concept that can support such applications in that the communication, computation, control and storage resources are placed in a distributed manner at different edges within the CPS (NSF, 2016; Rabinovich et al., 2004). This CPS with this specific edge computing strategies is thus used to form an edge-centric TCPS, a system that is quite effective in reducing data loss and data delivery delays while ensuring the high bandwidth requirements (Rayamajhi et al., 2017; Chowdhury et al., 2018).

The addition of wireless connectivity would improve the operational efficiency of CAVs by providing real-time roadway information, such as traffic signal phasing and timing information, downstream traffic incident alerts and future traffic queue information.

In addition, temporal variation of roadway traffic can be captured by sharing Basic Safety Messages (BSMs) from each vehicle through the communication between vehicles as well as with roadside infrastructures (e.g., traffic signal, roadside unit) and traffic management centers. During the early deployment of CAVs, data will be acquired from only a limited number of CAVs (i.e., with a low CAV penetration rate) and not from other non-connected vehicles. Moreover, the data loss rate in the wireless CAV environment contributes to the unavailability of data from the limited number of CAVs. The lack of such limited CAV data in turn greatly increases the challenge of predicting traffic behavior, which changes dynamically over time. Therefore, the development of a machine learning technique to capture the stochastic variation of traffic patterns over time is necessary.

Furthermore, the noise of traffic data such as travel speed and space headway between vehicles could reduce the prediction accuracy of the machine learning model given the projected limited penetrations of connected vehicles. Therefore, improving the prediction accuracy through the use of these noisy data necessitates the development of a traffic prediction model combined with a noise reduction model. Such a model will be most useful in the accurate prediction of traffic data in real-time for both route planning and scheduling to reduce travel times, for assessing traffic conditions as they unfold and for optimizing a vehicle's energy resources to reduce fuel consumption (Ma et al., 2015; Hua et al., 2017; Ren et al. 2017; Ma et al., 2009; Ma et al., 2012; Khan et al., 2017).

In a TCPS, efficient incident management systems have both relieved congestion with the concurrent benefits in terms of economic savings, energy conservation, and improved environmental conditions, and health and safety aspects (Chowdhury et al., 2007). Implementing a traffic incident management (TIM) plan, the characteristics of which are real time detection and verification scheme, and regularly updated data in terms of response (e.g., tow truck arrival), scene management and traffic control, and quick clearance and recovery activities can clearly ease such congestion. Therefore, developing and evaluating a dynamic routing application that can support a TIM scheme is most important.

Any substantial level of user acceptance of a CAV system, Cooperative Adaptive Cruise Control (CACC) for example, requires the replication of the driving experience in a CACC equipped-vehicle without compromising driver comfort (Islam et al., 2018; Jones, 2016; Wang and Wang 2015). Thus, a key factor in any CACC systems design entails using driver models that represent driver car-following behaviors to achieve user acceptance of a CACC application. “User acceptance” can be defined as the safety and comfort of the occupant in the CACC vehicle in terms of acceptable vehicle dynamics (i.e., the maximum acceleration or deceleration) and string stability (i.e., the fluctuations in the vehicle’s position, speed, and acceleration). Both the user acceptable vehicle dynamics and string stability are critical for all safety-centered scenarios (e.g., incident management). Hence, it is important to develop an evaluation framework in terms of vehicle dynamics and string stability to investigate driver car-following models to inform CACC designs.

The simultaneous support of multiple connected and autonomous vehicles applications detailed in this dissertation by the same TCPS means that the only public investments required are those concerning infrastructure (e.g. investments in roadside units and backend computing platforms). Clearly, the limited need for such support in connected and autonomous vehicle applications can potentially provide significant economic benefits compared to its cost (He et al., 2012).

1.2 Research Objectives

The focus of this dissertation entails the development and evaluation of various applications relating to the CAVs, specifically adaptive queue prediction, traffic data prediction, dynamic routing and CACC applications, in an edge-centric TCPS. The followings are the objectives of this dissertation.

i. The first objective is to develop an adaptive traffic queue prediction algorithm to predict real-time traffic queue status in an edge-centric TCPS (Chapter 2).

ii. The second objective is to investigate noise reduction models to reduce the noise of the traffic data generated from the Basic Safety Messages (BSMs) for different penetration of CVs and evaluate the performance of the speed and space headway prediction model using the resulting filtered data (Chapter 3).

iii. The third objective is to evaluate a dynamic routing application in a connected vehicle environment to reduce incident recovery time and increase safety on a freeway (Chapter 4).

iv. The fourth objective is to develop an evaluation framework for the application of a driver car-following behavior model in a CACC controller design and evaluate the framework for the application of a driver car-following behavior model in a CACC controller design (Chapter 5).

The ultimate aim of this research is to contribute to the understanding of CAV applications under TCPS by accomplishing the corresponding objectives as discussed above.

1.3 Dissertation Outlines

This dissertation is organized into six chapters. Chapter 1 describes an overview of the research problem and objectives of this research. Chapter 2 discusses the development and evaluation of adaptive queue prediction algorithm for an edge-centric TCPS. Real-time traffic data prediction model is presented in Chapter 3. Chapter 4 presents the development of an evaluation framework for car-following models to ensure user acceptance for CACC controller design. A dynamic routing application in a CAV environment to reduce recovery time and increase safety on a freeway is presented in Chapter 5. Chapter 6 concludes the dissertation and discusses the implementation strategies of CAV applications in a TCPS.

1.4 References

Chowdhury, M., Rahman, M., Rayamajhi, A., Khan, S., Islam, M., Khan, Z., & Martin, J. (2018). "Lessons Learned from the Real-World Deployment of a Connected Vehicle Testbed," *In the Proceeding of 97th Annual Meeting of the Transportation Research Board*, Washington, D.C., January 2018.

Chowdhury, M., Fries, R., Ma, Y., Dunning, A., Hamlin, A. C., Ozbay, K., & Sarasua, W. (2007). "Benefit–Cost Analysis of Accelerated Incident Clearance." *Final Report. Prepared for the Federal Highway Administration and the South Carolina Department of Transportation*, FHWA-SC-07-01.

He, Y., Chowdhury, M., Ma, Y., & Pisu, P. (2012). "Merging mobility and energy vision with hybrid electric vehicles and vehicle infrastructure integration," *Energy Policy*, vol. 41, 599–609.

Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z., & Zhang, H. (2017). “Traffic Prediction Based on Random Connectivity in Deep Learning with Long Short-Term Memory.” *arXiv preprint arXiv:1711.02833*.

Islam, M., Li, F., Rahman, M., Islam, S., Chowdhury, M., & Wang, Y. (2018). “Human-Centered Time Headway Controller for Cooperative Adaptive Cruise Control (CACC).” *In the Proceeding of 97th Annual Meeting of the Transportation Research Board*, Washington, D.C., January 2018. (No. 18-06472).

Jones, S. (2013). Cooperative adaptive cruise control: Human factors analysis (No. FHWA-HRT-13-045).

Khan, S. M., Dey, K. C., & Chowdhury, M. (2017). “Real-time traffic state estimation with connected vehicles.” *IEEE Transactions on Intelligent Transportation Systems*, 18(7), 1687-1699.

Ma, X., Tao, Z., Wang, Y., Yu, H., & Wang, Y. (2015). Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, 54, 187-197.

Ma, Y., Chowdhury, M., Sadek, A., & Jeihani, M. (2009). “Real-time highway traffic condition assessment framework using vehicle–infrastructure integration (VII) with artificial intelligence (AI).” *IEEE Transactions on Intelligent Transportation Systems*, 10(4), 615-627.

Ma, Y., Chowdhury, M., Sadek, A., & Jeihani, M. (2012). “Integrated traffic and communication performance evaluation of an intelligent vehicle infrastructure integration (VII) system for online travel-time prediction.” *IEEE Transactions on Intelligent Transportation Systems*, 13(3), 1369-1382.

NSF 2016. *NSF Workshop on Grand Challenges in Edge Computing*. http://iot.eng.wayne.edu/edge/goals.php. Accessed July 26, 2018.

Rabinovich, M., Z. Xiao, & A. Aggarwal, (2004). “Computing on the edge: A platform for replicating internet applications.” *Web content caching and distribution*, 2004, pages 57–77. Springer.

Rayamajhi, A., Rahman, M., Kaur, M., Liu, J., Chowdhury, M., Hu, H., McClendon, J., Wang, K.C., Gosain, A. & Martin, J., (2017). “ThinGs In a Fog: System Illustration with Connected Vehicles.” *In Vehicular Technology Conference (VTC Spring), 2017 IEEE 85th (pp. 1-6). IEEE*.

Ren, H., Song, Y., Liu, J., Hu, Y., & Lei, J. (2017). “A Deep Learning Approach to the Prediction of Short-term Traffic Accident Risk.” *arXiv preprint arXiv:1710.09543*.

Wang, X., & Wang, Y. (2015). “Human-Aware Autonomous Control for Cooperative Adaptive Cruise Control (CACC) Systems.” In *Proceedings of ASME 2015 Dynamic Systems and Control Conference, Volume 2,* p. V002T31A001, American Society of Mechanical Engineers, Washington, D.C., 2015.

Zhao, Z., Chen, W., Wu, X., Chen, P. C., & Liu, J. (2017). “LSTM network: a deep learning approach for short-term traffic forecast.” *IET Intelligent Transport Systems*, 11(2), 68-75.

CHAPTER 2

CV APPLICATION: ADAPTIVE QUEUE PREDICTION

2.1 Introduction

The U.S. Department of Transportation (USDOT) has emphasized the need for new lightweight vehicles to install vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications to support various safety, mobility, and environmental CV applications to address such issues as queues on the roadway. These queues, which can be caused by daily recurring congestion, work zones, incidents and weather conditions, are the major cause of secondary collisions, accounting for approximately 34% of rear-end collisions of the total reported crashes (National Safety Council, 2015; Dimitriou et al. 2018). In addition, they can disrupt regular traffic flow by introducing shockwaves into the upstream traffic. The successful minimization of this secondary collision risk and the resulting traffic flow shockwaves depends on i) the quick detection of a traffic queue including its location, duration, and length, ii) the rapid preparation of a warning message for vehicles approaching the traffic queue, and iii) the broadcast of traffic queue information to approaching vehicles in real-time (Balke et al. 2014).

The traditional queue prediction algorithm used in a CAV environment is based on the average gap between vehicles and the average speed information of the surrounding vehicles in the traffic stream (Balke et al. 2014). Typically, a queued state is determined by the speed of the vehicle as well as the separation distance between the subject vehicle and the one immediately in front of it. If the separation distance is not possible to calculate in a traffic stream, the average speed of CAVs can be used to detect queue states. The queued state along with the Basic Safety Message (BSM) is then transmitted to other CVs every one-tenth of a second using Dedicated Short Range Communication (DRSC).

However, this algorithm detects queues depending on a specific threshold of vehicle speed and separation distance. In addition, it is not applicable in mixed traffic scenarios, one in which non-connected vehicles are between the CVs, making the separation distance between connected vehicles higher than the threshold. In the early days of CVs, data can be collected from only a limited number of CVs (i.e., a low penetration rate) and not from the remaining non-connected vehicles. Moreover, the rate of data loss in the wireless CV environment also contributes to the lack of data from a limited number of CVs (Vivek et al., 2014). Thus, it is challenging to predict traffic behavior (e.g., queue prediction), which changes dynamically over time with this limited CV data.

One potential solution to this issue involves edge computing, a new concept for the TCPS in which the resources for communication, computation, control, and storage are placed at the different edges in the CV system (Rabinovich et al., 2004). This system has the ability to reduce data loss rate and data delivery delay as well as fulfill the high bandwidth requirement. For example, a Roadside Unit (i.e., RSU), located in the next immediate edge layer from the associated CVs, can offer data sharing with low latency to support CV applications. For example, communication bandwidth-hungry sensor data generated from devices, such as video cameras, can be aggregated, analyzed and distributed in the edges within a CV Environment.

In this study, the author develops and evaluates the performance of an adaptive queue prediction application in an edge-centric TCPS. This algorithm was developed using a machine learning approach with a real-time feedback loop, and subsequently evaluated using Simulation of Urban Mobility (SUMO, 2017) and ns3 (Network Simulator 3) (NS3 2017) to determines its efficacy on a roadway network in Clemson, South Carolina, USA. The performance of this edge-centric TCPS was measured in terms of accuracy at varying CV penetration levels and data loss.

The rest of this chapter is structured as follows. Section 2.2 reviews the related work on edge-centric cyber-physical systems and CV applications using machine learning in a CV environment. Section 2.3 then discusses the architecture for a deployment-ready, edge-centric TCPS and the adaptive queue prediction algorithm developed for this edge-centric TCPS. The evaluation of this adaptive queue prediction algorithm is discussed in Section 2.4 along with the analysis of the results. Finally, Section 2.5 concludes with with a concluding discussion.

2.2 Related Work

The past research in this area is reviewed beginning with edge-centric computing for a TCPS and followed by connected vehicle applications that use machine learning strategies.

2.2.1 Edge-centric Computing for a Cyber-Physical System

The term edge-centric computing, first introduced in 2002, was used to refer to applications deployed over Content Delivery Networks (CDN), type of computing motivated by the agreement of various software companies to distribute their software using CDN edge computing servers (Sivasubramanian et al., 2007). These CDN edge servers had the advantages of proximity to and computing resources availability at the edge, advantages that helped achieve high scalability. Initially, edge-centric computing concept was restricted to these CDN servers distributed around the world, with researchers applying this concept to deploy and replicate applications in these networks (Sivasubramanian et al., 2007). However, the edge-centric computing concept can be applied to CV systems because a wealth of devices and roadway sensors, both wired and wireless, are connected with one another and available at the periphery of CV systems.

Fog Computing is a concept similar to edge computing, defined by Cisco (IoT, 2017) as a paradigm that extends Cloud Computing and services to the edge of the network. The primary characteristics of Fog Computing include i) its proximity to the users, ii) its applicability if the geographical distribution is very dense, and iii) its suitability for mobility applications. The primary benefit of edge and Fog Computing is that computing services can be provided at the different edges of a CPS *(*Bonomi et al., 2012*)*, meaning minimizing computation at the cloud and battery exhaustion in mobile terminals while ensuring reliability is its key feature. Edge-centric TCPS is appropriate for CV systems and applications as different heterogeneous edge computing resources are available for real-time data analytics at the different edges of such systems.

2.2.2 Connected Vehicle Applications Development using Machine Learning

The Connected Vehicle Reference Implementation Architecture (CVRIA) provides a framework and a system architecture, which is appropriate for the research and development of CV applications (CVRIA, 2018). CVRIA assumes that a CV platform is an integration of vehicle sensors, GPS, automotive systems, safety devices, and information and entertainment systems. The On-Board Unit (OBU) of each CV provides processing, storage, and V2V and V2I communication functions. It interacts with Roadside Equipment (RSU) and other vehicles (referred to as Remote Vehicle OBUs) using BSMs and communication protocols over a dedicated wireless network operating at a licensed spectrum at 5.9 GHz. To date, CVRIA has defined more than ninety-five CV applications, which include safety, mobility and environmental applications (ARC-IT, 2018).

For traffic operational analysis, different types of algorithms, such as embedded algorithms for loop detector systems, computer vision based algorithms and machine learning based algorithms, have been used for addressing various traffic-related problems (Hernandez et al., 2017; Coifman and Kim, 2013; Ma et al., 2012; Mossi et al., 2011; Rong et al., 2013). With the latter being of interest to the work reported here, such algorithms provide for more accurate traffic operational analysis than statistical methods as they can learn from previous experience with similar roadway conditions as seen in, for example, the case-based reasoning (CBR) machine learning method *(*Aamodt and Plaza, 1994*)*. The primary assumption for predicting traffic behavior is that new behavior will have a solution closest to that of the most similar previous experience. The CBR method has been used in such transportation-related applications as crash analysis, the detouring of traffic during crash events and traffic signal plan calculation (Andersen, 2012; Chowdhury et al., 2006; Schutter et al., 2003; Li et al., 2005).

Support Vector Machine (SVM) and Support Vector Regression (SVR) are also widely used supervised machine learning techniques. SVMs, which are primarily used for classification, outlier detection and regression problem (SVM, 2017), have been used to predict travel time, speed, and traffic flow conditions, and to detect traffic incidents (Chowdhury et al., 2006; SVM, 2017; Ma et al., 2009; Vanajakshi and Rilett, 2004). In addition, Artificial Neural Networks (ANN) have been used extensively in studying traffic data forecasting, driver behavior modeling and vehicle classification (Chowdhury et al., 2006; Ma et al., 2009; Comert and Cetin, 2009). However, there is no existing literature investigating the use of an adaptive queue prediction algorithm in a CV environment in an edge-centric TCPS using machine learning.

2.3 Adaptive Queue Prediction Algorithm in an Edge-centric TCPS

This section describes the details of an edge-centric TCPS system for connected vehicle applications and the adaptive queue prediction algorithm in the edge-centric TCPS.

2.3.1 Edge-Centric Cyber-Physical System

The edge-centric TCPS consists of three edge levels, i) system edge, ii) fixed edge, and iii) mobile edge layers as shown in Figure 2.1(a), forming a hierarchical cyber-physical system architecture that can address the complexity and scale issues of CV systems. The system edge, which pertains to cloud-based services, involves a single end-point for a cluster of fixed edges. A fixed edge, which includes a general-purpose processor (i.e., edge device) and a DSRC based roadside unit (RSU), can communicate with the mobile edge using DSRC and with the system edge using Optical Fiber/WiFi. The fixed edge can also be extended to support a video camera and other sensing devices, such as weather sensors.

CVs participating in a system act as mobile edges and are equipped with an On-Board Unit (OBU). The fixed edges are backhauled to a system edge that can effectively serve as a backend resource, and the OBU can exchange data with the RSU through DSRC communication. Using the DSRC, a mobile edge layer can perform V2I communication. Figure 2.1(b) shows network connectivity among different nodes of the edge-centric TCPS.

Real-world deployments covering large geographical areas require more than one network technology to support communication needs, including, for example, DSRC, WiFi, Long Term Evolution (LTE) and optical fiber. In this system, the choice of communication depends not only on the location of mobile edges from the fixed edge and system edge placement but also on the requirements necessitated by the applications and system services. In this edge-centric system, since the fixed edge layer can support more than one networking technology, it is designed to be equipped with DSRC Radios, Wi-Fi hotspots, Cellular 3GPP, LTE networks or fiber networks. It is also designed to support a wide range of applications and devices such as weather monitoring sensors and video cameras. A simple publish/subscribe messaging protocol is employed at each node to support data transfer among intermittently connected ad-hoc mobile edges, these messaging services enabling applications to send and receive messages via different networks. Messages are transferred as a topic and routed through these various networks, reaching a specific edge layer based on the subscription of topics.

The advantages of the edge-centric TCPS include proximity, intelligence, scalability and control (Premsankar et al., 2018). The proximity to the edge provides the flexibility to communicate efficiently with the immediate edge layers, meaning information distribution is quicker than with a centralized system. For example, the mobile edge layer can send information to the fixed edge layer instead of sending information to the system edge directly. With the increasing number of connected vehicles, roadway sensors and devices will be becoming more powerful, and these large amounts of raw data need to be processed into a usable format based on different CV applications requirements. Thus, intelligence functions need to be distributed between edge layers to reduce the time needed for computationally expensive applications. This edge-centric TCPS can be easily scalable by increasing the number of edges in the different layers to meet the growing demand of computational resources. In addition, it is necessary to manage and control the different edge layers that can assign or delegate computation, synchronization or storage to other edge layers selectively. Furthermore, the location of personal and social media data can be in the edge, and control of data privacy and security issues can reside in an edge depending on the suitability.

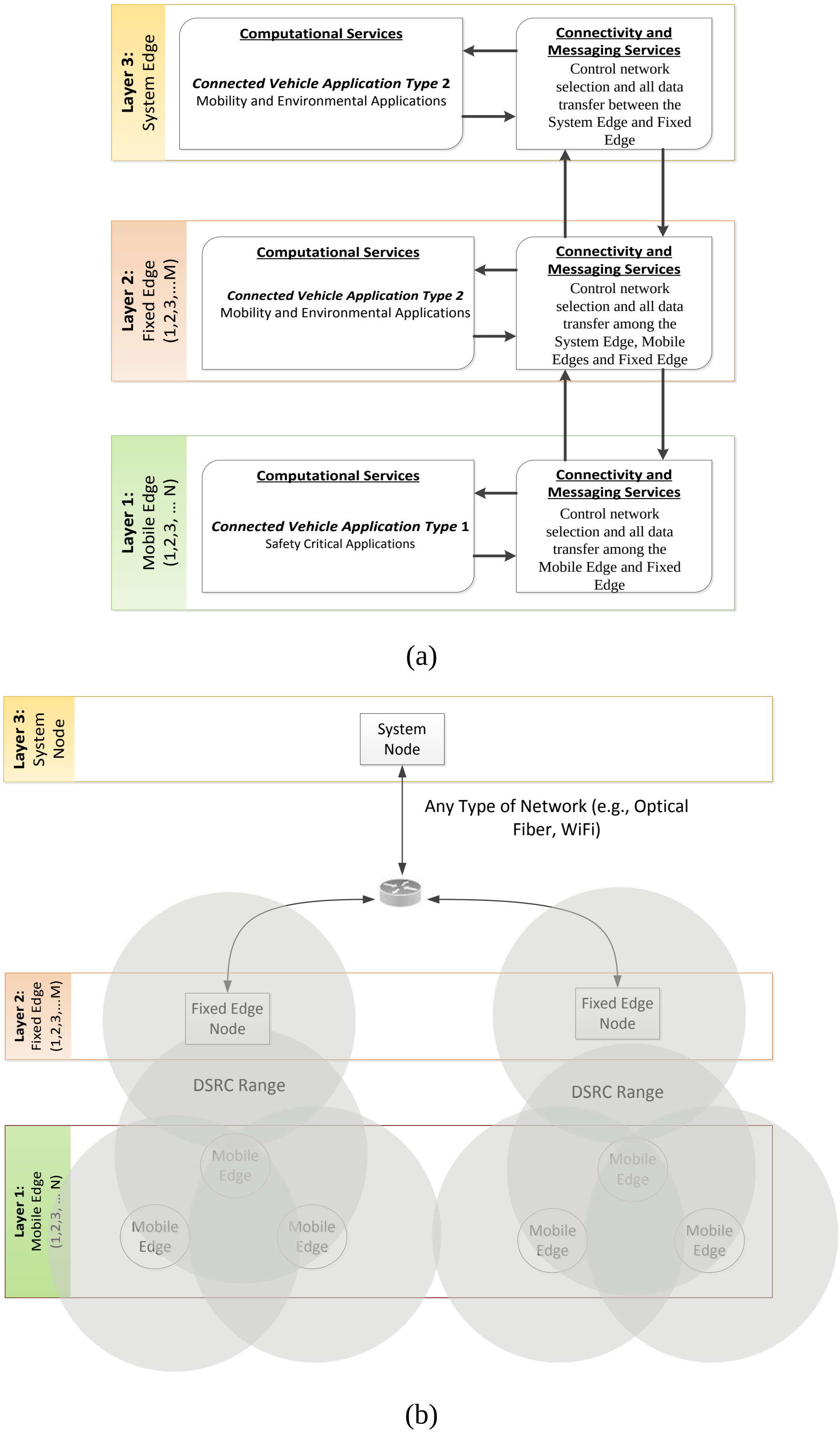

Layer 3: System Edge
Computational Services
Connected Vehicle Application Type 2
Mobility and Environmental Applications
Connectivity and Messaging Services
Control network selection and all data transfer between the System Edge and Fixed Edge
Layer 2: Fixed Edge (1,2,3,...M)
Computational Services
Connected Vehicle Application Type 2
Mobility and Environmental Applications
Connectivity and Messaging Services
Control network selection and all data transfer among the System Edge, Mobile Edges and Fixed Edge
Layer 1: Mobile Edge (1,2,3, ... N)
Computational Services
Connected Vehicle Application Type 1
Safety Critical Applications
Connectivity and Messaging Services
Control network selection and all data transfer among the Mobile Edge and Fixed Edge
(a)
Layer 3: System Node
System Node
Any Type of Network (e.g., Optical Fiber, WiFi)
Layer 2: Fixed Edge (1,2,3,...M)
Fixed Edge Node
Fixed Edge Node
DSRC Range
DSRC Range
Layer 1: Mobile Edge (1,2,3, ... N)
Mobile Edge
Mobile Edge
Mobile Edge
Mobile Edge
Mobile Edge
Mobile Edge
(b)

Fig. 2.1. Physical view of edge-centric cyber-physical system for connected vehicle applications (a) Edge-centric TCPS framework for CV applications; and (b) Communication networks in the edge-centric cyber-physical system in a CV environment

2.3.2 Adaptive Queue Prediction Algorithm using Machine Learning

In this research, an adaptive queue prediction algorithm is developed with a feedback loop for determining whether a CV is traveling in a queued state (as shown in Figure 2.2). This algorithm is deployed in different edge infrastructures, with the machine learning based queue prediction algorithm being deployed in the fixed edge layer, while the parameter estimation process for the machine learning method is implemented in the system edge to reduce the computational load at the fixed edges.

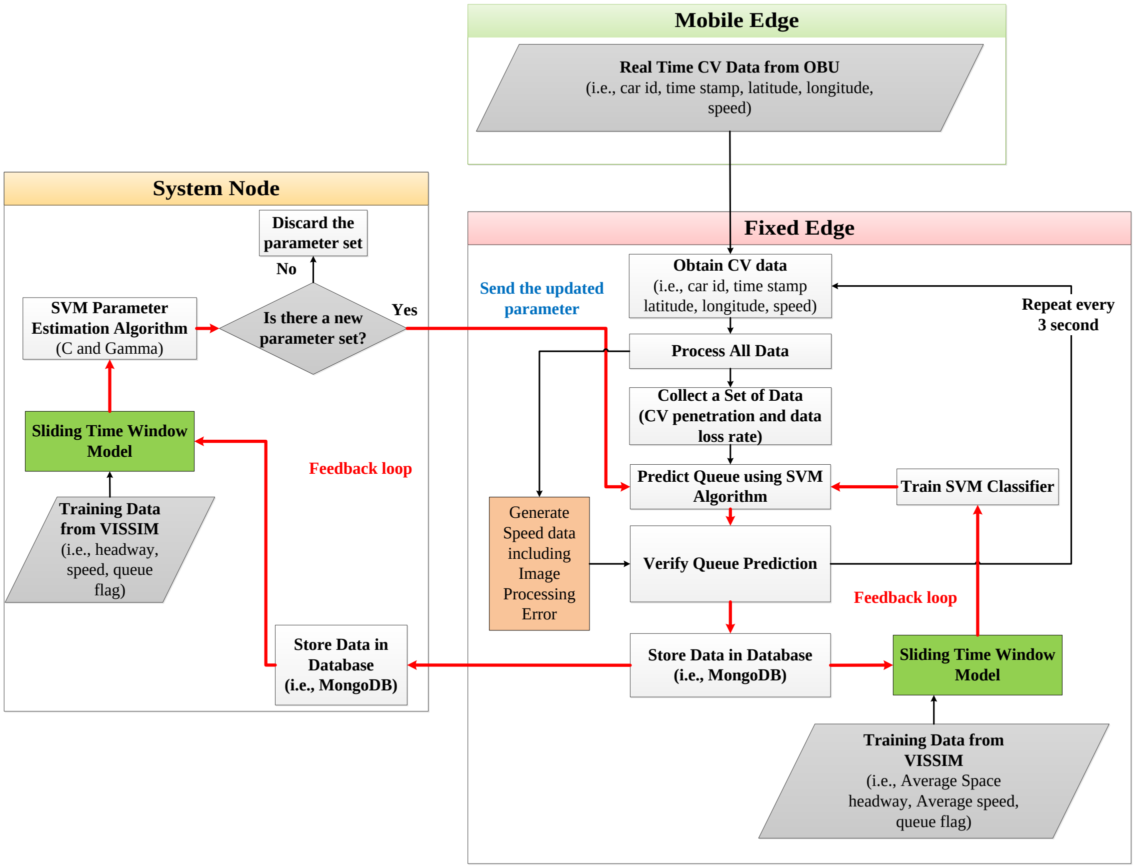

Mobile Edge
Real Time CV Data from OBU
(i.e., car id, time stamp, latitude, longitude, speed)
System Node
Discard the parameter set
No
Yes
SVM Parameter Estimation Algorithm
(C and Gamma)
Is there a new parameter set?
Sliding Time Window Model
Feedback loop
Training Data from VISSIM
(i.e., headway, speed, queue flag)
Store Data in Database
(i.e., MongoDB)
Fixed Edge
Send the updated parameter
Obtain CV data
(i.e., car id, time stamp latitude, longitude, speed)
Repeat every 3 second
Process All Data
Collect a Set of Data
(CV penetration and data loss rate)
Predict Queue using SVM Algorithm
Train SVM Classifier
Generate Speed data including Image Processing Error
Verify Queue Prediction
Feedback loop
Store Data in Database
(i.e., MongoDB)
Sliding Time Window Model
Training Data from VISSIM
(i.e., Average Space headway, Average speed, queue flag)

Fig. 2.2. Adaptive queue prediction algorithm in the edge-centric TCPS.

Each mobile edge node provides BSMs (i.e., the time stamp, latitude, longitude and speed of the vehicle) to the fixed edge every one-tenth of a second. After being received, these basic connected vehicle data are processed and aggregated in the fixed edge, and then the aggregated data are normalized for queue prediction. For training the machine-learning model, labeled simulation data can be used initially. At the fixed edge, the predicted queue status is verified with the average speed on the road network. If the average speed of all vehicles is less than 5 mph (FHWA, 2017), the traffic state is detected as a queue state, and these predicted data are stored in a database. As the queue prediction becomes verified labeled data, it is added to the training dataset for the next prediction time window, and the SVM parameters are updated at the system edge node as shown in Figure 2.2. As the traffic state changes over time, the evolving data stream is added to the training data set for implementing a feedback loop in the system, meaning the queue prediction algorithm adapts with changes in the traffic situations. This edge-centric queue prediction algorithm minimizes the computing workload of processing, aggregating and predicting queues using data from each CV at a mobile edge node or a fixed edge node. The supervised machine learning method (i.e., Support Vector Machine), the preparation of the training and testing data from raw data, the data normalization, the feedback loop development using fixed and dynamic sliding time windows, and the queue prediction accuracy after testing with real-time data are discussed more fully on the following sections.

2.3.2.1 *Support Vector Machine*

In this study, a Support Vector Machine (SVM) was used for predicting queue status. A radial basis kernel function, which was used for the binary classification problem (queue or no queue), was implemented using the Scikit learn library tool in Python (SVM, 2017). While linear kernel, polynomial kernel, and sigmoid kernel are also types of SVM kernel functions, the performance of a radial basis kernel in classification problems is better than other available kernel functions (Ma et al., 2012; Chowdhury et al., 2006). The two coefficients for a radial basis kernel for classification using SVM are C and γ. These two parameters were estimated using the training data set by applying a grid search method to find their optimal values (Hsu and Lin, 2002). As the SVM parameters were estimated in the system edge, the time needed for estimating the parameter set does not affect the real-time application. As suggested by FHWA (2017), this study used headway and speed from the CV generated data for the queue prediction.

2.3.2.2 *Raw Data Processing and Data Normalization*

This study simulated Perimeter Road in Clemson, SC, USA, in the microscopic traffic simulator, VISSIM (VISSIM, 2017), recording individual vehicle data to train the SVM model. The VISSIM output file (.fzp), which contains different microscopic traffic data (e.g., speed, headway, queue flag), was analyzed using a Python script. In this dataset, "queue flag" indicating whether there is a queue, is not a numerical output. The microscopic vehicle trajectory data are used to calculate average speed and headway for all vehicles for every three seconds as an aggregate of three seconds is used for queue prediction later. For training the SVM, queue flags were labeled either 1 or 0, with 1indicating a queue and 0 no queue. These data processing steps were performed by Python coding. A grid-search method was used to find the optimal values of C and γ, with the initial range of C varying from -2 to 2 and γ from -4 to 4. As the SVM parameters were estimated in the system edge, the time needed to estimate the parameter set did not affect the real-time application. When the system edge sent the updated parameter to the fixed edge, the queue was predicted using the updated parameter. SVM-related literature has reported that these machine learning algorithms are sensitive to the scale and variance of the input data (Attig and Perner, 2011; Graf et al., 2003). Thus, both training and real-time test data are needed for normalizing using the same 0 to 1 scale applying the following equation:

$$Y' = \frac{Y - \min\left[Y\right]}{\max\left[Y\right] - \min\left[Y\right]}$$

Where Y' is the normalized vlaue of speed or headway and

Y is the observed speed or space headway

2.3.2.3 *Feedback loop Development*

As shown in Figure 2.2, the feedback loop includes the verification of the predicted queue flag and the policy for adding verified predicted data to the training data set. This study assumes that the adaptive queue prediction algorithm will be implemented in real-world deployment by verifying the queue prediction with the average speed using a video image processing system with a video camera. In the simulation environment, this study calculated the average speed for all the data from 100% CVs and added a margin of error based on the calculated average speed resulting from the video processing. By doing so, the error added to the estimated speed can replicate the real-world scenario in this algorithm with simulated data. This can be done by randomly selecting the average speed value from a range of speed data for a specific prediction time using the average speed from the simulation data and the margin of error (IMAGE, 2017).

To update the training data set, this study used a fixed and a dynamic sliding time window. According to the fixed sliding time window, the predicted labeled data were stored for a specified time in the head of the dataset, and the same number of data rows were removed from the tail of the dataset. The focus of this sliding time window technique is to remove old data from and add recent data to the training data set. However, if the same pattern of data is added over time and the unique data pattern removed, the prediction accuracy is reduced due to the unavailability of labeled data with different patterns in the training data set, especially in evolving data streams. Because of this challenge, this study introduced a dynamic sliding window technique (Bifet, 2009) for updating the training dataset. In this technique, recent verified data are added to the training data set, and older data dropped dynamically based on statistical rules (hypothesis test using mean and variance). In the feedback loop for this study, a parameter-free adaptive size sliding window method, which provides theoretical guarantees of the data pattern for the evolving real-time data streams (Bifet, 2009), was implemented.

There are two inputs in this dynamic sliding time window algorithm, the confidence value $\delta \in (0,1)$ and the sequence of real values $x_1, x_2, x_3, ..., x_t$ (the real values here are the headway and speed data from the real-time CV environment). Each x_t is generated at time *t* according to the data distribution D_t, and $\mu_t \quad \sigma_t^2$ are the expected mean and variance values of x_t from distribution D_t with x_t being normalized on a scale of [0, 1] from an interval [a, b] such that $a \leq x_t \leq b$ with probability 1. In this case, μ_t and σ_t^2 are unknown for the distribution D_t for all *t*. This dynamic sliding time window technique keeps the latest speed ($x_{i,speed}$) and headway ($x_{i,headway}$) values for a sliding window *W*. If the length of the window *W* is *n*, then $\hat{\mu}_w$ is the observed average value of all the headway or speed in *W*, and μ_w is the unknown average value of all the headway or speed of μ_t for $t \in W$. All the data generated are indexed with *t*. However, the instantaneous values of μ_t can be significantly different from μ_w or $\hat{\mu}_w$. As μ_w is the expected value of $\hat{\mu}_w$, these two values become closer for larger time windows. In this case, this research added new samples if the newly added data were distinct from the existing training data and the algorithm added data to the system and the older portion of the data was dropped.

The distinction between the two windows depends on the value of the global error, δ, the window length and the values of the data stream. At every step, this algorithm measures the value of $\hat{\mu}_w$, which is an approximate value of μ_w. The difference in the distribution of elements in time window W_0 and W_1 are measured using a statistical test. The elements of the time windows are statistically different if the difference between the average observed values of these windows exceeds a threshold, $\varepsilon_{threshhold}$. The value of $\varepsilon_{threshhold}$ for the two windows (*W_0* and *W_1*) is computed as follows. Let *n_0* and *n_1* be the number of elements of *W_0* and *W_1*, respectively. Let $\hat{\mu}_{w_0}$ and $\hat{\mu}_{w_1}$ be the average values in time windows, *W_0* and *W_1*, and μ_{w_0} and μ_{w_1} the expected average values of time window, *W₀* and *W₁*, respectively. Multiple hypothesis testing can be avoided by using δ' instead of global error, δ. The threshold $\varepsilon_{threshhold}$ is calculated as follows:

$$\varepsilon_{threshhold} = \sqrt{\frac{1}{2m} . ln\frac{4}{\delta'}} \qquad \text{where, } m = \frac{1}{\frac{1}{n_0}+\frac{1}{n_1}} \text{ ,n } = n_0 + n_1 \, and \;\; \delta' = \frac{\delta}{n}$$

2.3.2.4 *Accuracy*

In this research, accuracy indicates that the percentage of correctly predicted queues and no queues over the sum of queue and no queue data from the queue prediction verification with 100% CV data. This queue flag is considered as the “ground truth” value. For this binary classification problem, the classification accuracy is measured using the following formula for each test data set:

$$A = \frac{x}{n} * 100$$

where,

A = Accuracy(%)

x = Number of correctly classified queue events

n = Number of total events

2.4 Evaluation of Adaptive Queue Prediction Algorithm

The adaptive queue prediction algorithm in the edge-centric TCPS was evaluated by simulating the Perimeter Road Corridor in Clemson, South Carolina. Based on simulation experiments, the reliability of the edge-centric TCPS for CV applications was determined using the performance measure, accuracy, both with and without the feedback loop. The adaptive queue prediction algorithm was also compared with the threshold-based queue prediction algorithm.

2.4.1 Experimental Set-up

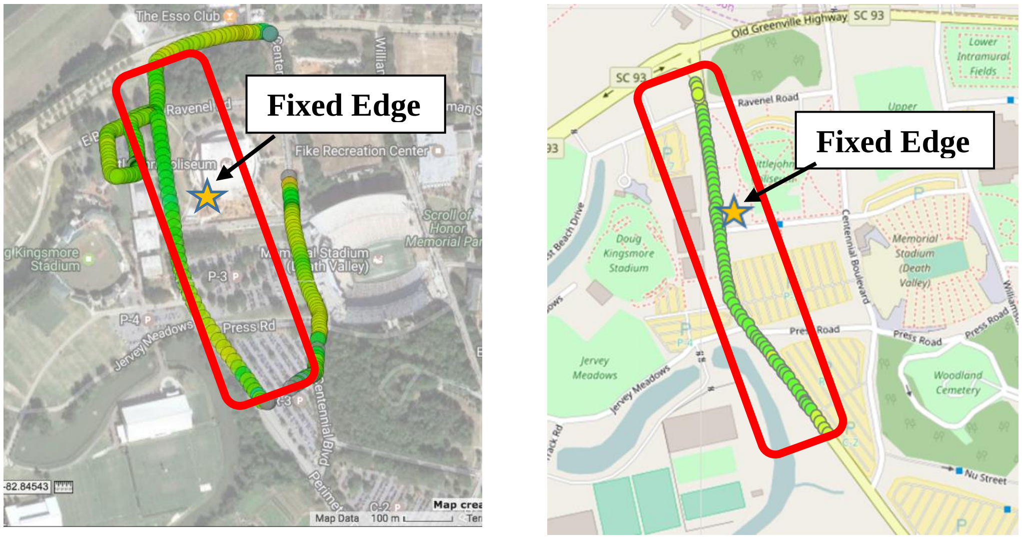
Using the Network Simulator Version 3 (ns-3) (NS3, 2017), the CV environment was developed in the simulation and the effect of different levels of CV penetration on the data loss rate were evaluated with a stationary wireless node acting as an RSU. The mobility of nodes was created using SUMO traffic simulator (SUMO, 2017) to mimic real vehicular movement. An RSU was placed at the Jervy Gym location of Perimeter Road as shown in Figure 2.3. The propagation loss model based on Benin et al. (2012) was added to the ns-3 model to emulate the DSRC range between the RSU and the moving vehicles. An application was developed in the ns-3 simulation to broadcast messages from the vehicles to the RSU using the DSRC network. The simulation tests comprised 200 vehicles per hour per lane on Perimeter Road, a four-lane corridor (two lanes each direction) with a 35 mph speed limit. This study assumed that each vehicle in the simulation was equipped with a DSRC radio only. Fixed edge and system edge were implemented with the Transmission Control Protocol (TCP) to receive information from the mobile and fixed edges. Below are the DSRC simulation parameters in ns3:

- **Propagation loss model:** Three log distance
- **Fading:** Nakagami-m
- **Tx Power Level:** 23 dBm
- **Frequency:** 5.9 GHz

- **Channel BW:** 10 MHz

- **Packet Size:** 250 Bytes (both case)

The DSRC node in each vehicle was used to collect and transfer vehicle data (i.e., time stamp, car ID, latitude, longitude and speed) to a fixed edge node (RSU) for the queue prediction algorithm. The DSRC coverage of the fixed edge through a real-world field experiment is shown in Figure 2.3(a), as this study deployed a fixed edge at the Jervy Gym location and a mobile edge in a conventional vehicle. The DSRC coverage (communication range) of the fixed edge was simulated in ns3 as shown in Figure 2.3(b). As Figure 2.3 shows, the DSRC coverage range in terms of roadway length on the Perimeter Road portion was similar to that in the simulation.



(a) (b)

Fig. 2.3. DSRC range at Jervy Gym location on the perimeter road (a) DSRC Range from Field Experiment; and (a) DSRC range in Simulation

2.4.2 Experimental Scenarios

To evaluate the performance of the adaptive queue prediction algorithm, a baseline scenario of the queue prediction application without a feedback loop was established. This study considered four data loss rates at seven CV penetration levels (as shown in Table 2-I). The range of data loss rate was specified from 2% to 16% based on (Vivek et al., 2014). Then four experimental scenarios with the feedback loop were evaluated, with raw data being collected from each CV (i.e., time stamp, car ID, latitude, longitude and speed) to predict the queue state. The fixed edge used optical fiber to communicate with the backhaul server (system edge), and all mobile edge device-equipped vehicles sent data to the RSU using DSRC. TCP communication was used to connect with the system edge layer when real-time communication was required. After collecting raw data from the CVs (mobile edge), the fixed edge published data to the raw data processing unit at the fixed edge. These processed data were subsequently used for queue state prediction. The database system was hosted using MongoDB to store data at the fixed edge and at the system edge.

Table 2-I Experimental scenarios with and without a feedback loop.

Experimental Scenario	Data Loss Rate (%)	CV Penetration (%)
Scenario 1	2	10, 20, 30, 40, 50, 75, 100
Scenario 2	4	10, 20, 30, 40, 50, 75, 100
Scenario 3	8	10, 20, 30, 40, 50, 75, 100
Scenario 4	16	10, 20, 30, 40, 50, 75, 100

2.4.3 Analysis of Experiments

This section presents the accuracy and statistical significance test results for the adaptive queue prediction algorithm using ns3 simulation. A python script was used in this study to calculate the space headway between two immediate vehicles using the latitude and longitude data of each. The average speed and space headway were also calculated for all CVs every three seconds. Figure 2.4 compares the accuracy of the queue algorithm without the feedback loop with different CV penetration levels and data aggregation times ranging from 1 to 5 seconds. Data loss rate was not considered in these simulation experiments. The results found that the accuracy of the queue prediction algorithm increased with increasing CV penetration level. Moreover, the queue detection accuracy increased with increasing data aggregation time of up to 3 seconds, after which its accuracy decreased. Based on this analysis, a data aggregation time of three seconds was selected for the queue prediction algorithm.

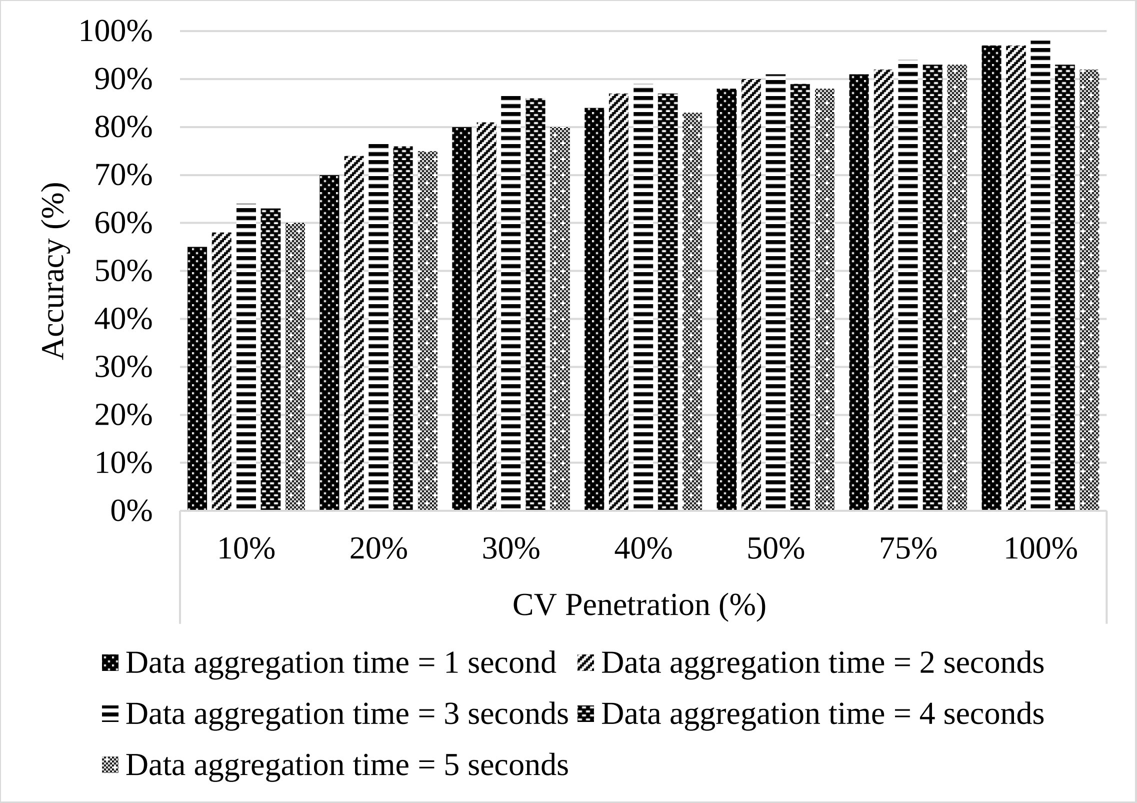
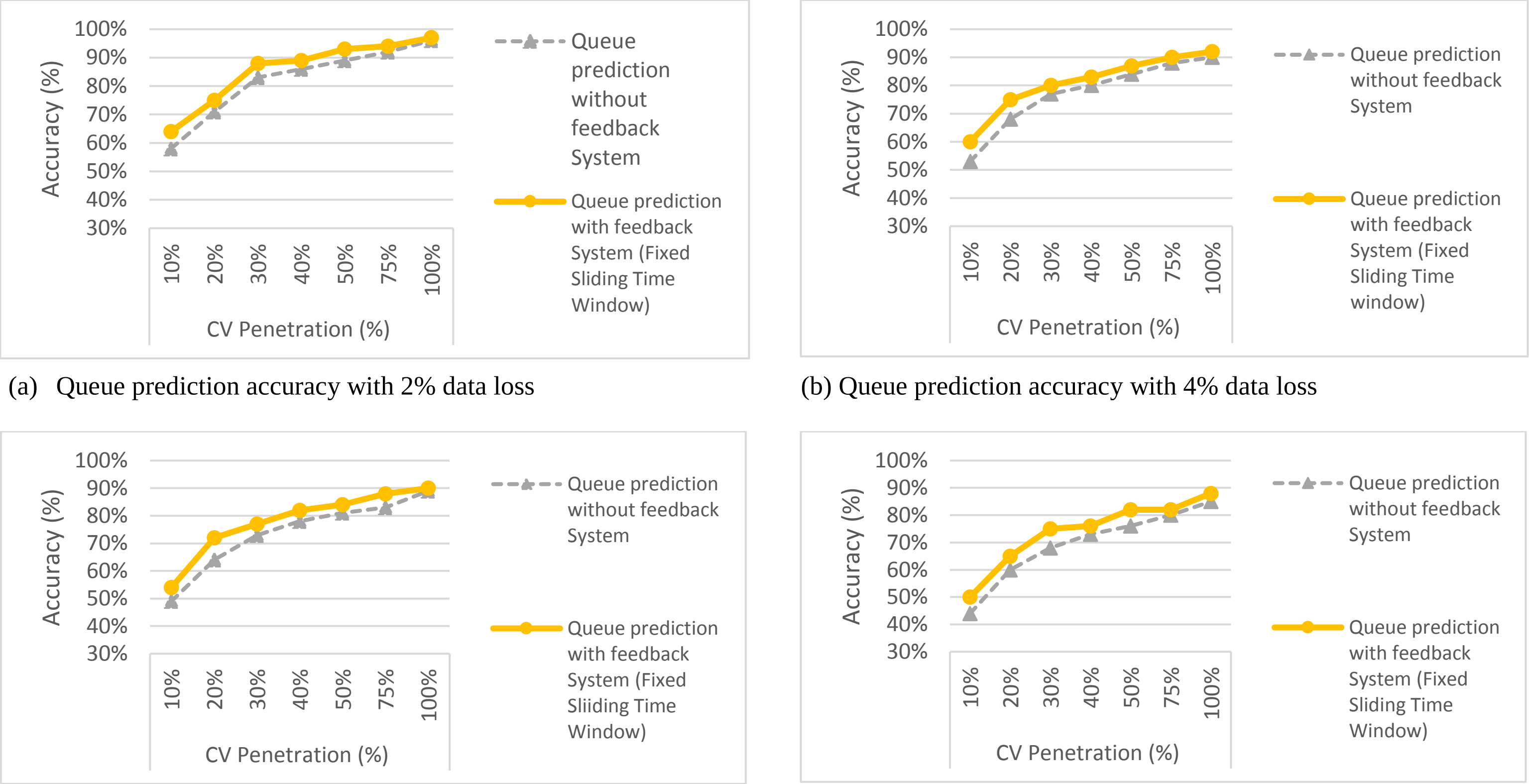


Fig. 2.4. Adaptive queue prediction algorithm accuracy without feedback loop with varying CV penetration levels and data aggregation time.

Figure 2.5 compares the accuracy of the queue prediction algorithm with the feedback loop using a fixed time sliding window of 9 seconds and without the feedback loop varying the CV penetration level and data loss rate. As this figure shows, the accuracy of the queue prediction algorithm increased with increasing CV penetration level for different data loss rates in both cases (i.e., with a feedback loop and without feedback loop). Moreover, the queue prediction was more accurate for low penetration levels than with high CV penetration levels for each data loss rate scenario. Further statistical analysis revealed that the queue detection accuracy with the feedback loop was significantly higher for each CV penetration level at a 95% confidence level than the queue prediction accuracy without the feedback loop.

Figure 2.6 compares the queue prediction algorithm with the feedback loop using a fixed sliding time window and a dynamic sliding time terms window in of algorithm accuracy. One significant finding is that the accuracy of the dynamic sliding time window is higher than for the fixed sliding time window for low penetration levels of CVs. With the high penetration of CVs (50% and above), the performance of the fixed and dynamic sliding scenarios was similar.



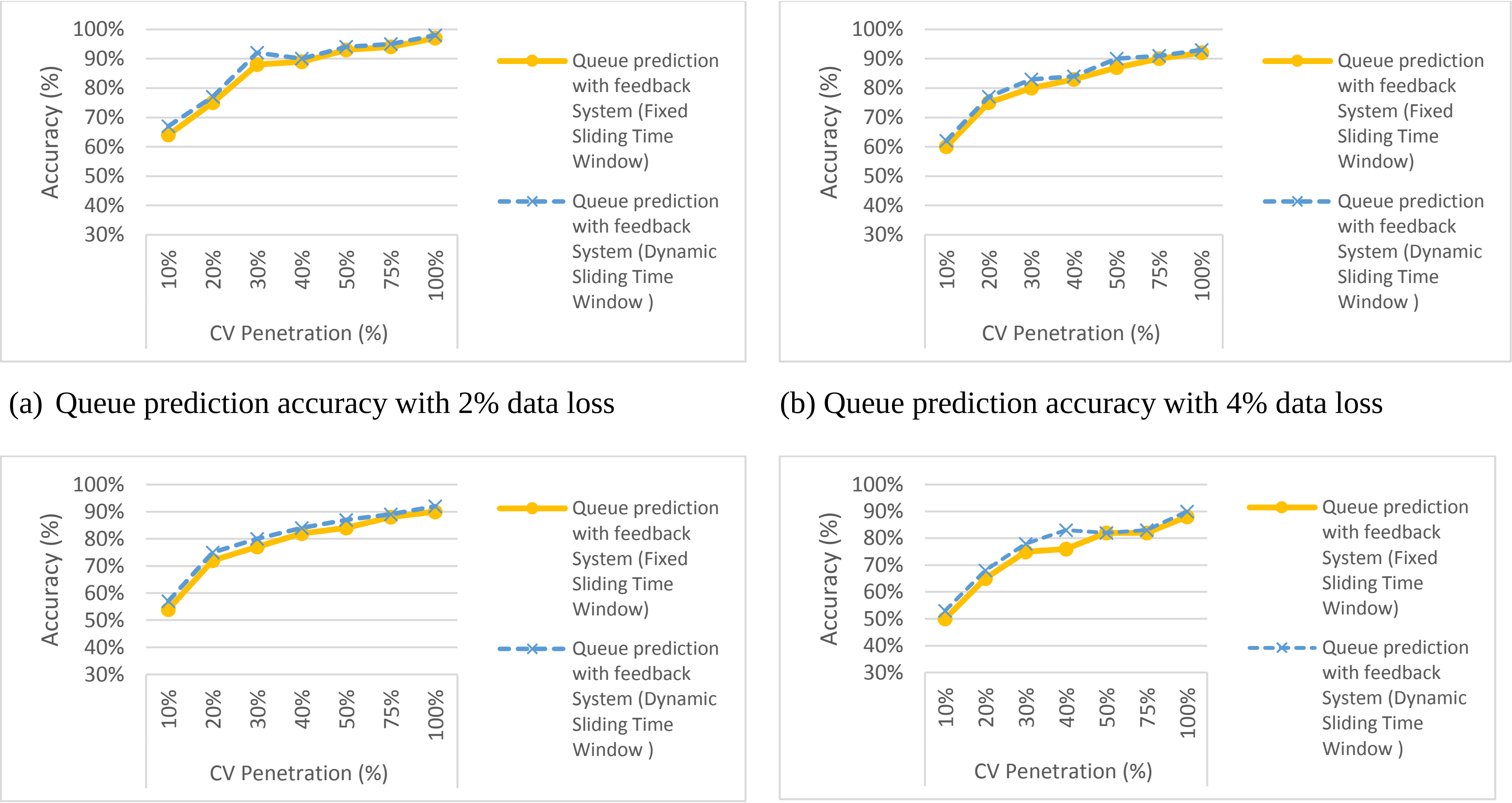
(a) Queue prediction accuracy with 2% data loss

(b) Queue prediction accuracy with 4% data loss

(c) Queue prediction accuracy with 8% data loss

(d) Queue prediction accuracy with 16% data loss

Fig. 2.5. Queue prediction accuracy varying the CV penetration levels and data loss rates with and without the feedback loop.



(a) Queue prediction accuracy with 2% data loss

(b) Queue prediction accuracy with 4% data loss

(c) Queue prediction accuracy with 8% data loss

(d) Queue prediction accuracy with 16% data loss

Fig. 2.6. Comparison of queue prediction accuracy varying the CV penetration levels and data loss rates between Fixed Sliding Time Window and Dynamic Sliding Time Window for the feedback loop.

Figure 2.7 compares the adaptive queue prediction algorithms and the threshold-based algorithm for 100% penetration of CVs with different data loss rates. In terms of accuracy, the adaptive queue prediction algorithms performed better than the threshold-based algorithm. The accuracy difference between the two was larger with higher data loss rates, suggesting for higher data loss, the speed and headway cannot be measured accurately. The threshold-based algorithm can detect a queue only if the average speed is less than 5 mph and the gap between two vehicles is less than 20 ft. Thus, the performance of the threshold-based algorithm decreases because of a predetermined threshold. This algorithm performs well only at 100% penetration of CVs with a 2% data loss.

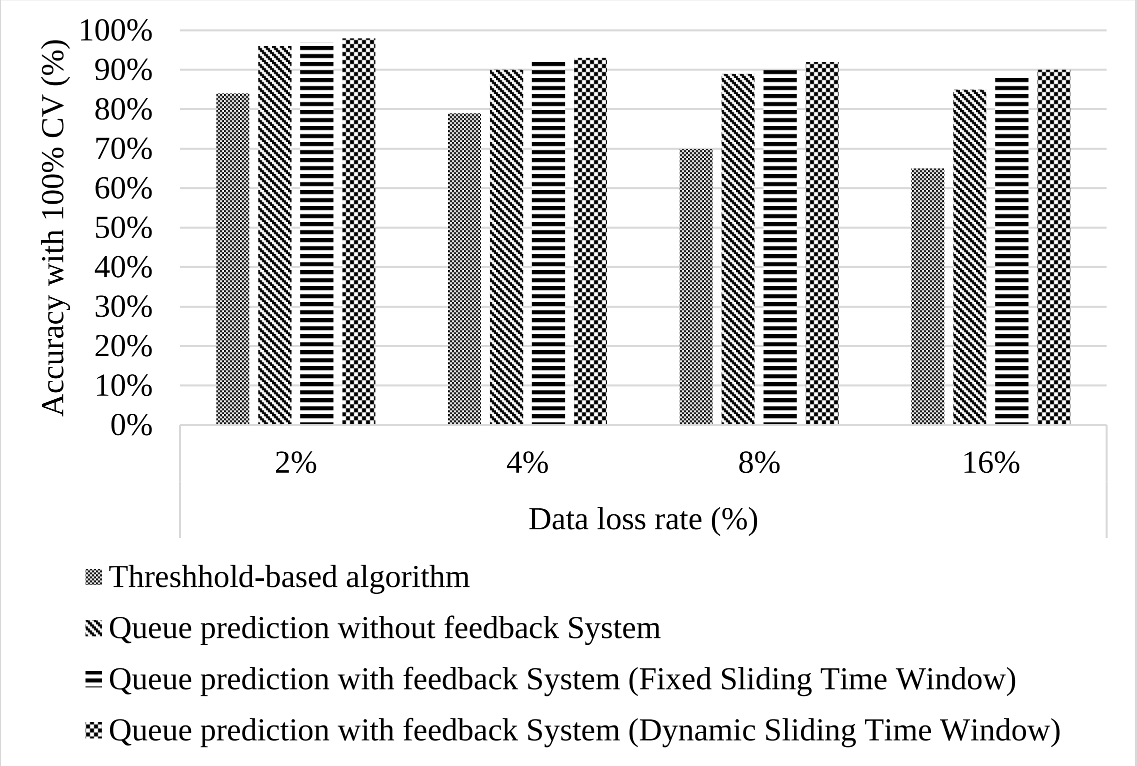


Fig. 2.7. Comparison of queue prediction accuracy varying the data loss rates between Fixed Sliding Time Window and Dynamic Sliding Time Window for the feedback loop.

2.5 Summary and Conclusions

For the initial CV environment, data will only be able to be collected from a limited number of CVs in mixed traffic, a situation compounded by the fact that the data loss rate in a wireless CV environment contributes to the unavailability of data from a limited number of CVs. In this study, a feedback system was developed using SVM to improve the queue prediction accuracy in dynamic traffic conditions. It evaluated the accuracy of the queue prediction algorithm at different CV penetration levels and data loss rates in an integrated SUMO and ns-3 simulation. The simulation analyses revealed that the queue prediction using the feedback system (i.e., adaptive queue prediction algorithm) exhibits a higher accuracy than the queue prediction without a feedback system. The feedback system investigated here was developed using both a fixed and dynamic sliding time window. The unique feature of the dynamic time window is that verified queue prediction data sets with unique patterns are added from the evolving data streams in real-time depending on dynamic traffic conditions. Our analyses demonstrated that the dynamic sliding time window exhibited a higher accuracy than the fixed sliding time window for CV penetration levels of under 50% and that the performance of both the fixed and dynamic sliding scenarios was similar with CV penetration levels of 50% and above. Statistical significance tests indicated no significant difference between the traffic queue prediction using an adaptive queue prediction algorithm and ground truth data.

2.6 Acknowledgements

This material is based upon work supported by the Connected Multimodal Mobility University Transportation Center (C^2M^2) at Clemson University, Clemson, South Carolina, USA, and the National Science Foundation (NSF) under US Ignite Grant #1531127. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the C^{2}M^2 University Transportation Center or the NSF. The author thanks Aniqa Chowdhury for editorial work on the draft.

2.7 References

Attig, A., & P. Perner, P. (2011). “The Problem of Normalization and a Normalized Similarity Measure by Online Data,” *Tran. CBR*, 4(1), 2011, pp 3-17.

Aamodt, A., & E. Plaza, E. (1994) “Case-based reasoning: Foundational issues, methodological variations, and system approaches,” *AI communications*, 7(1), pp 39-59.

Andersen, O. J. (2012). “How Case-based Reasoning can be used to predict and improve Traffic Flow in Urban Intersections,” *Thesis for Master of Science in Informatics, Norwegian University of Science and Technology*. http://www.diva-portal.org/smash/get/diva2:621991/FULLTEXT01.pdf. Accessed July 28, 2017.

Balke, K., Charara, H. & Sunkari, S. (2014). *Report on dynamic speed harmonization and queue warning algorithm design* (No. FHWA-JPO-14-168). United States. Dept. of Transportation. ITS Joint Program Office.

Bifet, A (2009). “Adaptive learning and mining for data streams and frequent patterns,” *ACM SIGKDD Explorations Newsletter*, *11*(1), 55-56.

Benin, J., M. Nowatkowski, and H. Owen (2012). “Vehicular network simulation propagation loss model parameter standardization in ns-3 and beyond,” *Southeast conference, 2012 Proceedings of IEEE*, 1–5.

Bonomi, F., R. Milito, J. Zhu, & S. Addepalli (2012). “Fog computing and its role in the internet of things,” *Proceding of the 2nd Workshop on Mobile Cloud Computing (MCC)*, pages 13–16. ACM.

CVRIA. Connected Vehicle Reference Implementation Architecture (CVRIA). http://www.iteris.com/cvria/html/applications/applications.html. Accessed July 28, 2017.

Coifman, B., & S. Kim (2013). “Assessing the Performance of the Speed Info Sensor,” (No. FHWA/OH-2013/21), *Ohio Department of Transportation*. https://ntl.bts.gov/lib/51000/51200/51209/134723_FR.pdf. Accessed July 28, 2017.

Chowdhury, M., A. Sadek, Y. Ma, Y., N. Kanhere, & P. Bhavsar (2006). “Applications of artificial intelligence paradigms to decision support in real-time traffic management,” *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1968(1), pp 92-98.

Comert, G. & M. Cetin (2009). “Queue length estimation from probe vehicle location and the impacts of sample size,” *European Journal of Operational Research*, 197(1), pp. 196-202.

C.W. Hsu & C.J. Lin (2002). “A simple decomposition method for support vector machine,” (1–3), 2002, pp. 219-314

De Schutter, B., S. P. Hoogendoorn, H. Schuurman, H & S. Stramigioli (2003). “A multi-agent case-based traffic control scenario evaluation system,” *Proceedings of the IEEE Intelligent Transportation Systems*, Vol. 1, pp. 678-683.

Dimitriou, L., Stylianou, K. & Abdel-Aty, M.A., (2018). “Assessing rear-end crash potential in urban locations based on vehicle-by-vehicle interactions, geometric characteristics and operational conditions,” *Accident Analysis & Prevention*.

Graf, A. B., Smola, A. J., & Borer, S. (2003). “Classification in a normalized feature space using support vector machines.” *Neural Networks, IEEE Transactions on*, 14(3), 597-605.

Hernandez, S., A. Tok, & S. G. Ritchie, UCI - ITS - WP - 13 – 4. Density Estimation using Inductive Loop Signature based Vehicle Re-identification and Classification. http://www.its.uci.edu/its/publications/papers/ITS/UCI-ITS-WP-13-4.pdf. Accessed July 28, 2017.

IMAGE (2017). Vehicle speed determination from video streams using image processing. http://ieeexplore.ieee.org/document/7860375/citations. Accessed July 28, 2017.

IoT (2017). from Cloud to Fog Computing. http://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing. Accessed July 28, 2017.

Li, K., & N. M. Waters, N. M (2005). “Transportation networks, case-based reasoning and traffic collision analysis: A methodology for the 21st century,” *Methods and Models in Transport and Telecommunications Springer Berlin Heidelberg,* pp. 63-92.

Ma, Y., Chowdhury, M., Sadek, A., & Jeihani, M. (2009). “Real-time highway traffic condition assessment framework using vehicle–infrastructure integration (VII) with artificial intelligence (AI).” *IEEE Transactions on Intelligent Transportation Systems*, 10(4), 615-627.

Ma, Y., Chowdhury, M., Sadek, A., & Jeihani, M. (2012). “Integrated traffic and communication performance evaluation of an intelligent vehicle infrastructure integration (VII) system for online travel-time prediction.” *IEEE Transactions on Intelligent Transportation Systems,* 13(3), 1369-1382.

Mossi, J. M., A. Albiol, & V. N. Ornedo, V. N. (2011). “Real-time traffic analysis at night-time,” *18th IEEE International Conference on Image Processing (ICIP)*, 2011, pp. 2941-2944.

ns3. Network Simulator (ns3). https://www.nsnam.org/. Accessed July 28, 2017.

National Safety Council. Injury Facts®, 2015 Edition. Itasca, IL: Author Library of Congress Catalog (2015) Card Number: 99-74142

Premsankar, G., Di Francesco, M. & Taleb, T., (2018). “Edge computing for the Internet of Things: a case study,” *IEEE Internet of Things Journal*, 5(2), pp.1275-1284.

Rabinovich, M., Z. Xiao, & A. Aggarwal (2004). “Computing on the edge: A platform for replicating internet applications,” *Web content caching and distribution*, pages 57–77. Springer.

Rong, Y. U., W. A. N. G., Guoxiang, J. Zheng, & W. A. N. G. Haiyan (2013). “Urban Road Traffic Condition Pattern Recognition Based on Support Vector Machine,” *Journal of Transportation Systems Engineering and Information Technology*, 13(1), 2013 pp 130-136.

SUMO. Simulation of Urban Mobility (SUMO). http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/. Accessed July 28, 2017.

SVM. Support vector machines. http://scikit-learn.org/stable/modules/svm.html. Accessed July 28, 2017.

Sivasubramanian, S., G. Pierre, M. V. Steen, & G. Alonso (2007). “Analysis of caching and replication strategies for web applications,” *IEEE Internet Computing* 11, 2007, no. 1.

Vanajakshi, L. & L. R. Rilett (2004). “A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed,” *IEEE Intelligent Vehicles Symposium,* 2004, pp. 194-199.

VISSIM. 5.40 User Manual, PTV Vision, Karlsruhe, Germany, 2011.

Vivek, N., S.V. Srikanth, P. Saurabh, T. P. Vamsi, & K. Raju (2014). “On field performance analysis of IEEE 802.11 p and WAVE protocol stack for V2V & V2I communication,” *2014 International Conference on Information Communication and Embedded Systems (ICICES), 2014,* pp. 1-6, IEEE.

CHAPTER 3

CV APPLICATION: TRAFFIC DATA PREDICTION

3.1 Introduction

With the advancement of new technology in the field of Intelligent Transportation Systems (ITS), transportation agencies have implemented such initiatives as Advanced Traffic Management Systems (ATMS) and Advanced Traveler Information Systems (ATIS) to inform travelers about current and future traffic conditions (Vanajakshi and Rilett, 2004; Ma et al., 2012; Khan et al., 2017). These traffic management strategies, which depend on accurate prediction of traffic data including travel speed and space headway between vehicles, are used for route planning and scheduling to reduce travel time, for future traffic condition assessment, and for energy optimization to reduce fuel consumption (Ma et al., 2015; Hua et al., 2017; Ren et al. 2017; Ma et al., 2009; Ma et al., 2012; Khan et al., 2017).

This real-time prediction is challenging because the traffic flow of a roadway stochastically changes over time, for example depending on the time of day or day of the week (Ma et al., 2015). As a result, it is important to capture the temporal relationship over time to predict traffic data accurately. Currently, these data have been collected through a wide range of such roadway traffic sensors as inductive loop detectors and video cameras. The problem is that such data cannot simultaneously capture the stochastic nature of the traffic flow in terms of spatial and temporal variation for a specific segment of a roadway as these sensors are deployed at a fixed location (Ma et al., 2015; Zhao et al., 2017).

These issues are currently being addressed by connected vehicle (CV) technologies that provide interconnection between transportation systems allowing vehicles to share Basic Safety Messages (BSMs) by communicating with one another as well as with transportation roadside infrastructures (e.g., traffic signal, roadside unit) and Traffic Management Centers (TMC). In this CV system, BSMs provide trajectory data, such as the location, speed, acceleration and deceleration, of each vehicle for every one-tenth of a second (Liu and Khattak, 2016; Du et al., 2017). This more accurate temporal data also then imply the spatial variation. Providing real-time BSMs with this temporal and spatial variation requires data-driven approaches, such as machine learning models for capturing the non-linearity of traffic patterns.

More specifically, recurrent neural networks (RNNs), a type of machine learning model that can capture temporal variation and predict time series data, have been used to predict freeway traffic volume (Zhao et al., 2017). However, previous researchers have confirmed that these traditional RNNs are unable to capture the long temporal dependency for an input sequence because of vanishing gradient or exploding gradient problems (Zhao et al., 2017; Ma et al., 2015; Hochreiter and Schmidhuber, 1997). To address these limitations, a special RNN architecture, the Long Short-Term Memory Neural Network (LSTM) (Hochreiter and Schmidhuber, 1997) has been developed for time series prediction.

Over the past decade, the LSTM has been successfully used in the following areas: (i) robot control; (ii) speed recognition; (iii) handwriting recognition; and (iv) human action recognition. More recently, it has been used to predict univariate traffic data, such as volume and speed, using data collected from roadway sensors (Zhao et at., 2017; Ma et al., 2015). However, there is limited, if any, research using LSTM and BSMs from connected vehicles to predict traffic data. Because the penetration of connected vehicles in the near future will be limited, traffic data such as average speed and space headway between vehicles will be noisy, thus reducing the prediction accuracy of this model. This resulting noise in the traffic data because of a less than 100% CV penetration is analogous to the inaccurate data collection resulting from a sensor fault.

To improve the prediction accuracy using these noisy data, this research first investigated noise reduction models, the standard Kalman filter and Kalman filter based Rauch–Tung–Striebel (RTS) data smoothing techniques, to reduce the noise from the traffic data measured from BSMs. Second, the performance of the LSTM prediction model was evaluated for predicting traffic data using the resulting filtered data. Using a vehicle penetration rate ranging from 5% to 90%, Enhanced Next Generation Simulation (NGSIM) data (Montanino and Punzo, 2013), which contain vehicle trajectory data for every one-tenth of a second, was used as the BSMs for the evaluation of the LSTM prediction model.

The remainder of this research is structured as follows. Section 3.2 describes the related work on traffic data prediction using machine learning and noise reduction models, while Section 3.3 discusses the analysis of traffic data from BSMs of CVs in a mixed traffic scenario (connected and non-connected vehicles) to explore the noise resulting from the speed and space headway data. Section 3.4 analyzes a method for predicting traffic data using different noise reduction models at low penetration of CVs, with the evaluation of resulting method being discussed in Section 3.5 along with the analysis results, and Section 3.6 provides a concluding discussion.

3.2 Related Work

The related work analyzes existing research on different types of recurrent neural network models for traffic prediction and noise reduction models.

3.2.1 Recurrent Neural Network Models for Traffic Prediction

Feed Forward Neural Networks (FFNN), the simplest neural networks, have been used extensively to study forecasting travel time and traffic flow as well as subsequent traffic patterns (Park and Rilett, 1999; Chowdhury et al., 2006; Ma et al., 2009). However, they are unable to capture temporal and spatial variations in time series problems as they do not have a memory mechanism that allows for the evolution of traffic patterns for traffic flow predictions in the future. In addition, spatial and temporal patterns and optimal look back intervals must be determined beforehand for input into FFNN for the time series prediction, a step requiring data preprocessing using statistical methods (e.g., correlation analyses, principal component analysis, and genetic algorithm) to prepare a large enough time series dataset to capture spatial and temporal patterns that is not efficient for real-time time series predictions.

More recently, RNNs have been explored for capturing variations over time for a time series problem including, for example Time-Delay Neural Network (TDNN), Jordan–Elman Neural Network, State-Space Neural Network (SSNN)). The first two models have been applied to traffic speed predictions using 30-s loop-detector speed data from a freeway segment of Interstate 4 in Orlando, Florida, the results showing that these models outperformed non-linear statistical time series model (Ishak et al., 2003). The SSNN model has also been used for real-time short-term freeway travel time prediction using synthetic and real-world data (Van Lint et al., 2002, 2005; Liu et al., 2006), one example being the real-time data collected from freeway and urban scenarios for the Regiolab-Delft project (Muller et al. 2005; Van Zuylen and Muller 2002). However, these RNN models are unable to capture temporal and spatial relationship for a long-term time series problem because of vanishing gradient and exploding gradient problems.

To address these issues for long-term time series problems, the LSTM model has been explored for traffic data prediction. For example, Ma et al. (2015) used a three-hidden-layer LSTM model for traffic speed prediction utilizing microwave sensor data. The hidden layer of this model includes a memory block for capturing the non-linear patterns of speed over the time. This research found that the LSTM provided more accurate predictions than traditional RNN models by determining optimal time lags using a trial and error method. More recently, Zhao et al. (Zhao et al., 2017) constructed a multi-layers LSTM network for traffic volume prediction. Their model includes an Origin-Destination Correlation (ODC) matrix integrated in the LSTM network. This ODC matrix captures the correlations between the temporal and spatial patterns among different links of a road network, thus improving the performance of the LSTM model by capturing traffic flow evolution over time and space. This study found that the two-dimensional LSTM was more accurate than existing traffic forecast methods for short-term travel speed prediction. In further study, Wang et al. (2017) developed a deep neural network using an LSTM for predicting driver behavior. As driver behavior is a time-dependent phenomenon and an LSTM can mimic human memory, they developed an LSTM based car-following model that can replicate driver behavior using microscopic NGSIM data. This research found that this deep neural network model exhibits significantly higher accuracy than existing car-following models.

However, past research has not explored using an LSTM to predict multivariate traffic data using BSMs in a connected vehicle environment at a low penetration of CVs. As the related work indicates that this type of RNN has the capability of capturing long-term dependency for predicting time series data, this study used an LSTM model for predicting traffic data in a connected vehicle environment. In this environment, the LSTM model can learn non-linear time-variant traffic behavior from a training data set and predict traffic data based on the real-time input of traffic data. However, this learning capability of the LSTM model can be reduced by the noise in the data from a mixed traffic environment (i.e., connected and non-connected vehicles), as one cannot expect 100% CVs in the near future.

3.2.2 Noise Reduction Models

Noise reduction models have been used extensively to analyze given measurements and to estimate accurate measurements because of inaccuracies of sensor collected data. In the past, vehicle trajectories data were filtered using the following methods: (i) averaging (Ossen and Hoogendoorn 2008); (ii) locally weighted regression using the tri-cube weight function (Toledo et al. 2007); (iii) filtering (Punzo et al. 2005; Montanino and Punzo 2013) and (iv) moving average techniques (Thiemann et al. 2008). The noise reduction accuracy of these methods depends on window size. Kanagaraj et al. (2015) and Rim et al. (2016) used locally weighted regression techniques for smoothing erroneous vehicle coordinates and speed data, respectively, both finding that the accuracy of locally weighted regression varies based on the polynomial order. More recently, Punzo et al. (2005, 2011) used moving average and low pass filtering techniques to correct GPS-based trajectory data, with the latter study analyzing the vehicle trajectory and speed data and evaluating the accuracy in terms of jerk, consistency, and spectral analysis. They found that the low pass filter performs very well in terms of accuracy.

Another widely used data smoothing technique is the Kalman Filter, which is used to reduce noise from sensor fault in sensor-collected data. This filter, named after Rudolf E. Kálmán, who provided the concept for this method (Kalman, 1960), estimates the current state based on a sequence of previous noisy observations. There are three types of Kalman filter smoothing, fixed-interval smoothing, fixed-point smoothing, and fixed-lag smoothing in addition to several variations including the standard Kalman filter, the extended Kalman filter and the scented Kalman filter (Gadsden and Lee, 2017). If the noise in the sensor-collected data is Gaussian, the standard Kalman filter is applicable for the noise reduction. The standard Kalman Filter has been found effective in estimating air vehicle sensor errors. In the past, Ervin et al. (1991) used a Kalman filter to smooth the vehicle trajectory data.

Using the standard Kalman filter process, Rauch et al. (1965) developed an efficient method based on the RTS algorithm, a two-pass algorithm that reduces the computational effort required for Kalman filter smoothing since it requires the standard Kalman filter to be implemented only in the forward direction. The forward pass is the standard Kalman filter while the backward recursion is introduced to reduce the inherent bias in the Kalman filter estimates. Based on their applicability, the RTS and standard Kalman filter have been implemented as noise reduction models. In a mixed traffic scenario (connected and non-connected vehicles), data collected from the low penetration of connected vehicles (e.g., 5 % CVs, 10% CVs) in addition to the temporal variation of traffic make the data noisy.

3.3 Analysis of Traffic Data with Different Penetration of CVs

This section describes the data collected from each vehicle of a 500m (1650 ft) roadway section on Interstate 80 segment in Emeryville (San Francisco), California (NGSIM, 2006). Then, data are analyzed for 10 different penetration of CVs to analyze the noise at the low penetration of CVs.

3.3.1 Basic Safety Messages (BSMs) Obtained from the Enhanced NGSIM Dataset

The original NGSIM dataset, which was collected through video cameras, represents 45 minutes of the peak afternoon period, specifically 4:00 PM to 4:15 PM, and 5:00 PM to 5:30 PM) (NGSIM, 2006). The vehicle trajectory data of each vehicle were generated through video image processing. However, since the original NGSIM data contain inconsistencies and noise, Punzo et al. (2011) improved the data set using a multistep procedure to reconstruct the original I 80-1 dataset (4.00 PM to 4.15 PM) for each vehicle trajectory, and subsequently conducted an extensive exploratory study to determine the accuracy of NGSIM trajectory data (Punzo et al., 2011). They reconstructed the original data measurements while preserving: i) the dynamics of the actual vehicle while being driven (i.e., shifting gears, vehicle stoppages); ii) the internal vehicle trajectory consistency (i.e. the vehicle trajectory consistency for acceleration/deceleration, speed, and space travelled); and iii) the consistency in the platoon (i.e. the actual space headway between the follower and leader vehicles in the traffic stream) (Montanino and Punzo, 2013). This reconstructed data set is referred to as the Enhanced NGSIM dataset.

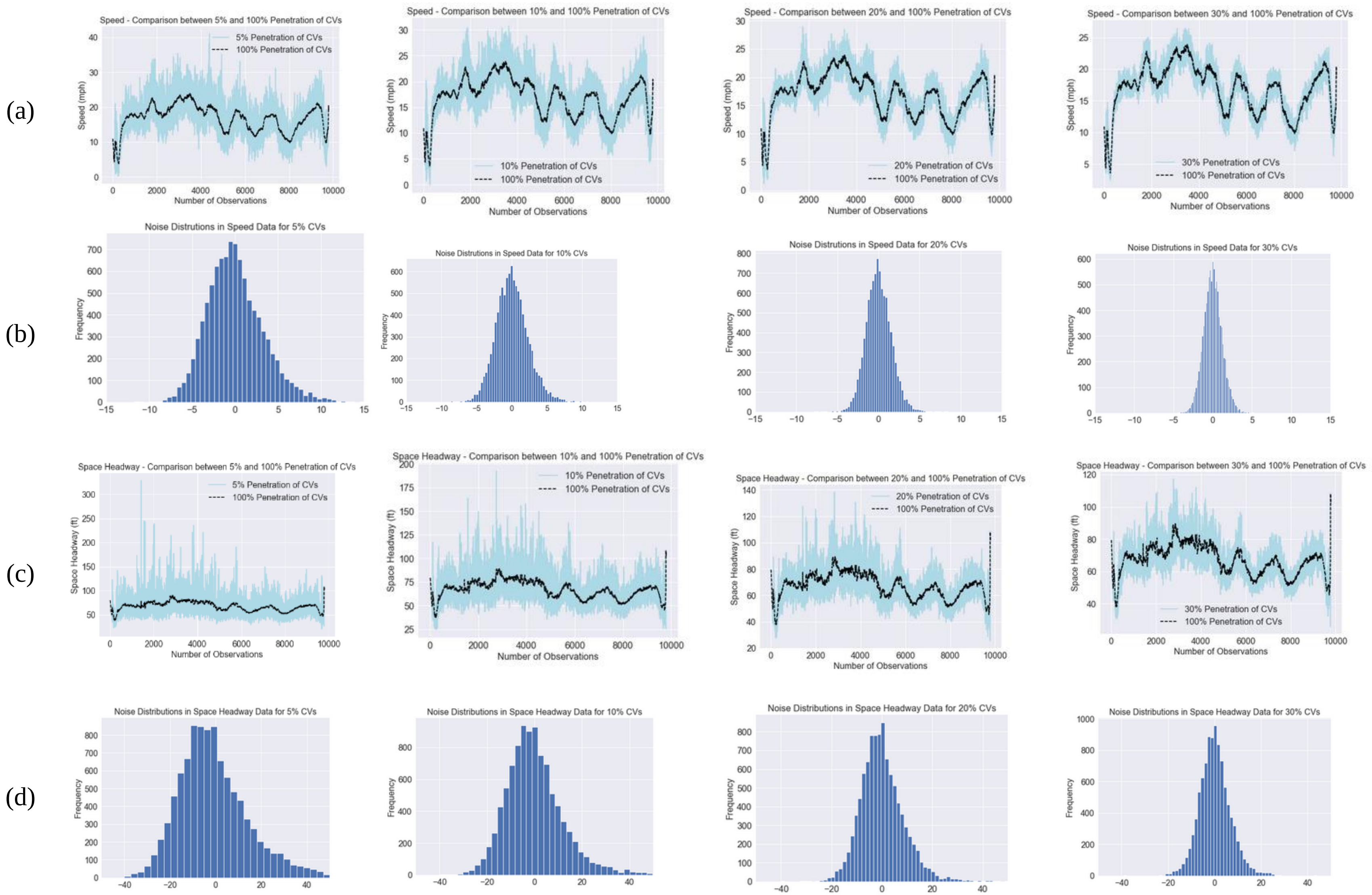
As the Enhanced NGSIM dataset was collected with a frequency of one-tenth of a second, it represents a sample of the BSMs (i.e., Vehicle ID, Timestamp, Lane ID, Location, Acceleration/Deceleration, Vehicle Length, Vehicle Class ID, Follower Vehicle ID, and immediate Preceding Vehicle ID) in a connected vehicle environment. The study reported here used trajectory data from 3,335 vehicles with a frequency of 10 Hz from the Enhanced NGSIM I80-1 dataset. More specifically, this study explored the noise and outliers in the speed and space headway data from 5% to 90% penetration of CVs (5%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, and 90%).

3.3.2 Exploring Noise in the Speed and Space Headway Data in a Mixed Traffic Scenario

Figures 3.1(a) and 3.1(c) compare the four penetrations of CVs, specifically, 100% CVs with 5%, 10%, 20% and 30% penetration, while Figures 3.1(b), and 3.1(d) present the histogram of the noise distributions in the speed and space headway data. To analyze the type of noise distributions, the noise for each observation was calculated by subtracting the speed or space headway data for a specific penetration from the 100% penetration of CVs. This process was followed for all ten penetration rates studied here. As this comparison indicates, both the speed and space headway data become noisy over time. The histogram plots of the noise distributions indicate that all distributions look normally distributed (a Gaussian distribution). In addition, the analysis of the distributions indicates that the noise distributions for other six penetration rates followed the similar distribution.

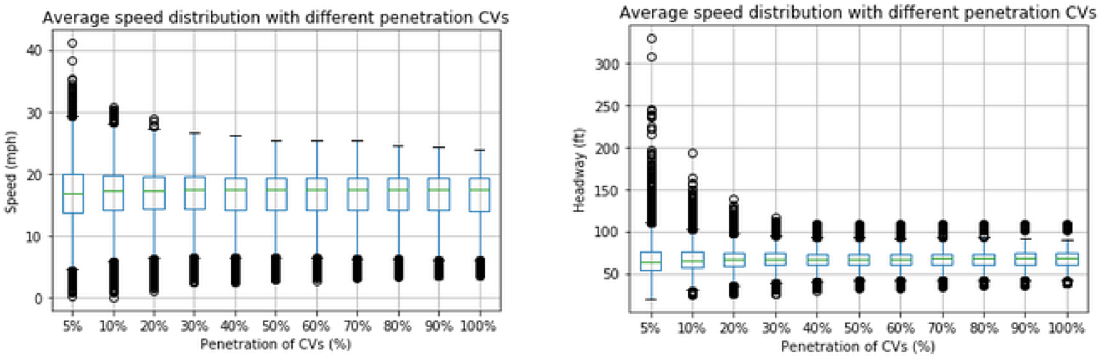
To confirm the normality (Gaussian distribution) of the noise, the data were further analyzed using a quantile-quantile plot (Q-Q plot). This plot generated idealized samples based on the Gaussian or normal distribution from the given noise. The idealized samples were divided into groups, called quantiles. Each data point in the sample was paired with a similar member from the idealized distribution at the same cumulative distribution, and the resulting points were plotted as a scatter plot with the idealized value on the x-axis and the data sample on the y-axis. The resulting plot indicated that the idealized samples followed the normal distribution lines, confirming that the noise distributions for all the penetration rates followed a normal distribution.

This study subsequently analyzed the variation in the outliers by changing the penetration of CVs. Figures 3.2(a) and 3.2(b) show the box plots of the speed and space headway data with varying the penetrations of CVs. As these figures show, the number of outliers increases with decreasing penetration rates of CVs, also indicating that the speed and headway data are changing drastically and, therefore, suggesting that the representation of the variation of these data over time is not accurate.



(a) Comparison between different penetration of CVs and 100 % CVs for speed data; (b) Histogram of noise distributions in speed data; (c) Comparison between different penetration of CVs and 100 % CVs for space headway data; (d) Histogram of noise distributions in space headway data.

Fig. 3.1. Exploring noise in speed and space headway distribution for different penetration of CVs.



(a) Average speed distribution (b) Average space headway distribution

Fig. 3.2. Average speed and space headway distribution varying penetration of CVs.

3.4 Traffic Data Prediction Model Development

The general framework for the traffic data prediction method reported here using LSTM combined with the noise reduction model is presented in Figure 3.3. Average speed and space headway time series data extracted from BSMs at different penetration rate of CVs were used as the input for the noise reduction model. After the filtered data were put into a temporal sequence, training and testing data were prepared. Using the normalized training data, the optimal hyperparameter set of the LSTM model was determined. Then, this set and the testing data were used to evaluate the data prediction method based on the LSTM model.

As enhanced NGSIM data were collected every one-tenth of a second for each vehicle, these data can used as a part of the BSMs in a connected vehicle environment. BSMs in this study include time stamp, location coordinate, speed, acceleration/deceleration, relative speed, lane number, leader vehicle number and follower vehicle number. To prepare the data as a time series, this study used the Frame ID sequence from the NGSIM data as each video frame was created every one-tenth of a second. Then, using the location coordinate of each vehicle, the space headway of a subject vehicle was calculated. In this study, speed and space headway between vehicles were calculated at every one-tenth of a second. The entire dataset contained 9800 samples. Then the average speed and average space headway for the different penetration levels of the connected vehicles were calculated, data that were used as the input of the noise reduction model.

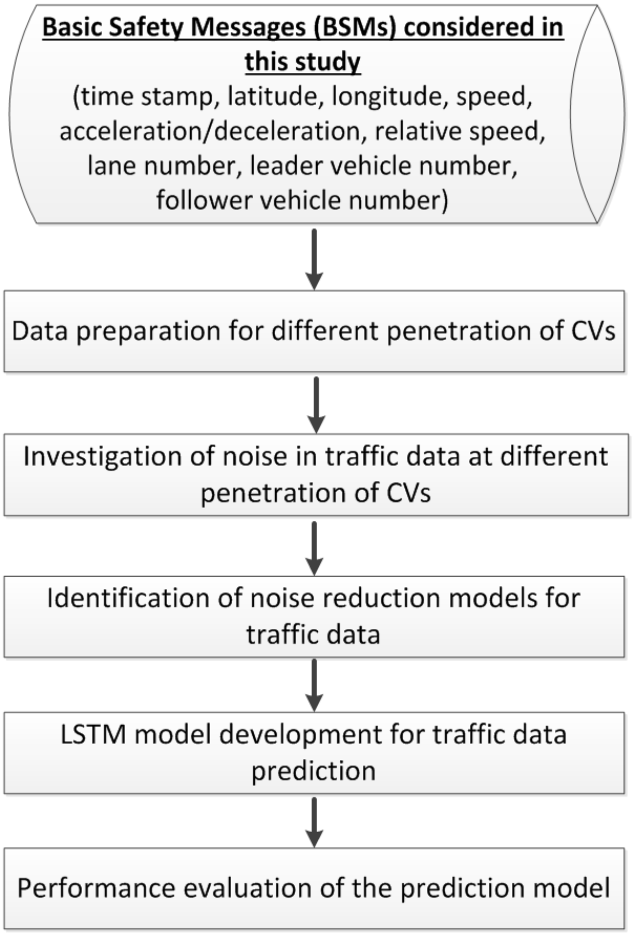

Basic Safety Messages (BSMs) considered in this study
(time stamp, latitude, longitude, speed, acceleration/deceleration, relative speed, lane number, leader vehicle number, follower vehicle number)
Data preparation for different penetration of CVs
Investigation of noise in traffic data at different penetration of CVs
Identification of noise reduction models for traffic data
LSTM model development for traffic data prediction
Performance evaluation of the prediction model

Fig. 3.3. Traffic data prediction method using LSTM combined with a noise reduction model.

3.4.1 Noise Reduction Model

This study used two noise reduction models, the Standard Kalman filter and the Kalman Filter based RTS filter.

3.4.1.1 *Kalman Filter*

According to Kalman filter, the state of speed or space headway at time t evolves from the state at *(t-1)*.

$$x_t = A_t x_{t-1} + B_t u_t + w_t$$

where

A_t is the state transition matrix that transforms state $\boldsymbol{x}_{t-1}$ to state $\boldsymbol{x}_t$

B_t is the control input matrix for measuring the correction for external influences based on control vector u_t;

w_{t-1} is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution $\boldsymbol{N}$ with covariance Q_k: $w_{t-1} \sim N\left(0, Q_{t-1}\right)$

At time t, z_t is a measurement value calculated based on the linear combination of the new estimated state x_t and the measurement noise v_t.

$$z_t = H_t x_t + v_t$$

where

H_t is the measurement matrix that transforms the new estimated state to a measured state

v_t is the measurement noise which is assumed to be zero-mean Gaussian white noise with covariance R_t: $v_t \sim N\left(0, R_t\right)$

The Kalman filter algorithm consists of two stages for reducing the noise of the speed and space headway data: i) prior estimation of the new state and ii) measurement update. Using the following equations, new speed and space headway are estimated at time *t*.

$$\hat{x}_{t|t-1}^{prior} = A_t \hat{x}_{t-1|t-1} + B_t u_t$$

$$P_{t|t-1}^{prior} = A_t P_{t-1|t-1} A_t^T + Q_t$$

Next, the prior estimation of state $\hat{x}_t^{prior}$ and covariance P_t^{prior} are required for updating the measurement at time *t*. Then the current speed or space headway $\hat{x}_t$ can be estimated at time t:

$$\hat{x}_{t|t} = \hat{x}_{t|t-1}^{prior} + K_t \left(z_t - H_t \hat{x}_{t|t-1}^{prior} \right)$$

P_t **is** calculated for updating the value of ***x*** at time *(t-1)* as follows:

$$P_t = \left(I - H_t K_t \right) P_{t|t-1}^{prior}$$

where *K_t* is the Kalman gain. Using the prior covariance P_t^{prior}, the Kalman gain can be calculated as follows:

$$K_t = P_{t|t-1}^{prior} H_t^T \left(H_t P_{t|t-1}^{prior} H_t^T + R_t \right)^{-1}$$

3.4.1.2 *Kalman Filter Based Rauch–Tung–Striebel (RTS) Model*

The Rauch–Tung–Striebel (RTS) smoother uses the same forward pass as the standard Kalman filter algorithm. The resulting prior and posterior state estimates $\hat{x}_{t|t-1}$ and $\hat{x}_{t|t}$, and covariances $P_{t|t-1}$ and $P_{t|t}$ from the forward pass are used in the backward pass, which computes the smoothed state estimates $\hat{x}_{t|n}$ and covariance $P_{t|n}$ (where $t<n$). To do so, the backward steps can be completed using the following recursive equations:

$$\hat{x}_{t|n} = \hat{x}_{t|t} + C_t\left(\hat{x}_{t+1|n} - \hat{x}_{t+1|t}\right)$$
$$P_{t|n} = P_{t|t} + C_t\left(P_{t+1|n} - P_{t+1|t}\right)C_t^T$$

Where,

$$C_t = P_{t|t}F_{t+1}^T P_{t+1|t}^{-1}$$

where $x_{t|t}$ is the a-posteriori state estimate of time step *t*; $x_{t+1|t}$ is the a prior state estimate of time step *t+1*; $P_{t|t}$ is the posterior covariance estimate of time step *t*, and $P_{t+1|t}$ is the a prior state estimate of time step *t+1*.

3.4.2 Preparation of the Training and Testing Dataset

After processing the BSMs from the enhanced NGSIM time series data, the time series data were input into a supervised learning problem. Specifically, the observation of speed or space headway at the current time step was used as an input to forecast the observation at the next time step. The data between 0 and 1 were rescaled to meet the default hyperbolic tangent activation function of the LSTM model. After predicting traffic data using scaled data, the Root Mean Square Error (RMSE) value for a predicted traffic data was calculated based on the scaled data. To develop and evaluate the LSTM model, the dataset was divided into two groups: 1) training dataset and 2) testing dataset.

3.4.3 Long Short-Term Memory (LSTM)

LSTM model used in this research consists of three layers: (i) an input layer, (ii) a recurrent hidden layer, and (iii) an output layer. The model input in the input layer is denoted as $x = (x_1, x_2, x_3, ..., x_t)$, and the output sequence in the output layer is denoted as $h = (h_1, h_2, h_3, ..., h_t)$, where *t* is the prediction period. In the context of speed and space headway prediction, x can be considered as current speed or space headway data, and h is the predicted speed. Of these layers, the primary layer is the recurrent hidden layer, which consists of a memory block, which solves the vanishing gradient or exploding gradient problems of traditional RNNs. The memory block consists of three gates: (a) a forget gate, (b) an input gate, and (c) an output gate. These three gates control what information needs to be removed or added from the previous cell state to the new cell state. The input gate controls the activations of input into the memory block. The input gate i_t decides which values need to be updated using a sigmoid activation function:

$$i_t = sigmoid\left(w_i x_t + u_i h_{t-1} + b_i\right)$$

where

$w, u =$ parameter matrices

$b \quad =$ bias

The forget gate decides what information needs to be forgotten from the previous cell state. Using a sigmoid layer, the forget gate layer f_t, which is represented by the following equation, makes this decision.

$$f_t = sigmoid\left(w_f x_t + u_f h_{t-1} + b_f\right)$$

Based on the input gate and forget gate information, the previous cell state, c_{t-1}, is updated to the new cell state c_t. To obtain the new cell state, the previous cell state c_{t-1} is multiplied by f_t to forget unnecessary information from the previous state. Then, new candidate values $i_t \odot \tanh\left(w_c x_t + u_c h_{t-1} + b_c\right)$ are added to define how much is needed to update each state value:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh\left(w_c x_t + u_c h_{t-1} + b_c\right)$$

The output gate controls the activations of output into the memory block. At the output gate, a sigmoid layer decides what parts of the cell state to output, o_t:

$$o_t = sigmoid\left(w_o x_t + u_o h_{t-1} + b_o\right)$$

Then, cell state c_t is put through *tanh* (to push the values to between −1 and 1) activation functions and multiplied by the output of the sigmoid gate output o_t to predict speed or space headway *h:*

$$h_t = o_t \odot \tanh\left(c_t\right)$$

However, the prediction accuracy of the LSTM model depends on the determination of the optimal hyperparameter that includes the number of neurons, the number of epochs, batch size, dropout rate, and learning rate.

3.4.4 Optimal LSTM Hyperparameter Determination

For the time series problem, traditional hyperparameter selection methods such as the grid search method and the random search method (Bergstra and Bengio, 2012) cannot be applied to determine the optimal hyperparameter set. Thus, this study used a trial and error procedure and the Root Mean Square Error (RMSE) metric to determine the optimal LSTM hyperparameter set. RMSE measures the square root of the average of the squared errors, which quantifies the difference between the predicted values and the actual values. The mathematical formulation of RMSE is as follows:

$$\text{RMSE} = \frac{1}{N}\sqrt{\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2}$$

where

N represents the total sample size

y_i is the actual value traffic data (speed or space headway) and

$\hat{y}_i$ is the predicted value of traffic data (predicted speed or predicted space headway).

This study used a box and whisker plot to identify the optimal parameter set for a specific hyperparameter. Figure 3.4 shows the box and whisker plot for the number of neurons selection process for the LSTM model. Thirty samples for each value of a hyperparameter were created and subsequently plotted in the box and whisker plot to select the optimal hyperparameter set. The plot shows the median (green line), 25th and 75th percentiles of the data. This comparison also indicates that the optimal number of neurons is 100. A similar procedure was followed for the selection of other hyperparameters (i.e., number of the epoch, batch size, dropout rate, learning rate). The optimal hyperparameter values for LSTM model are number of epochs = 400; number of neurons = 100; batch size = 50; dropout rate = 0.2; and learning rate = 0.001.

This research applied a stochastics gradient descent algorithm with adaptive learning rate tricks (Kingma and Ba, 2014) to train the LSTM. This new gradient decent algorithm is an extension of the stochastic gradient descent algorithm. It was used here because of its applicability in computer vision and natural language processing applications.

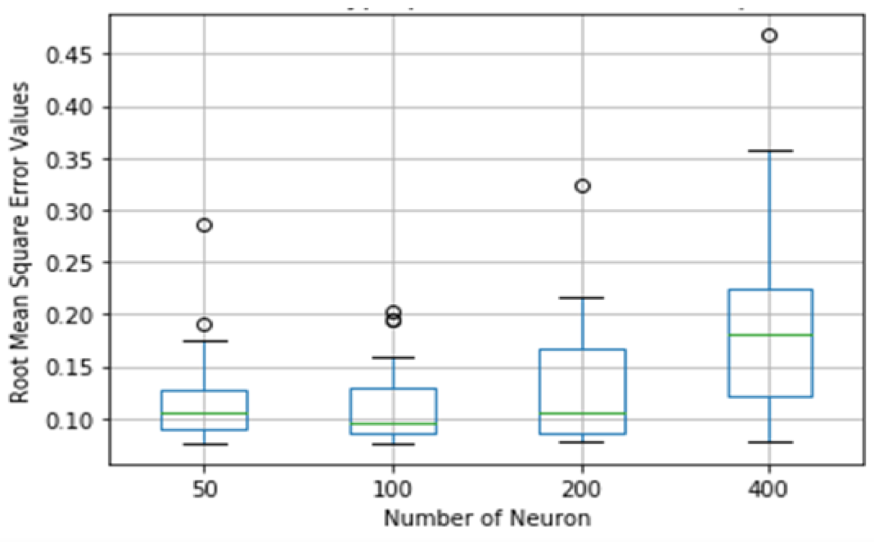


Fig. 3.4. Box and Whisker plot for tuning the number of neuron.

3.4.5 Training and Testing LSTM model

To investigate if the LSTM model has a good fit with the training and testing datasets, the overfitting and underfitting problems must be determined using optimal the hyperparameters first to ensure the prediction capability of the model. A good fit is one for which the performance of the trained model is good on both the training and testing data sets. An underfit model performs well on the training dataset but poorly on the testing dataset, while for an overfit model the performance is good on the training set initially and it continues to improve whereas on the testing set its improves to a point and then begins to degrade. If the loss on training and testing datasets decreases and stabilizes around the same point, then the model has a good fit with the training and testing data. To assess the good fit of the model using training and testing datasets, Mean Absolute Error (MAE), defined as the average of the absolute error, is used as a measurement of the loss and as a metric to evaluate the performance. The mathematical formulation of MAE is given below:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \hat{y}_i\right|$$

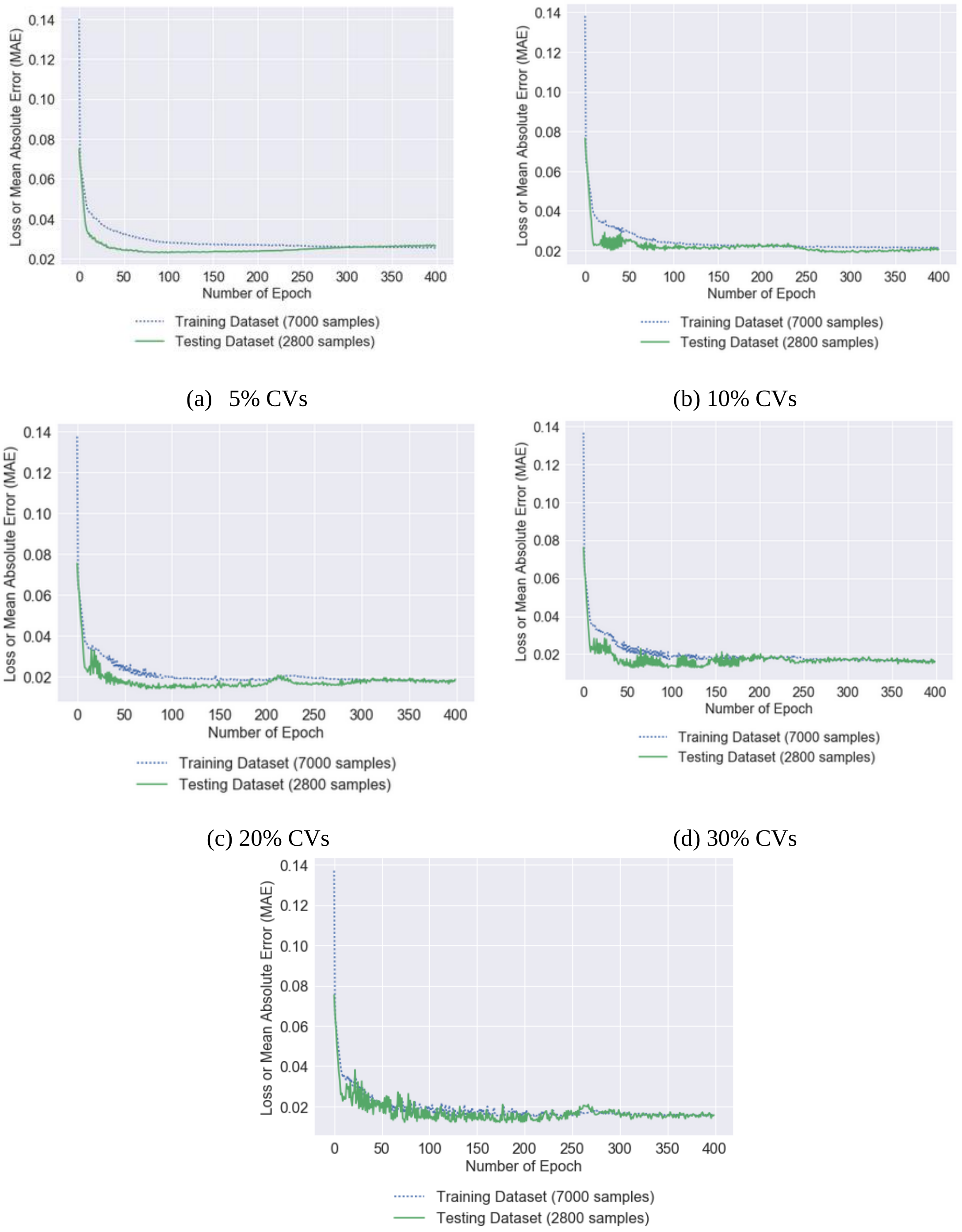
where:

N represents the total sample size

y_i is the actual value traffic data (speed or space headway) and

$\hat{y}_i$ is the predicted value of traffic data (predicted speed or predicted space

To evaluate the good fit of the LSTM model, the author plotted several examples from 5% to 40% penetration of CVs of the MAE profiles using both the training and testing datasets and the optimal hyperparameters as seen in Figure 3.5. Based on the comparison of MAE values of these two datasets, each model exhibits a good fit with the optimal hyperparameter set.



(a) 5% CVs

(b) 10% CVs

(c) 20% CVs

(d) 30% CVs

(e) 40% CVs

Fig. 3.5. Comparison of Mean Absolute Error (MAE) for the training and testing datasets using LSTM with the optimal parameter set.

3.5 Evaluation Results and Discussions

As the evaluation of the traffic data prediction was based on the performance of the LSTM model, this study used the filtered data resulting from the noise reduction models to predict the speed and space headway. The performance of the traffic data prediction using LSTM in conjunction with different noise reduction models was compared with the LSTM baseline model without noise reduction using RMSE, MAE, and MAPE as evaluation metrics.

Figures 3.6, 3.7, and 3.8 compare the speed prediction for the RMSE, MAE and MAPE values, respectively, for the LSTM with the three noise reduction filters, RTS, Kalman Filter and Moving Average and LSTM without a noise reduction filter. All RMSE, MAE and MAPE values were calculated based on normalized ground truth and the predicted value of speed using the optimal hyperparameter set for each of the prediction models as identified in Section 3.4.4. The range of the speed data normalization scale is from zero (0) to one (1). As shown in Figure 3.6, based on the comparison of the RMSE values, the results indicate that the LSTM combined with the three noise reduction models performs better than the LSTM alone. More specifically, the LSTM combined with the RTS filter provided lower RMSE values compared to filtered speed data using the Moving Average and the standard Kalman filters. For example, this value was reduced from 0.101 using LSTM only to 0.051 using the combination of LSTM and RTS filter at a 5% CVs penetration rate. Figure 3.7 shows similar comparison results for the MAE values.

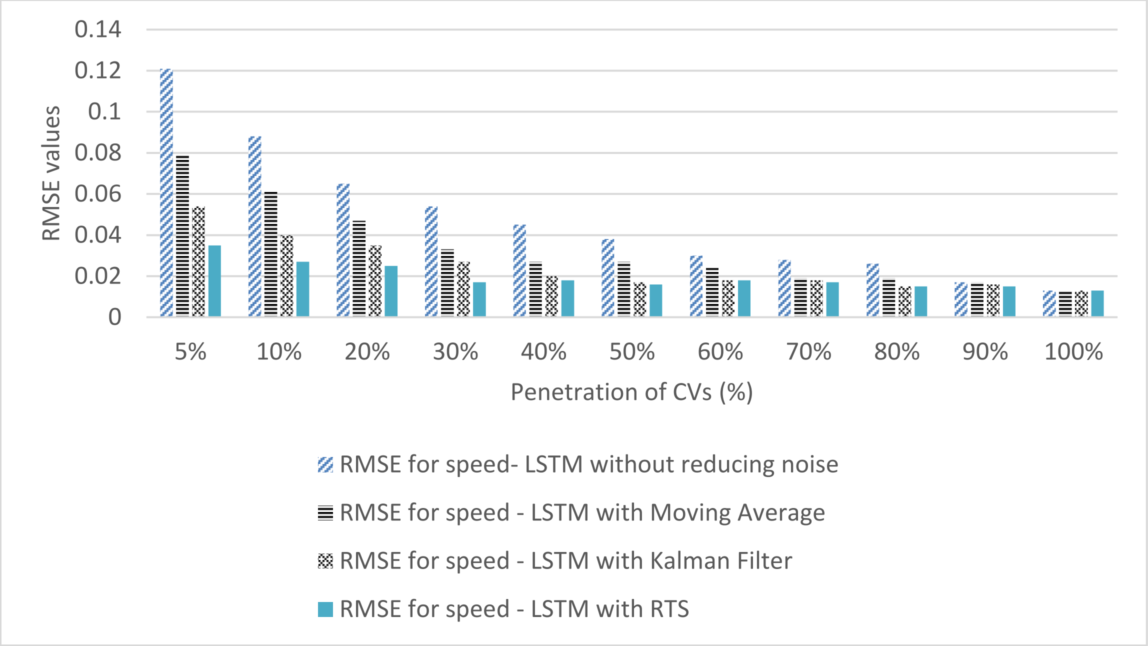


Fig. 3.6. Comparison of RMSE for speed prediction between LSTM with Moving Average, standard Kalman and RTS noise reduction filters, and LSTM without noise a reduction filter.

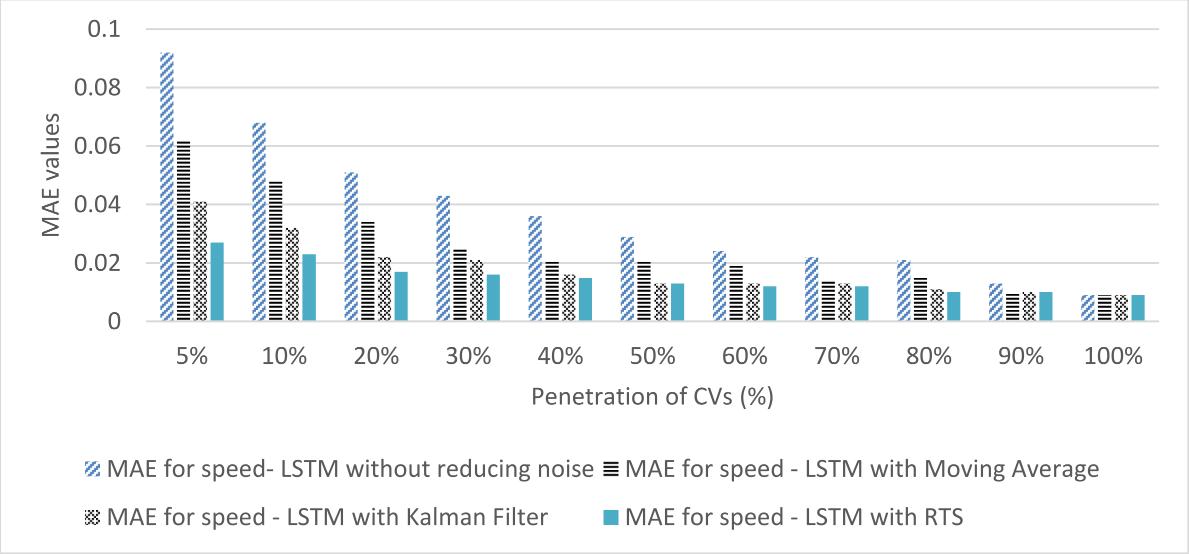


Fig. 3.7. Comparison of MAE for speed prediction between LSTM with Moving Average, standard Kalman and RTS noise reduction filters, and LSTM without noise a reduction filter.

As shown in Figure 3.8, compared to LSTM alone, LSTM combined with the RTS noise reduction model reduced MAPE from 19% to 5% for speed prediction at a 5% penetration of CVs. This research found that LSTM combined with the RTS noise reduction model reduced MAPE ranges from 1% to 14% for speed for different penetration of CVs.

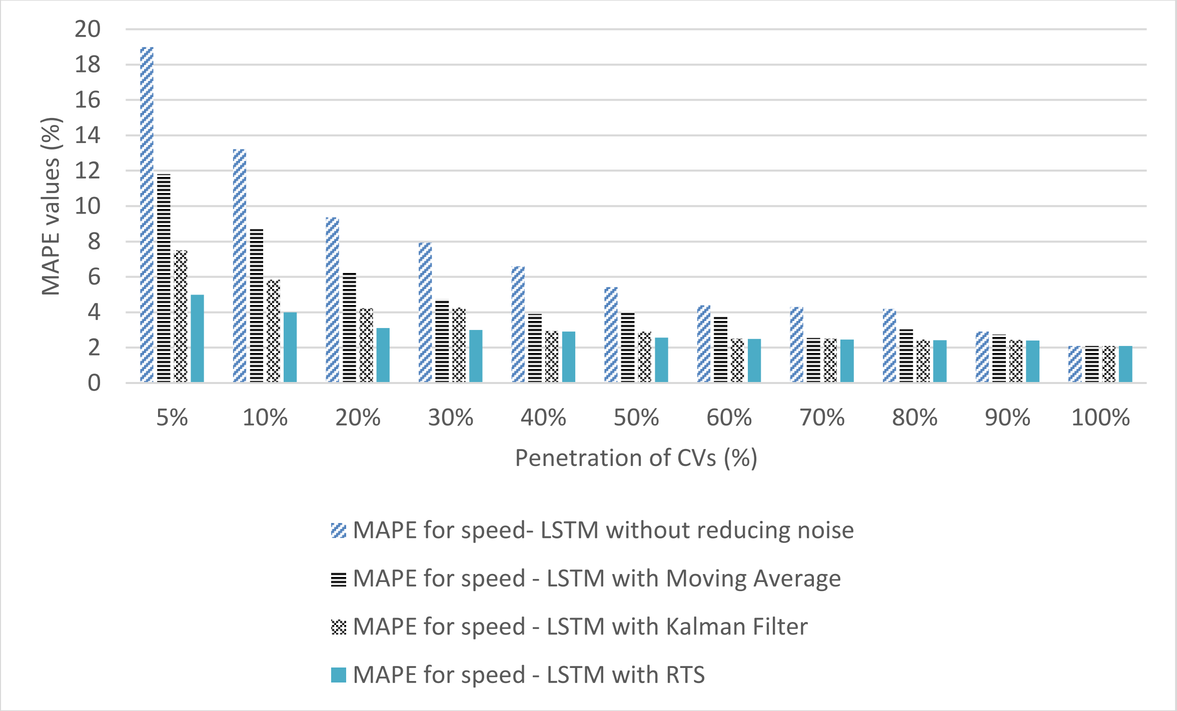


Fig. 3.8. Comparison of MAPE for speed prediction between LSTM with Moving Average, standard Kalman and RTS noise reduction filters, and LSTM without noise a reduction filter.

Figures 3.9, 3.10 and 3.11 compare the RMSE, MAE and MAPE values for space headway prediction, respectively, for the LSTM with the three noise reduction models explored here and the LSTM without a noise reduction filter. Similar to the speed prediction, the results indicate that LSTM combined with all three noise reduction models performs better than LSTM without a noise reduction model. Furthermore, LSTM combined with RTS filter provided lower MAE values similar to the RMSE values.

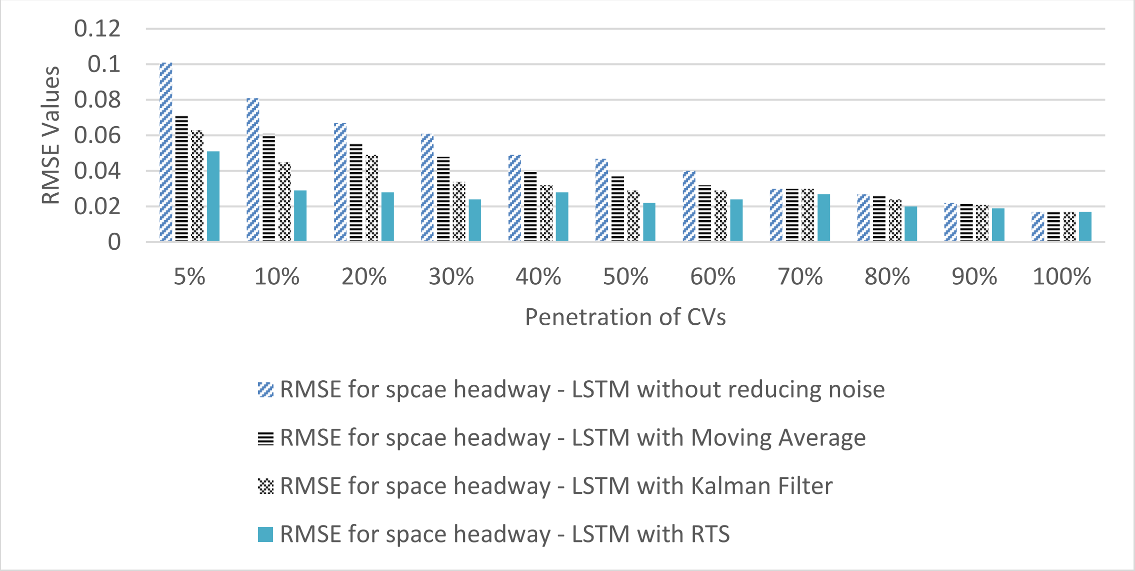


Fig. 3.9. Comparison of RMSE for space headway prediction between LSTM with Moving Average, standard Kalman and RTS noise reduction filters, and LSTM without noise reduction filter.

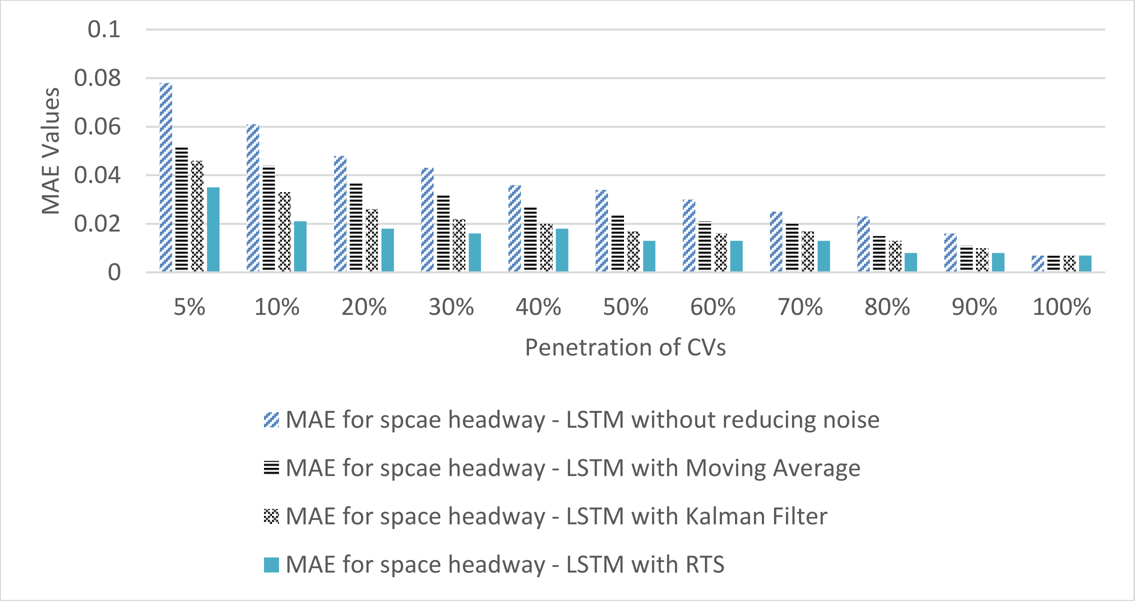


Fig. 3.10. Comparison of MAE for space headway prediction between LSTM with Moving Average, standard Kalman and RTS noise reduction filters, and LSTM without noise reduction filter.

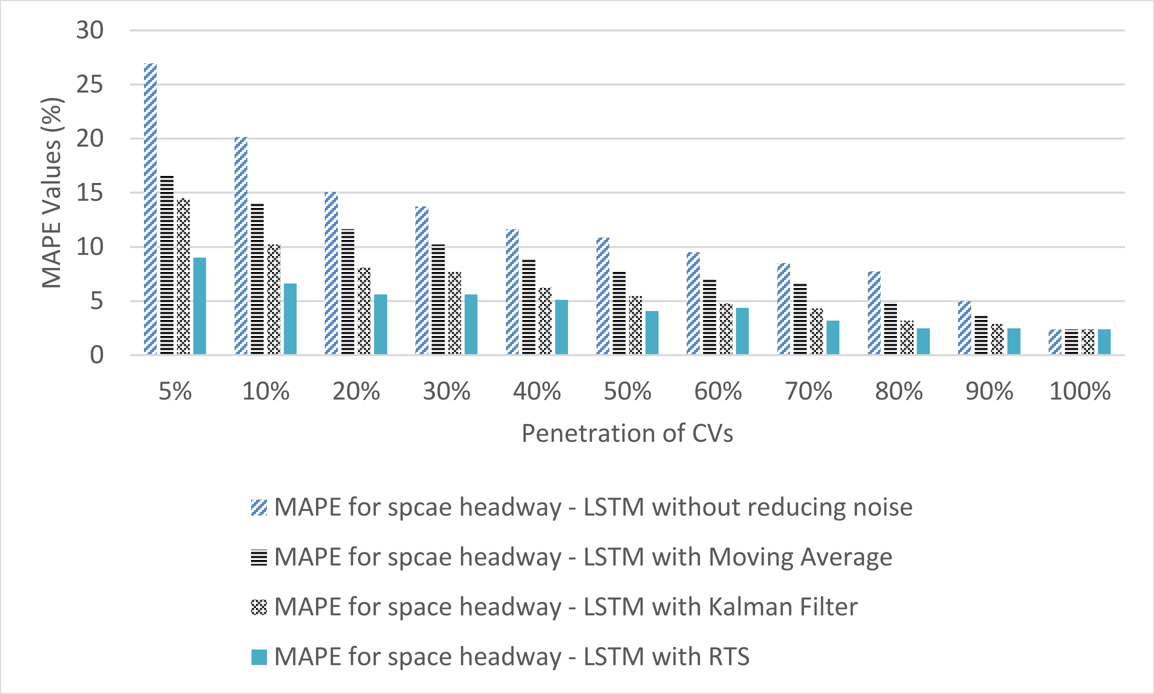
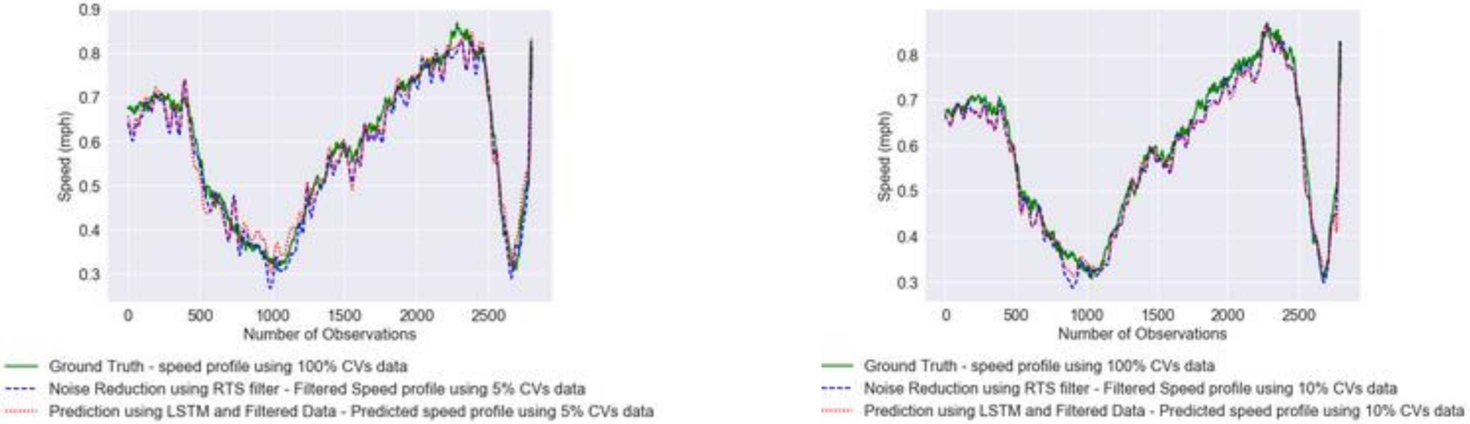


Fig. 3.11. Comparison of MAPE for space headway prediction between LSTM with Moving Average, standard Kalman and RTS noise reduction filters, and LSTM without noise reduction filter.

As shown in Figure 3.11, compared to the LSTM alone, LSTM combined with the RTS noise reduction model reduced MAPE from 27% (using LSTM only) to 9% for space headway prediction with a 5% penetration of CVs. The results indicated that LSTM combined with the RTS noise reduction model reduced MAPE ranges from 2% to 18% for space headway prediction with different penetrations of CVs compared to the corresponding MAPEs for LSTM alone.

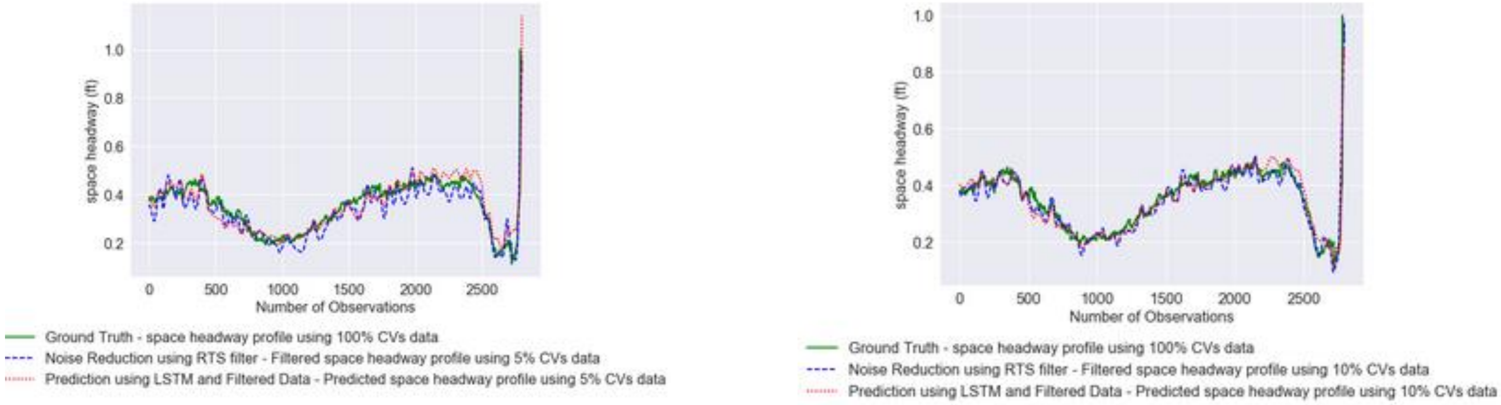
Figures 3.12 and 3.13 compare the ground truth data (actual speed/space headway using 100% connected vehicles)with the speed and space headway profile with a low penetration of CVs after reducing noise using RTS and the predicted speed and space headway profile using LSTM model with 5% to 40% penetration rates of CVs as examples. These figures illustrate the predicted speed profiles of speed and space headway visually.



(a) 5% CVs

(b) 10% CVs

Fig. 3.12. Comparison of (1) Ground truth, (2) Noise reduction using RTS filter, and (3) Prediction using LSTM and filtered data from 5% to 30% penetration rates of CVs.



(a) 5% CVs

(b) 10% CVs

Fig. 3.13. Comparison of (1) Ground truth, (2) Noise reduction using RTS filter, and (3) Prediction using LSTM and filtered space headway data from 5% to 30% penetration rates of CVs.

Table 3-I summarizes the RMSE, MAE and MAPE values using LSTM combined with the RTS models. These values indicate that the prediction model performs better with increasing penetration of CVs. The significant difference between the predicted and actual value with 100% CVs data was analyzed, with Table 3-II providing a summary of statistical significance test to identify if the predicted data is significantly different from the actual data. The significance test was conducted at a 95% confidence interval. The results indicated that the predicted speed using only LSTM is significantly different from the actual value for CV penetration ranging from 5% to 50%, while space headway is significantly different from the actual value for the 5% to 60% penetration range. The statistical analyses showed that the predicted space headway using LSTM combined with RTS is not significantly different from the actual value from 5% to 100% CV penetration. In this analysis, individual vehicle-generated data was aggregated at time intervals of one-tenth of a second to assess the prediction of the speed and space headway, the results indicating that the LSTM method requires less than a hundred milliseconds (80ms) to predict the speed and space headway, within an acceptable limit for real-time traffic applications.

Table 3-I Summary of RMSE, MAE and MAPE values using LSTM with RTS

Traffic Data	**Measure of Effectiveness**	**Penetration of Connected Vehicles**										
		5%	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
		LSTM with RTS										
Speed	**RMSE**	0.035	0.027	0.025	0.017	0.018	0.016	0.018	0.017	0.015	0.015	0.013
	MAE	0.027	0.023	0.017	0.016	0.015	0.013	0.012	0.012	0.01	0.01	0.009
	MAPE (%)	4.99	4	3.1	3	2.9	2.57	2.49	2.45	2.42	2.4	2.09
Headway	**RMSE**	0.051	0.029	0.028	0.024	0.028	0.022	0.024	0.027	0.02	0.019	0.017
	MAE	0.035	0.021	0.018	0.016	0.018	0.013	0.013	0.013	0.008	0.008	0.007
	MAPE (%)	9.02	6.62	5.6	5.6	5.1	4.07	4.36	3.2	2.48	2.49	2.4

Table 3-II Summary of Statistical Significance Test

Traffic Data		**Penetration of Connected Vehicles**									
	5%	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
		LSTM without noise reduction filter									
Speed	×	×	×	×	×	×	√	√	√	√	√
Space headway	×	×	×	×	×	×	×	√	√	√	√
	LSTM with RTS										
Speed	√	√	√	√	√	√	√	√	√	√	√
Space headway	√	√	√	√	√	√	√	√	√	√	√

Note: × = the actual and predicted values significantly different with 95% confidence interval; √ = the actual and predicted values are not significantly different with 95 % confidence interval.

3.6 Summary and Conclusions

This study investigated an LSTM model combined with a noise reduction model to predict traffic data using a vehicle penetration rate ranging from 5% to 100%. The traffic data for speed and space headway used here were generated from the Enhanced Next Generation Simulation (NGSIM) dataset, which contains vehicle trajectory data for every one-tenth of a second. The analyses revealed that the model developed here for traffic data prediction can predict speed and space headway for different penetrations of connected vehicles with no significant difference from the ground truth data. Specifically, LSTM combined with a RTS noise reduction model reduced MAPE from 19% to 5% for speed prediction and from 27% to 9% for space headway prediction at a 5% penetration of CVs compared to LSTM alone. The reduction of MAPE value ranges from 1% to 14% for speed and 2% to 18% for space headway prediction with penetration of CVs ranges from 5% to 100% compared to LSTM alone. The statistical significance test with a 95% confidence interval confirmed that the predicted speed and space headway using LSTM combined with RTS is not significantly different from the ground truth for 5% to 100% CV penetration. A comparison of the three noise reduction models (Moving Average, standard Kalman filter and RTS filter) suggests that LSTM combined with RTS can achieve the best prediction performance in terms of RMSE, MAE and MAPE. In addition, the prediction accuracy of speed and space headway prediction improve as the penetration of CVs increases.

3.7 Acknowledgments

This research is based upon research partially supported by the USDOT Center for Connected Multimodal Mobility (Tier 1 University Transpiration Center) Grant. Any

opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the USDOT Center for Connected Multimodal Mobility, and the U.S. Government assumes no liability for the contents or use thereof.

3.8 References

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization." *Journal of Machine Learning Research* 13, no. Feb (2012): 281-305.

Chowdhury, M., A. Sadek, Y. Ma, Y., N. Kanhere, & P. Bhavsar (2006). “Applications of artificial intelligence paradigms to decision support in real-time traffic management,” *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 1968(1), pp 92-98.

Du, Y., Chowdhury, M., Rahman, M., Dey, K., Apon, A., Luckow, A., & Ngo, L. B. (2017). “A Distributed Message Delivery Infrastructure for Connected Vehicle Technology Applications,” *IEEE Transactions on Intelligent Transportation Systems, 19*(3), 787-801.

Ervin, R.D., MacAdam, C.C., Gilbert, K., & Tchoryk, P. (1991). “Quantitative characterization of the vehicle motion environment (VME),” In *Proceedings of the 2nd Vehicle Navigation and Information Systems Conference, 20 -23 October 1991*, Dearborn, Mich. Society of Automotive Engineers, Inc., Warrendale, Penn. pp. 1011-1029. doi:10.1109/VNIS.1991.205846.

Gadsden, S. A., & Lee, A. S. (2017). “Advances of the smooth variable structure filter: square-root and two-pass formulations,” *Journal of Applied Remote Sensing*, 11(1), 015018.

Hochreiter, S., Schmidhuber, J., (1997). “Long short-term memory,” *Neural Comput*. 9 (8), 1735–1780.

Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z., & Zhang, H. (2017). “Traffic Prediction Based on Random Connectivity in Deep Learning with Long Short-Term Memory,” *arXiv preprint arXiv:1711.02833*.

Ishak, S., Kotha, P. & Alecsandru, C. (2003). “Optimization of dynamic neural network performance for short-term traffic prediction,” *Transport. Res. Rec.: J. Transport*. Res. Board 1836, 45–56.

Kingma D. P. & J. Ba. (2014). “Adam: A method for stochastic optimization.” [Online]. Available: https://arxiv.org/abs/1412.6980

Kalman, R. E. (1960). “A new approach to linear filtering and prediction problems,” *Journal of basic Engineering*, 82(1), 35-45.

Khan, S. M., Dey, K. C., & Chowdhury, M. (2017). “Real-time traffic state estimation with connected vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, 18(7), 1687-1699.

Kanagaraj, V., Asaithambi, G., Toledo, T., and Lee, T.C. 2015. “Trajectory data and flow characteristics of mixed traffic,” *Transportation Research Record*, 2491: 1‑11. doi:10.3141/2491-01.

Lingras, P., Sharma, S., & Zhong, M., (2002). “Prediction of recreational travel using genetically designed regression and time-delay neural network models,” *Transport. Res. Rec.: J. Transport. Res. Board* 1805, 16–24.

Liu, J. & Khattak, A.J., (2016). “Delivering improved alerts, warnings, and control assistance using basic safety messages transmitted between connected vehicles,” *Transportation research part C: emerging technologies*, 68, pp.83-100.

M. Montanino & V. Punzo (2013). “Making NGSIM data usable for studies on traffic flow theory,” *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2390, pp. 99–11, 2013.

Ma, X., Tao, Z., Wang, Y., Yu, H., & Wang, Y. (2015). “Long short-term memory neural network for traffic speed prediction using remote microwave sensor data,” *Transportation Research Part C: Emerging Technologies*, 54, 187-197.

Ma, Y., Chowdhury, M., Sadek, A., & Jeihani, M. (2009). “Real-time highway traffic condition assessment framework using vehicle–infrastructure integration (VII) with artificial intelligence (AI),” *IEEE Transactions on Intelligent Transportation Systems*, 10(4), 615-627.

Ma, Y., Chowdhury, M., Sadek, A., & Jeihani, M. (2012). “Integrated traffic and communication performance evaluation of an intelligent vehicle infrastructure integration (VII) system for online travel-time prediction,” *IEEE Transactions on Intelligent Transportation Systems*, 13(3), 1369-1382.

NGSIM (2006). “NGSIM I-80 Data Analysis,” Cambridge Systematics, Inc., Cambridge, MA, USA, Summary report, 2006.

Ossen, S., & Hoogendoorn, S.P. (2008). “Validity of trajectory-based calibration approach of car-following models in presence of measurement errors,” *Transportation Research Record*, 2088: 117‑125. doi:10.3141/2088-13.

Park, D., & Rilett, L.R., (1999). “Forecasting freeway link travel times with a multilayer feedforward neural network,” *Comput. Civ. Infrastruct. Eng.* 14 (5), 357–367.

Punzo, V., Formisano, D., & Torrieri, V. (2005). “Part 1: traffic flow theory and carfollowing: nonstationary Kalman filter for estimation of accurate and consistent car-following data,” *Transportation Research Record*, 1934: 1‑12. doi:10.3141/1934-01.

Punzo, V., Borzacchiello, M.T., & Ciuffo, B. (2011). “On the assessment of vehicle trajectory data accuracy and application to the Next Generation SIMulation (NGSIM) program data,” *Transportation Research Part C: Emerging Technologies*,19: 1243‑1262. doi:10.1016/j.trc.2010.12.007.

Rauch, H. E., Striebel, C. T., & Tung, F. (1965). “Maximum likelihood estimates of linear dynamic systems,” *AIAA journal*, 3(8), 1445-1450.

Ren, H., Song, Y., Liu, J., Hu, Y., & Lei, J. (2017). “A Deep Learning Approach to the Prediction of Short-term Traffic Accident Risk,” *arXiv preprint arXiv:1710.09543*.

Rim, H., Park, S., Oh, C., Park, J., & Lee, G. (2016). “Application of locally weighted regression-based approach in correcting erroneous individual vehicle speed data,” *Journal of Advanced Transportation*, 50(2): 180‑196. doi:10.1002/atr. 1325.

Thiemann, C., Treiber, M., & Kesting, A. (2008). “Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data,” *Transportation Research Record*, 2088: 90‑101. doi:10.3141/2088-10.

Vanajakshi, L., & Rilett, L. R. (2004). “A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed,” In *Intelligent Vehicles Symposium*, 2004 IEEE (pp. 194-199). IEEE.

Van Lint, J.C., (2006). “Reliable real-time framework for short-term freeway travel time prediction,” J. *Transport. Eng.* 132 (12), 921–932.

Van Lint, J.W.C., Hoogendoorn, S.P., & Zuylen, H.V. (2002). “Freeway travel time prediction with state-spaced neural networks: modeling state-space dynamics with recurrent neural networks,” *Transport. Res. Rec.: J. Transport. Res. Board* 1811, 30–39.

Van Lint, J.W.C., Hoogendoorn, S.P. & Zuylen, H.V., (2005). “Accurate freeway travel time prediction with state-space neural networks under missing data,” *Transport. Res. Part C* 13, 347–369.

Vanajakshi, L. & L. R. Rilett (2004). “A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed,” *IEEE Intelligent Vehicles Symposium*, pp. 194-199.

Vlahogianni, E.I., Karlaftis, M.G. & Golia, J.C. (2014). “Short-term traffic forecasting: where we are and where we’re going,” *Transport. Res. Part C* 43 (1), 3–19.

Zhao, Z., Chen, W., Wu, X., Chen, P. C., & Liu, J. (2017). “LSTM network: a deep learning approach for short-term traffic forecast,” *IET Intelligent Transport Systems*, 11(2), 68-75.

CHAPTER 4

CV APPLICATION: DYNAMIC ROUTING

4.1 Introduction

Non-recurring traffic incidents cause one-quarter of all traffic congestion in the United States (NTIMC, 2012). A total of 33,561 people died in motor vehicle crashes in 2012 (NTHSA, 2013). According to the U.S. Department of Transportation, the total societal cost of incidents exceeds $200 billion annually (Blincoe et al., 2013). Connected vehicle technology has the potential to improve safety and mobility, and reducing greenhouse gas emissions (Thomas, 2013). Cooperative systems in a connected vehicle environment can be divided into two main areas: Vehicle-to-Vehicle (V2V) cooperation, and Vehicle-to-Infrastructure cooperation. In this study, the author developed a dynamic routing strategy in a cooperative vehicle environment (V2V and V2I) to reduce recovery time and increase safety on a freeway. Through detection and verification, traveler information, response (tow truck arrival), scene management and traffic control, and quick clearance and recovery activities, the impact of such congestion can be eased with the implementation of a traffic incident management (TIM) plan (see Figure 4.1). Dynamic routing strategies in a cooperative vehicle environment have not only relieved congestion, but also have led to economic savings, energy conservation, environmental benefits, and improved health and safety (CHART, 2013).

Routing vehicles to alternate routes in the event of a major incident is one of the strategies to increase driver safety, ease congestion, improve traffic operations, and reduce negative environmental impact (Chowdhury et al., 2007). The traditional methods to route vehicles include relaying messages via Variable Message Signs (VMS), Highway Advisory Radio (HAR) services, etc. However, diverting vehicles to an alternate route is not always a feasible solution (Fries et al., 2007). Further, due to additional diverted traffic, the alternate route may face heavy congestion. The solution to this problem is to implement dynamic routing strategies. Dynamic routing strategy or Dynamic Traffic Assignment (DTA) assigns routes based on the time-varying demand (Mahmassani, 2001). However, implementation of these strategies in the real world requires an array of infrastructure sensors, which can collect, transmit and process data efficiently to reroute vehicles when needed. The two key concerns for a traffic incident management plan is verification time and response time (arrival time of a tow truck). Slower verification time or late arrival of a tow truck results in an increased queue length and secondary incidents on the main route during an incident. Hence, the impact of a dynamic routing strategy was evaluated in a traffic microsimulation environment by developing different scenarios based on different verification times and arrival times of a tow truck.

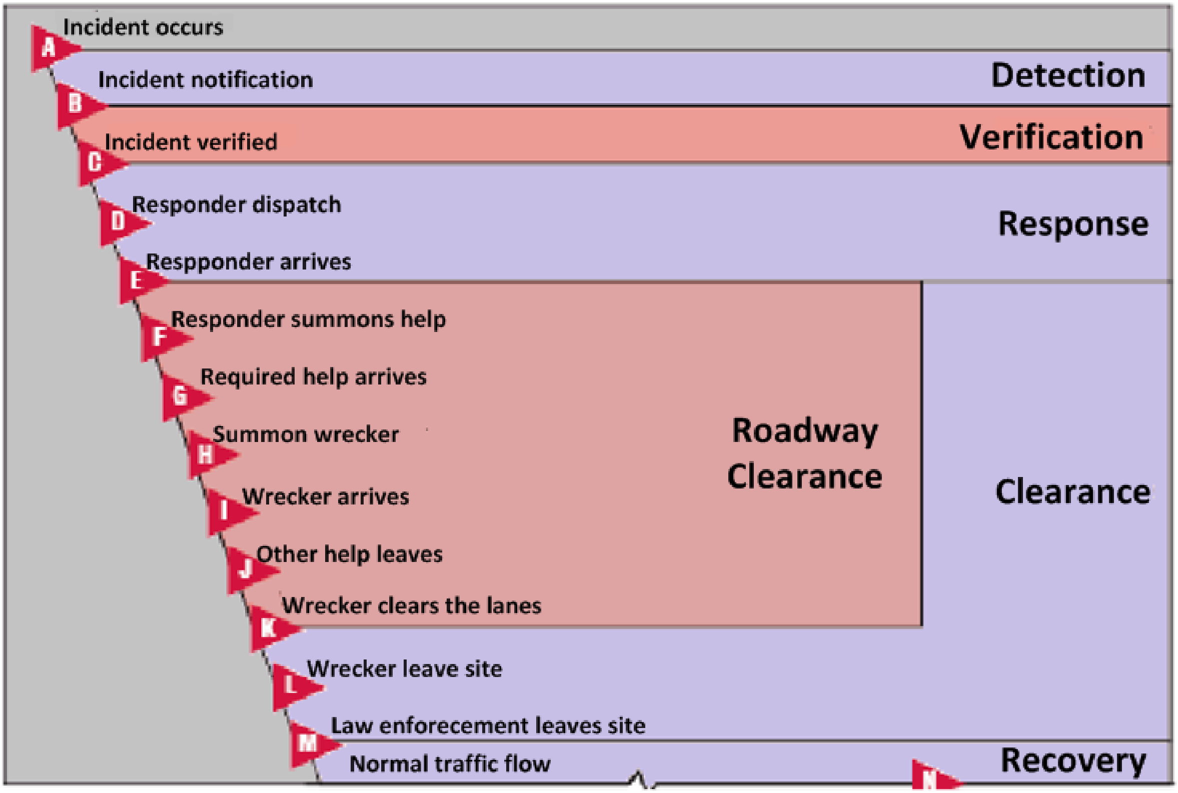


Fig. 4.1. Linear incident timeline (adopted from SCDOT 2017).

4.2 Related Works

Dynamic traffic routing can be defined as the real-time point diversion of traffic during non-recurrent congestion. Non-recurrent congestion, such as congestion caused by traffic incidents, is the source of over 50% of all travel time delays (Sisiopiku et al., 1994). Dynamic routing provides real-time navigation information, which is based on traffic conditions at the incident scene, to emergency responders (Incident, 2013). Sisiopiku et al. conducted a simulation test on two networks (Birmingham, Alabama and Chicago, Illinois) to evaluate the capabilities of dynamic traffic assignment to improve existing incident management (Sisiopiku et al., 1994). In this study, they found that the impact of traffic incidents could be improved with existing incident management plans that utilize route diversion strategies. Incident management authorities in South Carolina only consider route diversion for the most severe and long-lasting incidents, since longer incidents cause longer queues (SCDOT 2017). Another study showed that impact analysis of route diversion included costs of police units, VMS and HAR use, and communication and infrastructure costs (Chowdhury et al., 2007). Thus, local agencies in South Carolina only applies route diversion strategies in the most severe traffic incidents due to their costs.

4.3 Dynamic Routing Strategies in a Cooperative Vehicle Environment

The recent advances in connected vehicle technologies have paved the way to solve mobility problems. With the help of connected vehicle technologies, an individual vehicle can access incident information through infrastructure sensors and vehicles near the scene to make a decision. The dynamic routing strategy proposed in this study considered a connected vehicle environment in which the vehicles are routed based on the average speed information provided by infrastructure sensors on the main route as well as alternate routes.

In VISSIM, micro-simulation software, traffic diversion strategies can be modeled using Vehicle Actuated Programming (VAP). The dynamic routing strategy developed in this study was based on the average speed on I-26 and two alternate routes. In the dynamic routing strategy, traffic has been assigned to each alternate route based on the time-varying demand, keeping the average speed as a threshold to avoid congestion on alternate routes. For instance, Figure 4.2 demonstrates VAP (Vehicle Actuated Programming) logic for dynamic routing strategies that are developed for the section of I-26 in Berkeley, Orangeburg and Dorchester Counties with two alternate routes. These routes are recommended for the section in the SCDOT contingency plan for traffic incidents (see Figure 4.2).

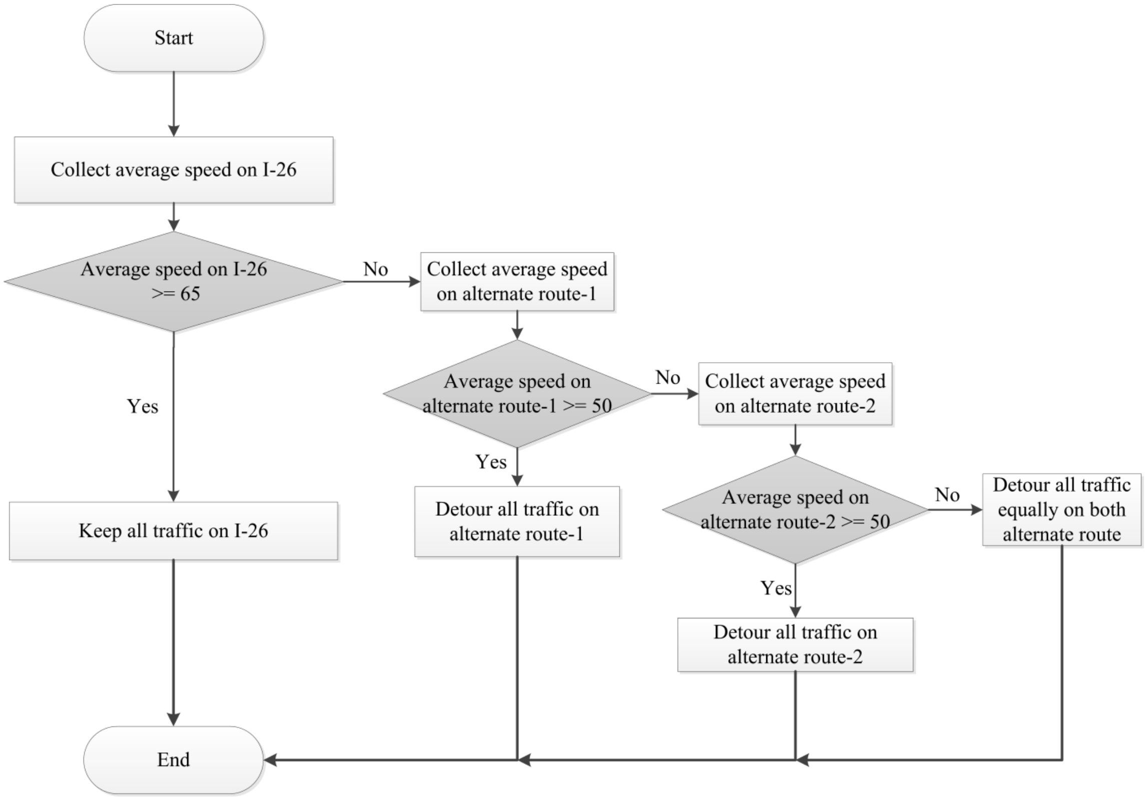

Start
Collect average speed on I-26
Average speed on I-26 >= 65
No
Collect average speed on alternate route-1
Yes
Average speed on alternate route-1 >= 50
No
Collect average speed on alternate route-2
Yes
Keep all traffic on I-26
Detour all traffic on alternate route-1
Average speed on alternate route-2 >= 50
No
Detour all traffic equally on both alternate route
Yes
Detour all traffic on alternate route-2
End

Fig. 4.2. VAP logic for the dynamic routing strategy

4.4 A Case Study: On a Section of I-26 in Charleston, South Carolina

The first phase in the two-phase study included the development of a calibrated traffic simulation network of the section of the I-26 in Charleston, SC. In the second phase, the dynamic strategy was modeled using the Vehicle Actuated Programming (VAP) feature provided by the traffic microsimulation software VISSIM 5.40 (PTV VISSION, 2012). As shown in Figure 4.3, the I-26 network included 17.6 mi of freeway with three interchanges, and the length of the two alternate routes is 40 miles and 30 miles respectively. The strategy was then evaluated for different incident scenarios.

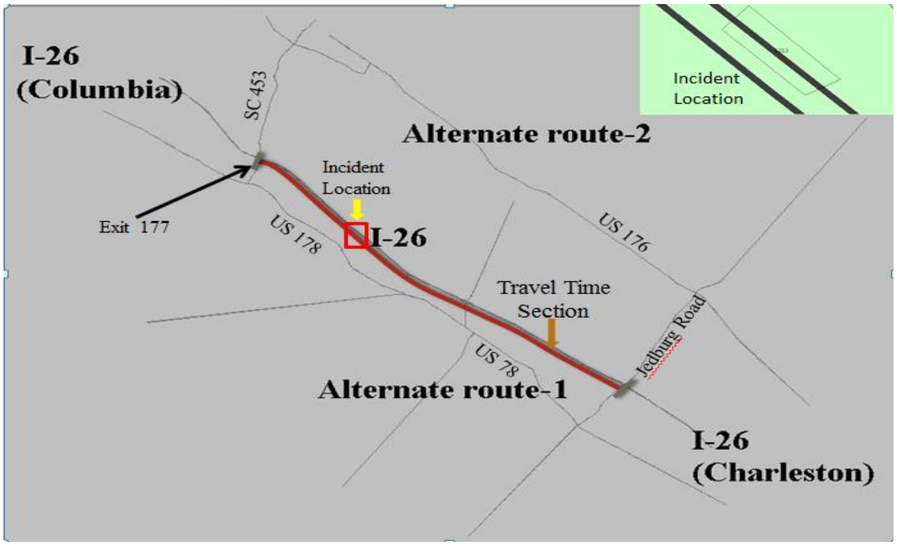


Fig. 4.3. Network developed in VISSIM

4.4.1 Traffic Simulation Network Development

As shown in Figure 4.4, the model development process included VISUM, a macroscopic planning software developed by PTV vision (PTV VISSION, 2012). The shape file obtained from the SCDOT GIS database was imported to VISUM. Roadway geometric data collected in the field was used to modify the network according to existing conditions. The network was then exported to VISSIM.

The 2012 Annual Average Daily Traffic (AADT) volume, which is collected from the SCDOT traffic count website (SCDOT, 2012), for each gateway point was converted to Directional Design Hourly Volume (DDHV) to develop an origin-destination (OD) matrix. Travel time data for the main route and two alternate routes were collected along with the volume data for key links for the calibration of the model. The final Origin-Destination (OD) volumes reflected twice the evening peak hour for any given location. The next step in the process was to use Dynamic Traffic Assignment (DTA) in the network to get turning volumes for each interchange and intersection. The travel time and volume data collected for calibration were checked against the simulation output for each run. During this iterative process, link cost, speed distribution and driver behavior parameters were adjusted as needed to recreate the volumes and travel times observed on the site visit. After the volumes and travel times were within 10 percent, the model was considered calibrated.

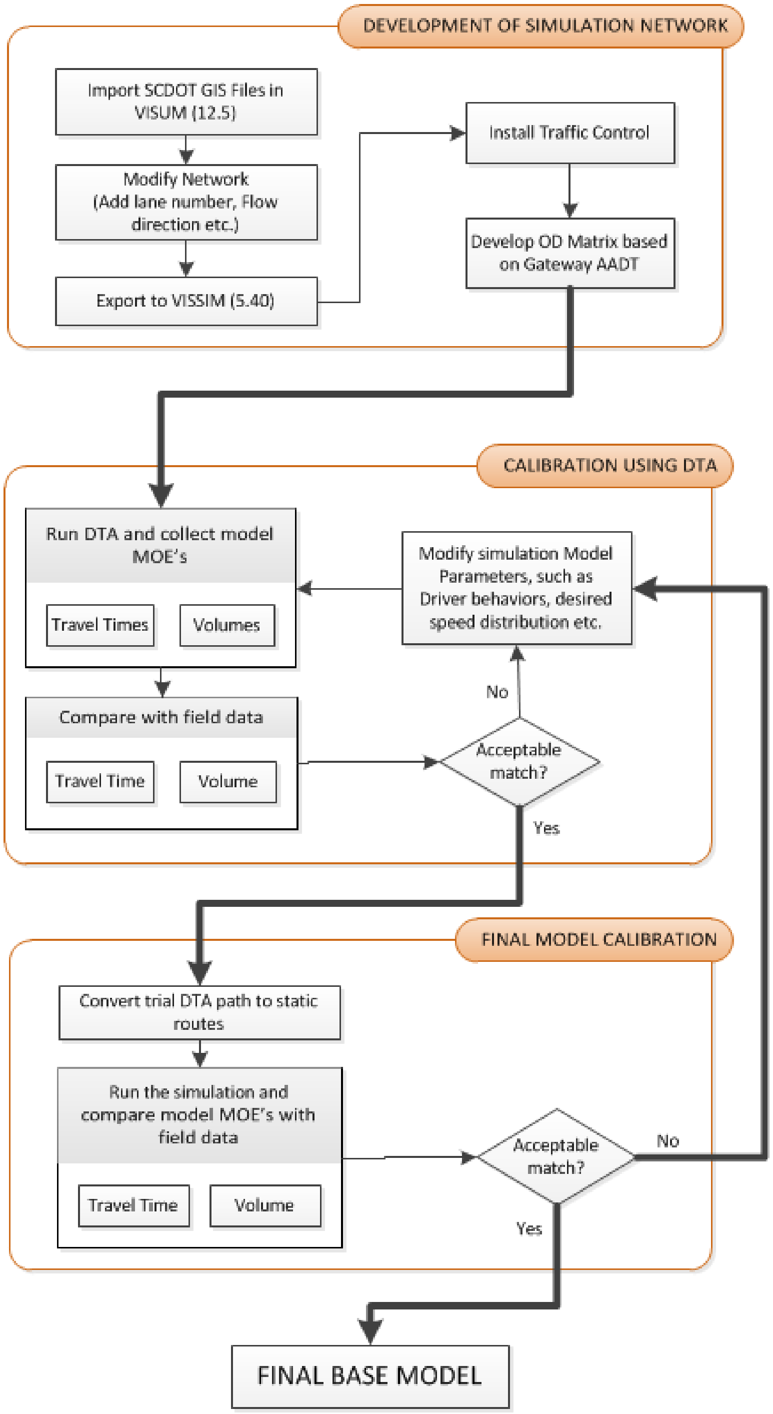

DEVELOPMENT OF SIMULATION NETWORK
Import SCDOT GIS Files in VISUM (12.5)
Modify Network (Add lane number, Flow direction etc.)
Export to VISSIM (5.40)
Install Traffic Control
Develop OD Matrix based on Gateway AADT
CALIBRATION USING DTA
Run DTA and collect model MOE's
Travel Times
Volumes
Modify simulation Model Parameters, such as Driver behaviors, desired speed distribution etc.
No
Compare with field data
Travel Time
Volume
Acceptable match?
Yes
FINAL MODEL CALIBRATION
Convert trial DTA path to static routes
Run the simulation and compare model MOE's with field data
Travel Time
Volume
Acceptable match?
No
Yes
FINAL BASE MODEL

Fig. 4.4. Simulation model development.

4.4.2 Incident Simulation

Since the incident management authority in South Carolina only considers route diversion for the most severe and long-lasting incidents, incidents that blocked both lanes of I-26 WB were modeled in the final calibrated simulation model. The incidents were created using the public transit stop feature of VISSIM 5.40. The total incident time was 46 minutes and the total simulation time was 7200 seconds.

4.4.3 Route Diversion Strategies

In order to develop the strategy at first, the base model was simulated without an incident. The link speed data from the simulation model output for the main route and two alternate routes were then used to calculate average speed under normal conditions. These average speed constants were used as speed thresholds in the dynamic routing strategy. As shown in Figure 4.3, the VAP logic will be executed at every simulation second, and it will check the speeds of the main route and alternate routes via the detectors in the simulation model. Based on these average speed values, the vehicles whose final destination is the I-26 end towards Columbia (as shown in Figure 4.3) will be rerouted.

4.4.4 Simulation scenarios

The simulation scenarios were designed to capture the impact of the dynamic routing strategy. The scenarios were developed based on the two key time parameters of a traffic incident management plan; (1) verification time (see Table 4-I) and (2) response time (tow truck arrival time) (see Table 4-II). The base scenario for both parameters included existing traffic volume with the incident but without routing. The base model output without the incident and dynamic routing were also used as a base-line to represent normal conditions.

4.4.5 Simulation Outcomes and Discussion

The simulation analysis was performed for two components: verification time and arrival time at the incident scene, using the dynamic routing strategy. Then, dynamic routing strategies were compared with the base scenario without dynamic routing. The author also compared dynamic strategies with a range of verification times and response times (tow truck arrival time at the incident scene) to investigate the effect of these strategies. The following sections discuss the results of the simulation analysis.

Table 4-I Simulation Scenarios based on Verification Time.

Simulation scenarios	Verification Time
Base scenario with the incident	5 minutes verification time
Base scenario with the incident and dynamic routing	5 minutes verification time
Base scenario with the incident and dynamic routing	4 minutes verification time
Base scenario with the incident and dynamic routing	3 minutes verification time
Base scenario with the incident and dynamic routing	2 minutes verification time
Base scenario with the incident and dynamic routing	1 minute verification time
Base scenario with the incident and dynamic routing	0 minute verification time

Table 4-II Recovery Time based on Arrival Time at Incident Scene

Simulation scenario	Recovery time (second)
Base scenario with the incident	10 minutes arrival time at the incident scene
Base scenario with the incident	9 minutes arrival time at the incident scene
Base scenario with the incident	8 minutes arrival time at the incident scene
Base scenario with the incident and dynamic routing	10 minutes arrival time at the incident scene
Base scenario with the incident and dynamic routing	9 minutes arrival time at the incident scene
Base scenario with the incident and dynamic routing	8 minutes arrival time at the incident scene

4.4.6 Effect of Incident on Verification Time

Figure 4.5 represents the effect of the incident on verification time for both incident without dynamic routing and incident with dynamic routing scenarios. The total duration of incident clearance time was 46 minutes, including a 5-minute verification time for the base incident scenario.

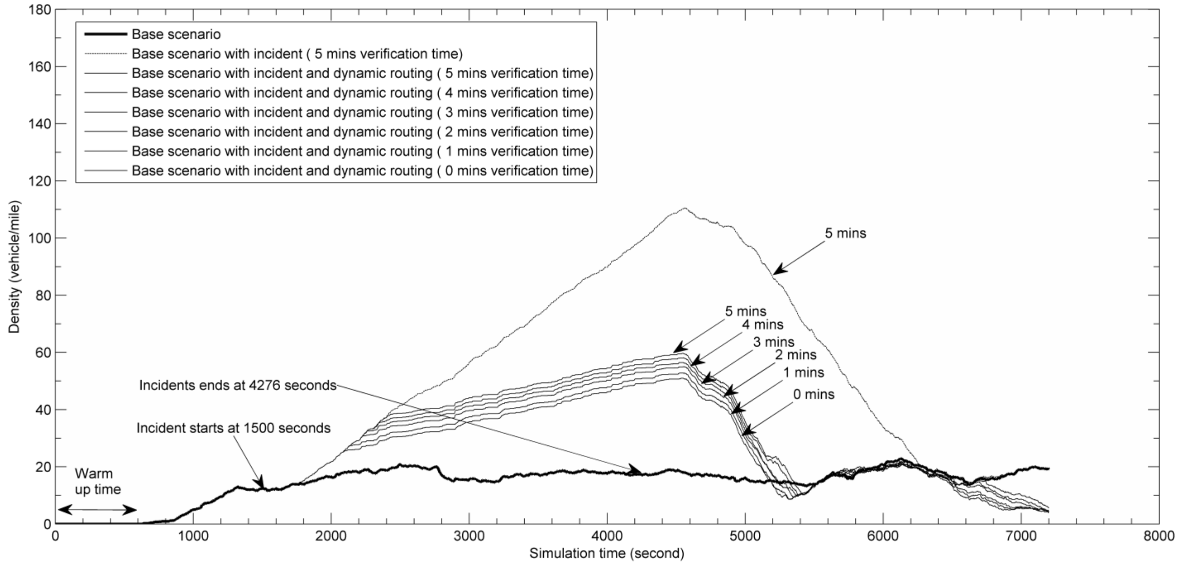


Fig. 4.5. Effect of dynamic routing strategies on incident verification time.

Figure 4.5 shows that in the last 800 seconds of simulation the density profile of the I -26 freeway is reducing, as traffic is rerouting to alternate roads when it reaches maximum base density. It was found that the recovery time was reduced by 52 % with the dynamic routing strategy. Recovery time was also reduced by a 5% per minute reduction of verification time with the dynamic routing strategy.

4.4.7 Effect of response time (tow truck arrival time at the incident scene)

Figure 4.6 represents the density profile for 8, 9, and 10 minute arrival times with and without dynamic strategies. Tow truck arrival times of 8 to 10 minutes, which is the time between notification of the incident to response units and their arrival at the incident scene, was used based on Chowdhury et al., 2007. The total incident clearance time for the base scenario with the incident was the same as the verification time scenario. For the base case, the arrival time was 10 minutes. From Figure 4.6, it is clear that the recovery time was significantly reduced. The recovery time was reduced by 60%, 61% and 61.27% for 10, 9, and 8 minutes, respectively.

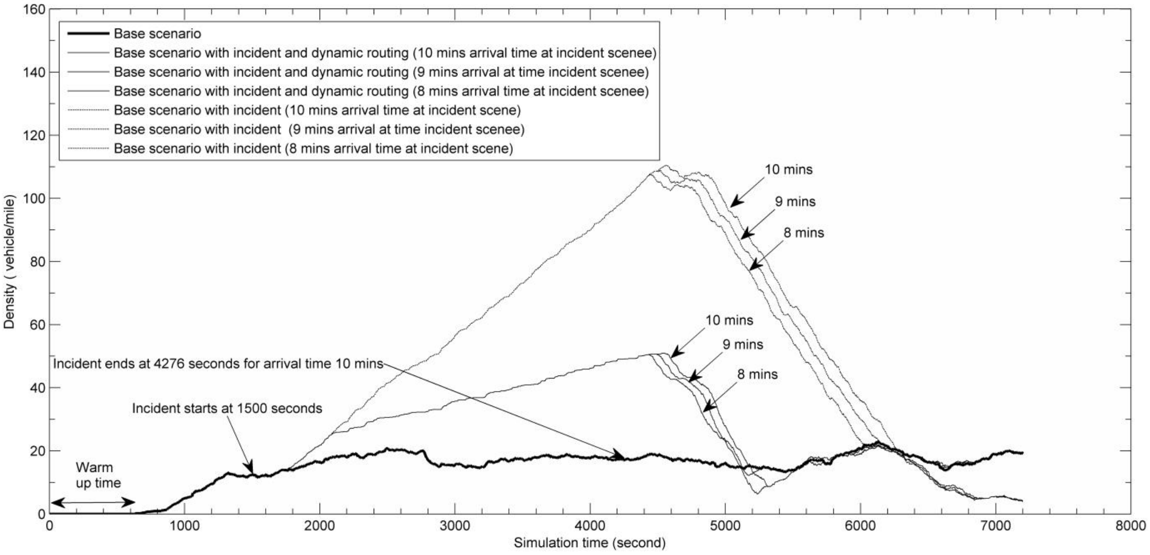


Fig. 4.6. Effect of dynamic routing strategies on arrival time at incident scene.

4.5 Summary and Conclusions

In this study, dynamic routing strategies have been developed to reduce recovery time for a major incident in a cooperative vehicle environment that would result in increased driver safety, improved traffic operations, and reduced negative environmental effects. Simulation experiments were conducted on a segment of roadway of I-26 using a traffic micro-simulator. The study confirmed that dynamic routing in a cooperative vehicle environment could prove to be a powerful tool in reducing traffic incident induced impacts on the network.

The case study demonstrated that incidents that are not cleared quickly could affect freeway traffic operations. The results also demonstrated that the dynamic routing strategy reduced recovery time by 52% with a reduced verification time and an on average 61% reduction in arrival time of a tow truck at the incident scene. Recovery time was also reduced by 5% per minute reduction of verification time with the dynamic routing strategy. This study also revealed that dynamic routing has an effect on verification time and tow truck arrival time. Thus, timely verification of an incident and quick response to the incident scene results in reduced recovery time and restoration of normal traffic conditions.

4.6 References

Blincoe, L.J., Seay, A.G., Zaloshnja, E.,Miller, T.R., Romano, E.O., Luchter, S. & Spicer R.S. (2002). “The economic impact of motor vehicle crashes 2000,”. *Report no. DOT HS-809-446.* Washington, DC: National Highway Traffic Safety Administration.

CHART (2013). Maryland CHART Project. July, 26, 2013. http://www.fhwa.dot.gov/cadiv/segb/views/document/sections/section8/8_5_3.cfm, Accessed 6/26, 2013.

Chowdhury, M., Fries, R., Ma, Y., Dunning, A., Hamlin, A. C., Ozbay, K., & Sarasua, W. (2007). “Benefit–Cost Analysis of Accelerated Incident Clearance,” *Final Report. Prepared for the Federal Highway Administration and the South Carolina Department of Transportation,* FHWA-SC-07-01.

Fries, R., Inamdar, I., Chowdhury, M., Taaffe, K., & Ozbay, K. (2007). “Feasibility of traffic simulation for decision support in real-time regional traffic management,” *Transportation Research Record: Journal of the Transportation Research Board*, 2035(1), 169-176.

Incident Scene Pre-Arrival Staging Guidance for Emergency Responders. Available online at http://www.iteris.com/cvria/html/applications/app55.html.

Mahmassani, H. S. (2001). “Dynamic network traffic assignment and simulation methodology for advanced system management applications,” Networks and Spatial Economics, 1(3-4), 267-292.

NTIMC (2012). About US. Available online at http://ntimc.transportation.org/Pages/AboutUs.aspx, Accessed 5/22, 2012.

National Highway Traffic Safety Administration. 2013. 2012 motor vehicle crashes: overview. Report no. DOT HS-811-856. Washington, DC: US Department of Transportation.

Sisiopiku, V. P., Li, X., Mouskos, K. C., Kamga, C., Barrett, C., & Abro, A. M. (2007). “Dynamic traffic assignment modeling for incident management,”. *Transportation Research record: Journal of the transportation research board*, 1994(1), 110-116.

Thomas A. Dingus, Director of the Virginia Tech Transportation Institute. Connected Vehicles: Our Future is Happening now. Available online at http://itsworldcongress.org/blog/connected-vehicles-our-future-is-happening-now/.

PTV Vision. VISSIM 5.30-04 User Manual. 5.40-04, 2012.

SCDOT. Traffic count website. Available online at http://www.scdot.org/getting/annualTraffic.aspx.

SCDOT. (2017). Integration of the Incident Command System (ICS) Protocol for Effective Coordination of Multi-Agency Response to Traffic Incidents. Available online at https://ntl.bts.gov/lib/61000/61900/61943/SPR-699-FHWA-SC-17-07.pdf. Accessed 5/22, 2012.

CHAPTER 5

CAV APPLICATION: COOPERATIVE ADAPTIVE CRUISE CONTROL (CACC)

5.1 Introduction

Advanced driver assistance systems are developed to assist drivers with different driving tasks. Adaptive cruise control (ACC) is one of the most common advanced driver assistance systems in traditional vehicles without real-time connectivity support between vehicles (Basak et al., 2013). An ACC system helps vehicles follow the leading vehicle at a certain gap by varying the vehicle speed. Cooperative adaptive cruise control (CACC) is an advanced ACC system that uses wireless communication between multiple connected vehicles in a platoon (CVRIA, 2016). Connectivity between the vehicles in a CACC platoon allows the vehicles to react much faster than the drivers in an ACC system (Jones, 2016). A CACC vehicle platoon could be compared to a train: the first vehicle in the platoon acts as the locomotive, and the other vehicles follow the leader (Zhao and Sun, 2013). The potential benefits of connected vehicles in improving safety and mobility and decreasing environmental impact have already been demonstrated through studies involving simulation and real-world pilot testing (Dey et al., 2016; Ma et al., 2012; Ma et al., 2009). ACC solely uses on-board sensors to determine the gap between vehicles and the relative speed between the subject vehicle and the vehicle immediately in front; a CACC system makes use of vehicle-to-vehicle communications to obtain multiple preceding vehicles' positions and speeds and reacts on the basis of this information. Connected vehicles participating in a CACC platoon will allow vehicles to react more quickly to roadway traffic conditions and events than would be possible with sensor-based ACC, as each vehicle in a CACC platoon reacts simultaneously to the platoon leader. The comfortable space headway between vehicles in a CACC platoon depends on human response to different driving conditions and roadway events (Dey et al., 2016; Taieb-Maimon and Shinar, 2001).

As CACC systems will allow a small gap or headway between vehicles in a platoon, the systems would increase roadway capacity significantly (Dey et al., 2016). CACC systems also have potential positive impacts such as (*a*) reduced fuel consumption, (*b*) reduced greenhouse gas emissions, (*c*) reduced travel time, and (*d*) improved traffic safety (Van Arem et al., 2006). Despite these advantages, the adoption of CACC technology could be hindered if end users are unsure about the comfort of this technology and feel insecure about autonomous driving in a platoon (Jones, 2016). The time headway under autonomous CACC driving is much shorter than the perception–reaction delay in manual driving, and therefore it is challenging for drivers to adapt to CACC application (*9*). CACC operations may also result in sudden acceleration or deceleration, which might cause discomfort to vehicle occupants in a platoon (Wang and Wang, 2015). These negative concerns could cause users to be reluctant to use the CACC feature in a vehicle. In addition, if the vehicle-to-vehicle communication is not reliable and the communication delay is too large, the subject vehicle might collide with the vehicle immediately in front. This problem is closely tied to the communication delay requirement between vehicles in a platoon, driving behavior factors, and the automated CACC algorithm. This research focuses on driving behavior and describes the development of a framework for evaluating driver behavior models that can be used in CACC system design to enable consideration of a user acceptance criterion.

A CACC system targeted to obtain a high level of user acceptance must replicate a similar driving experience in each CACC vehicle, and the expectations a human occupant has for a comfortable driving experience should not be violated. Thus, a key factor in CACC system design is to use driver models that represent driver car-following behavior and reactions to the surrounding environment to help maintain the longitudinal position and the minimum gap from the vehicle immediately in front of the subject vehicle in a CACC platoon. A reliable and efficient CACC system design requires the consideration of acceptable vehicle dynamics (i.e., acceptable acceleration or deceleration) and string stability (i.e., the amplification of the fluctuations in the vehicle's position, speed, and acceleration), as these factors help ensure user acceptance (i.e., comfort and safety) (Gunawan, 2012; Pueboobpaphan and Van Arem, 2010; Bose and Ioannou, 2001; Lei et al., 2012). Thus, the development of an evaluation framework is critical in assessing which driver car-following behavior model can provide a comfortable driving experience in terms of vehicle dynamics and string stability in a CACC platoon. The primary objective of this study is to develop a framework for evaluating driver car-following behavior models on the basis of user acceptability criteria to ensure that a CACC system maintains safe and comfortable vehicle dynamics and string stability in different roadway traffic states. In this research, the maximum acceleration or deceleration and the sum of squares of the errors (SSE) of the follower vehicles were measured to evaluate user acceptance in terms of vehicle dynamics and string stability.

The author evaluated and compared user acceptance, in terms of vehicle dynamics and string stability, for two driver car-following behavior models—the optimum velocity model (OVM) and the intelligent driver model (IDM)—in CACC system design through the evaluation framework developed in this study. To compare the suitability of these models with a CACC system, a system of first-order ordinary differential equations (ODEs) was derived for the OVM and the IDM for a platoon of vehicles.

In the next section of this research, related works on the evaluation of driver car-following behavior models for CACC system design are reviewed. The following section describes a framework for the evaluation of driver car-following behavior models in CACC system design. Details of a case study for the presented evaluation framework under the OVM and the IDM are then described. Finally, a concluding discussion is presented.

5.2 Related Work

This section discusses previous studies related to the use of driver car-following behavior models to ensure a safe and comfortable user experience for CACC vehicles. Each following vehicle in a CACC platoon has a preceding vehicle and a following vehicle; the last vehicle in the platoon has only a preceding vehicle (Zhao and Sun, 2013). Driver car-following behavior models are an integral component of CACC system design. The objective of driver car following behavior models is to replicate the behavior of a subject vehicle's driver while following a leader vehicle in a traffic stream as well as the driver's reaction to a stimulus that leads to acceleration or deceleration (Brackstone and McDonald, 1999). Driver car-following behavior has been extensively studied since the 1950s, and these studies have resulted in many models (Brackstone and McDonald, 1999; Panwai and

Dia, 2004; PTV VISSIM, 2012; Fritzsche, 1994). These models include (a) the Gazis–Herman–Rothery model, (b) the linear model, (c) the fuzzy logic–based model, (d) the collision avoidance model, (e) metamodels, (f) the OVM, (g) the IDM, and (h) psychophysical models. For CACC system design, potential driver car-following behavior models need to beevaluated for their efficacy in a CACC platoon design in different traffic states.

Van Arem et al. (2006) used the MIXIC traffic flow simulation model to evaluate the characteristics of traffic flow for CACC platooning. The MIXIC microscopic traffic simulation tool used a driver car-following behavior model, which resulted from the optimal control model developed by Burnham et al. (1974). A merging scenario was evaluated on a four-lane to three-lane highway. An analysis of the simulation results revealed that string stability and traffic capacity were improved with a CACC merging scenario compared with a non-CACC scenario. The user acceptance of the driver car following behavior model in terms of string stability and vehicle dynamics was not analyzed. Schakel et al. (2010) assessed traffic string stability and shockwaves for CACC and used the IDM driver car-following behavior model to evaluate traffic flow stability and shockwaves in mixed traffic; the study compared data from the field test and the CACC advisory system. It was found that shockwaves moved quickly in a CACC scenario if the penetration level of connected vehicles was 50% or higher. String stability was evaluated only on the basis of the position profile; the speed and acceleration profiles were not considered. Later, Zhao and Sun (2013) evaluated manual, ACC, and CACC scenarios in the Vissim traffic simulator for different platoon sizes and market penetrations. In another study, interactions were examined between vehicles in a platoon of six vehicles in shockwaves and were evaluated with the Wiedemann (1974) driver car-following behavior and lane-changing model in terms of vehicle dynamics. It was found that lane capacity increased significantly with a higher market penetration of CACC vehicles in the CACC platoon and that the platoon size had an insignificant impact on lane capacity increase. To evaluate the performance of CACC, Milanés and Shladover (2014) developed three control systems: (a) ACC that used real-world experimental data, (b) CACC control systems that used real-world experimental data, and (c) CACC control systems that used the IDM car-following model. Field experiments were performed with production vehicles to collect the actual speed, acceleration, and headway measurements of the vehicles.

The analysis revealed that CACC vehicles overcame the limitations of ACC vehicles (i.e., unstable driver car-following behavior and amplification of speed) and that the subject vehicle with the IDM controller did not closely follow the vehicle immediately in front of the subject vehicle in terms of change of speed. Lei et al. (2012) used the SUMO traffic simulator [which uses the Krauss (1997) driver car-following behavior model] with the MiXiM/OMNeT++ communication simulator to evaluate the string stability of a CACC platoon by varying the ratios of the communication packet loss, the frequency of the data being sent, and the time headway. It was found that a higher packet loss ratio reduced the performance of the string stability of the CACC platoon. Gunawan (2012) compared the vehicle dynamics of two vehicles (non-CACC scenario) with three driver car-following behavior models: the Gazis–Herman–Rothery model, the OVM with delay, and the IDM. Gunawan concluded that the OVM and the IDM could simulate different traffic states (i.e., the driver car-following behavior, free-flow, and congestion states) than the Gazis–Herman–Rothery model. The OVM produced realistic interactions between two vehicles in terms of vehicle dynamics with acceptable delay, and the IDM produced more rational vehicle dynamics with an additional driver delay parameter. Numerous studies have evaluated traffic flow efficiency through different car-following models. However, an evaluation framework for the application of a driver car-following behavior model in terms of user acceptance in CACC system design, with consideration of both vehicle dynamics and string stability, has not been studied.

5.3 Evaluation Framework of Driver Car-Following Behavior Models for CACC Controller Design

The general evaluation framework of a driver car-following behavior model for CACC system design comprises the design process for a CACC system and the evaluation of user acceptance in terms of vehicle dynamics and string stability. In the following subsections, the author explain the evaluation framework in detail.

5.3.1 Design Process of CACC System

5.3.1.1 *Extension of Driver Car-Following Models for CACC Platoon*

Before implementing traditional driver car-following behavior models for a CACC platoon, it is necessary to develop a system-level model by extending an existing driver car-following behavior model. This system-level model can represent a CACC system that allows adjustment of the CACC platoon speed and minimum headway simultaneously on the basis of information from multiple vehicles in the platoon. The key feature of this extension for a CACC platoon is that each CACC vehicle can react simultaneously to the platoon leader's speed while considering the immediately preceding vehicle's information. For example, there is one leader and five following vehicles in a CACC platoon. The first following vehicle receives the platoon leader's vehicle trajectory data (i.e., position, speed, and acceleration) in real time. Similarly, the first following vehicle acts as a leader vehicle for the second following vehicle, and the second vehicle receives the vehicle trajectory data from the first following vehicle. All of the following vehicles in the platoon adjust their own vehicle dynamics on the basis of the information of the immediately preceding vehicle (which acts as a leader vehicle). As the system of first-order differential equations for a platoon of vehicles is developed without considering the communication delay between vehicles or a delay parameter for driver perception–reaction time, all vehicles will react simultaneously. The position and speed profile of the leader vehicle and the initial position and speed of the follower vehicles are used as input parameters to solve the system of first-order ODEs for a platoon of CACC vehicles. The author simultaneously simulated three traffic states—(*a*) the uniform speed state, (*b*) the speed with constant acceleration state, and (*c*) the speed with constant deceleration state—to assess driver car-following behavior in a CACC platoon for different traffic states. These three scenarios can be simulated so that the vehicle dynamics and string stability of a CACC platoon can be assessed during the transition from one scenario to another.

5.3.1.2 Simulation of CACC Platoon

In this step, an extended driver car-following behavior model will be used to simulate *N* vehicles in a CACC platoon. Different traffic states (i.e., uniform speed, speed with constant acceleration, and speed with constant deceleration) need to be defined and simulated to evaluate the interactions between CACC vehicles. The speed profile of the CACC platoon leader needs to be adopted to define different traffic states. To initialize the simulation of a CACC platoon, the initial speed and the position of the leader as well as follower CACC vehicles are required. To determine platoon behavior in different traffic conditions, the simulation duration for each traffic state also needs to be defined. Simulation, using the selected driver car-following behavior models, will generate a vehicle position, speed, and acceleration profile for CACC platoon vehicles, and these profiles can be used to evaluate user acceptance in terms of vehicle dynamics and string stability.

5.3.1.3 *Evaluation of User Acceptance of Vehicle Dynamics and String Stability*

The evaluation of the vehicle dynamics and the string stability characteristics of CACC systems will justify the applicability of a specific driver car-following behavior model in CACC system design. Vehicle dynamics can be evaluated by calculating the maximum acceleration or deceleration boundary of a simulated CACC platoon in different traffic states and comparing with the comfortable acceleration or deceleration threshold: ±1 to ±2 ms^{-2} (Gunawan, 2012). Position, speed, and acceleration profiles are used to examine the string stability of a platoon (Lei et al., 2012). For example, the spacing error is used to measure fluctuation and is quantified as the two-norm of the spacing error (Sheikholeslam and Desoer, 1993). As the follower vehicles in a CACC platoon attempt to reach the leader vehicle’s speed, uncomfortable speed and acceleration will occur if the amplification of the follower’s speed and acceleration fluctuations is substantial compared with the amplification of the leader's speed and acceleration changes (Lei et al., 2012). To calculate the substantial fluctuation of speed and acceleration, the SSE ($\rho^2{}_{tw}$) was measured for each time window (*tw*) of the following vehicles in the CACC platoon with respect to the leader vehicle, as follows:

$$\rho_{tw}^2(t) = \sum_{n=1}^{N} \left(v_l\left(t\right) - v_n\left(t\right)\right)^2 tw(t)$$

where,

$tw(t) =$ time window between time t and $(t\,\text{-}1)$

$v_l(t) =$ speed of the leader vehicle at time t

$v_n(t) =$ speed of the n-th vehicle at time t

$n =$ index of the follower vehicles in a CACC platoon (1,2,3,...N), and

$\rho_{tw}^2\left(t\right) =$ SSE for tw at time t for the speed of all CACC follower vehicles in a platoon

This error will provide a measure of the comfort level of CACC occupants for a sudden speed or acceleration change in the follower vehicles. If the value of (ρ_{tw}^2) is zero, then all the follower vehicles are moving in the CACC platoon at the same speed, and there is no speed fluctuation. However, if the value of (ρ_{tw}^2) is greater than zero, there is a fluctuation in the speed or acceleration and the occupants' comfort criterion is compromised in terms of string stability. As the value of (ρ_{tw}^2) increases in correspondence with the follower vehicles' speed, the speed fluctuation increases, and driver discomfort also increases. To calculate the total SSE, the area under the SSE (ρ_{tw}^2) profiles of the following vehicles in the CACC platoon was measured with respect to the leader vehicle, as follows:

$$\text{Area under SSE} = \sum_{i=1}^{T} \rho_{tw}^{2}(t_i)$$

where i is the index of the time window $(1, 2, 3, ...T)$

5.4 Case Study: Evaluation of OVM and IDM

To select the driver car-following behavior models, the first step is to consider models that can represent different states of driver behavior, such as the uniform speed state, the speed with constant acceleration state, and the speed with constant deceleration state. Of all the available models, the OVM and the IDM have been used most in traffic flow studies because of their suitability to modeling diverse traffic states. Thus, the OVM and the IDM have been selected for an evaluation of their applicability to CACC system design through the evaluation framework developed in this research. The details of the evaluation of user acceptance in terms of vehicle dynamics and string stability are described next.

5.4.1 Driver Car-Following Behavior

Models: OVM and IDM In the following subsections, the mathematical formulation and the parameters of the OVM and the IDM are described in detail.

5.4.1.1 *OVM*

The OVM is a simple and realistic traffic flow model that considers traffic congestion dynamics. The OVM was first proposed by Newell, as follows (Newell, 1961; Bando et al., 1995; Bando et al., 1998; Mehmood and Easa, 2009):

$$\ddot{x}_n(t) = \frac{1}{\tau}\left[V\left(\Delta x_n(t)\right) - \dot{x}_n(t)\right] \qquad (1)$$

$$\Delta x_n(t) = x_{n-1}(t) - x_n(t)$$

where,

$n =$ Subject vehicle

n -1= Immediate leader vehicle

$\ddot{x}_n(t) =$ Acceleration of the *n*-th vehicle at time *t*

$\dot{x}_n(t) =$ Speed of the *n*-th vehicle at time *t*

τ= Speed relaxation time

$\Delta x_n(t) =$ Space headway between subject vehicle and leading vehicle

$x_{n-1}(t) =$ Position of the (*n*-1)-th vehicle at time *t*
$x_n(t) =$ Position of the *n*-th vehicle at time *t*

$V\left(\Delta x_n(t)\right) =$ Optimum velocity function of the *n*-th vehicle at time *t*

Later, Bando et al. introduced a delay parameter (t_d) to consider reaction time of a driver and revised Eq. (1) as follows (Bando et al., 1995):

$$\tau\ddot{x}_n(t) + \dot{x}_n(t) = V\left(\Delta x_n(t - t_d)\right) \qquad (2)$$

The optimum velocity (*V*) is the function of space headway, which is the distance from the bumper of the subject vehicle to the bumper of the vehicle immediately in front of the subject vehicle. According to the OVM, the space headway will be adjusted in such a way that collision is avoided between vehicles. For example, the subject vehicle's speed increases if the space headway increases. However, the subject vehicle's speed cannot exceed the maximum acceptable or comfortable speed, which is defined by the optimum velocity function.

Hence, the properties of the optimum velocity function are as follows: (*a*) it increases monotonically and (*b*) $\left|V\left(\Delta x_n\right)\right|$ has an upper bound. Many previous studies used a different optimum velocity function that showed these optimum velocity function properties (Bando et al., 1995; Bando et al., 1998; Mehmood and Easa, 2009; Tadaki et al., 1998). However, in this research the author use an optimal velocity function that considers the desired velocity of the subject vehicle and the braking distance, as proposed by Davis as follows (*Davis, 2002*):

$$V\left(\Delta x_n\left(t\right)\right)=v_0\left\lfloor \tanh\left(\frac{\left(\Delta x-D\right)}{b}-C_1\right)+C_2\right\rfloor \qquad (3)$$

where,

v_0 = Desired speed of the subject vehicle

D = Effective length of the subject vehicle

b = Required distance for braking

C_1 = Length constant

C_2 = Dimensionless constant

The author used OVM parameter values estimated in (Gunawan, 2012), which are:- $\dot{x}_0$ = 17.00 m/s; *D*=7.00 m; *b*=4.00 m; C_1=1.72 m; C_2=0.91; and τ=0.50s. The author used OVM and IDM parameter values estimated in (Gunawan, 2012) to simulate driver car-following behavior of each vehicle in the CACC platoon instead of calibrating driver car-following behavior model parameters. As the IDM and OVM model parameters, which were presented in (Gunawan, 2012), were estimated for urban driving scenario in a real-world roadway segment, the values of these model parameters can be used to simulate urban car-following behavior for the CACC vehicles to represent real-world effects. However, the driver car-following behavior changes with changing the traffic condition, it is required to calibrate driver car-following model parameters to represent realistic behavior of CACC vehicles based on the traffic condition.

5.4.1.2 <u>IDM</u>

The simplified version of the acceleration of a subject vehicle using IDM is modeled by the following equation (Triber and Helbing, 2000; Treiber et al., 2000):

$$\ddot{x}_n = a\left[1-\left(\frac{\dot{x}_n}{\dot{x}_0}\right)^{\delta}-\left(\frac{s^*\left(\dot{x}_n,\Delta\dot{x}_n\right)}{s_n}\right)^2\right] \qquad (4)$$

where,

a = Normal acceleration

$\dot{x}_n =$ Speed of the *n*-th vehicle at time *t*

$\dot{x}_0 =$ Desired speed when driving on a road

$s^*\left(\dot{x}_n, \Delta\dot{x}_n\right) =$ Desired minimum distance between *n*-th and ($n-1$)-th vehicle

$s_n =$ The distance between *n*-th and ($n-1$)-th vehicle

δ = Exponent for the vehicle's acceleration

The acceleration of a vehicle is divided into two parts: i) “desired” acceleration - when the subject vehicle moves with the free flow speed, and ii) braking deceleration – when the subject vehicle press brake because of the vehicle’s deceleration immediately in front of subject vehicle. “Desired” acceleration is $1-\left(\frac{\dot{x}_n}{\dot{x}_0}\right)^{\delta}$ and braking deceleration of the vehicle immediately in front of subject vehicle is $\left(\frac{s^*\left(\dot{x}_n, \Delta\dot{x}_n\right)}{s_n}\right)^2$. The subject vehicle is approaching the immediate front vehicle with “desired speed”, $\dot{x}_n$. If (*n*-1)-th vehicle length is l_{n-1}, the gap between the *n*-th and *(n-1)*-th vehicle is as follows:

$$s_n := x_{n-1} - x_n - l_{n-1}$$

For simplicity, the author assumed that the length (l_c) of all the vehicles is the same. The relative speed of the subject vehicle with respect to the vehicle immediately in front of subject vehicle is as follows:

$$\Delta\dot{x}_n := \dot{x}_n - \dot{x}_{n-1}$$

The desired gap s^* is defined as follows:

$$s^*\left(\dot{x}_n, \Delta\dot{x}_n\right) = s_0 + \max\left[0, \left(\dot{x}_n T + \frac{\dot{x}_n \Delta\dot{x}_n}{2\sqrt{ab}}\right)\right] \qquad (5)$$

where,

T= Time headway to safely follow the immediate leader vehicle

b = Normal comfortable braking deceleration

s_0 = Minimum space headway between subject vehicle and the vehicle immediately in front of the subject vehicle

As $\Delta\dot{x}_n$ is a non-zero term, the term $\frac{\dot{x}_n \Delta\dot{x}_n}{2\sqrt{ab}}$ is only active when the subject vehicle is moving in a traffic stream. The author used the IDM model parameter values estimated in (Gunawan, 2012) for the evaluation of the performance of the IDM model for a CACC controller design in this study that includes $\dot{x}_0$ =22.0 m/s, s_0 =2.0 m, T=0.5 s, b= 1.7 m/s^2, and δ *=4; l_c*=5.0m.

5.4.2 Transformation of Driver Car-Following Behavior Models from Single Vehicle to N Vehicles

5.4.2.1 *OVM*

OVM represents the dynamics of the position and speed of a single vehicle, which can be transformed into systems of differential equations to describe the dynamics of a CACC platoon. Second-order ODEs for OVM for N number of vehicles can be presented as follows:

$$\begin{aligned}
&\ddot{x}_1(t) = k\left[V\left(\left(x_0(t) - x_1(t)\right)\right) - \dot{x}_1(t)\right] \\
&\ddot{x}_2(t) = k\left[V\left(\left(x_1(t) - x_2(t)\right)\right) - \dot{x}_2(t)\right] \\
&\ddot{x}_3(t) = k\left[V\left(\left(x_2(t) - x_3(t)\right)\right) - \dot{x}_3(t)\right] \\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\
&\ddot{x}_N(t) = k\left[V\left(\left(x_{n-1}(t) - x_n(t)\right)\right) - \dot{x}_n(t)\right]
\end{aligned}$$

where,

$$k = \frac{1}{\tau}$$

$$\Delta x_n(t) = x_{n-1}(t) - x_n(t)$$

$$V\left(\Delta x_n(t)\right) = v_0\left[\tanh\left(\frac{(\Delta x - D)}{b} - C_1\right) + C_2\right]$$

That is,

$$\ddot{\boldsymbol{x}} = f\left(\dot{\boldsymbol{x}}, \boldsymbol{x}, \boldsymbol{p}\right) \qquad (6)$$

where, $\ddot{\boldsymbol{x}} = \left(\ddot{x}_1, \ddot{x}_2, \ddot{x}_3, ..., \ddot{x}_N\right)^T$, $\dot{\boldsymbol{x}} = \left(\dot{x}_1, \dot{x}_2, \dot{x}_3, ..., \dot{x}_n\right)^T$, $\boldsymbol{x} = \left(x_1, x_2, x_3, ..., x_N\right)^T$ and $\boldsymbol{p}$ are the vector of accelerations, the vector of velocities, the vector of positions, and the OVM parameter vector, respectively. Eq. (6) is a system of second-order ODEs for a platoon of *N* number of CACC vehicles so that all the vehicles can move simultaneously using the driver car-following behavior model. System of second order ordinary differential equations do not have exact solution and a numerical method is required to solve the system of equations *(34)*. As the second order ODE's increase the complexity in solving the system of equations using a numerical method and require high computational resource, the author converted the second order system of ODEs to first order system of ODEs. To convert the second-order ODEs to a system of first-order ODEs, the author assumes,

$$\begin{aligned}
&y_i = x_i,\ i = 1,2,3,...,N \\
&y_{i+N} = \dot{x}_i,\ i = 1,2,3,...,N \\
&\dot{y}_{i+N} = \ddot{x}_i,\ i = 1,2,3,...,N
\end{aligned}$$

Thus, the system of first-order equations of N number of vehicles in a CACC platoon becomes,

$$\begin{aligned}
&\dot{y}_1 = y_{N+1} \\
&\dot{y}_2 = y_{N+2} \\
&\dot{y}_3 = y_{N+3} \\
&.......... \\
&\dot{y}_N = y_{2N} \\
&\dot{y}_{N+1} = k\left[V\left(\left(x_0(t) - y_1(t)\right)\right) - y_{N+1}(t)\right] \\
&\dot{y}_{N+2} = k\left[V\left(\left(y_1(t) - y_2(t)\right)\right) - y_{N+2}(t)\right] \\
&.. \\
&\dot{y}_{2N} = k\left[V\left(\left(y_{N-1}(t) - y_N(t)\right)\right) - y_{2N}(t)\right]
\end{aligned}$$

The system of first-order equations is of the form, $\dot{\boldsymbol{y}} = g(\boldsymbol{y}, \boldsymbol{p})$, where the combined position and speed vector of the platoon of CACC vehicles is $\boldsymbol{y} = \left(y_1, y_2, y_3, ..., y_N, y_{N+1}, y_{N+2}, y_{N+3}, ..., y_{2N}\right)^T = \left(x_1, x_2, x_3, ..., x_N, \dot{x}_1, \dot{x}_2, \dot{x}_3, ..., \dot{x}_N\right)^T$. The first *N* number of elements of this vector represent CACC vehicle position, and the second

N number of elements represents speed of the CACC vehicles. The OVM parameter vector is $\boldsymbol{p}$.

5.4.2.2 *IDM*

In the IDM model, subject vehicle acceleration depends on the dynamics of the position and speed of the leader vehicle. The following formulations represent second-order ODEs for a platoon of *N* vehicles using IDM:

$$\ddot{x}_1(t) = a\left[1 - \left(\frac{\dot{x}_1}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*(\dot{x}_1, \Delta\dot{x}_1)}{s_1}\right)^2\right]$$

$$\ddot{x}_2(t) = a\left[1 - \left(\frac{\dot{x}_2}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*(\dot{x}_2, \Delta\dot{x}_2)}{s_2}\right)^2\right]$$

$$\ddot{x}_3(t) = a\left[1 - \left(\frac{\dot{x}_3}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*(\dot{x}_3, \Delta\dot{x}_3)}{s_3}\right)^2\right]$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$\ddot{x}_N(t) = a\left[1 - \left(\frac{\dot{x}_N}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*(\dot{x}_N, \Delta\dot{x}_N)}{s_N}\right)^2\right]$$

where,

$$s^*(\dot{x}_N, \Delta\dot{x}_N) = s_0 + \max\left[0, \left(\dot{x}_N T + \frac{\dot{x}_N \Delta\dot{x}_N}{2\sqrt{ab}}\right)\right] \tag{7}$$

$$\Delta x_N(t) := x_{N-1}(t) - x_N(t)$$

$$s_N(t) := x_{N-1}(t) - x_N(t) - l_c$$

That is,

$$\ddot{\boldsymbol{x}} = f(\dot{\boldsymbol{x}}, \boldsymbol{x}, \boldsymbol{p}) \tag{8}$$

where, $\ddot{\boldsymbol{x}} = \left(\ddot{x}_1, \ddot{x}_2, \ddot{x}_3, ..., \ddot{x}_N\right)^T$, $\dot{\boldsymbol{x}} = \left(\dot{x}_1, \dot{x}_2, \dot{x}_3, ..., \dot{x}_n\right)^T$, $\boldsymbol{x} = \left(x_1, x_2, x_3, ..., x_N\right)^T$ and $\boldsymbol{p}$ are the vector of accelerations, the vector of velocities, the vector of positions, and the IDM parameter vector, respectively. Eq. (8) is a system of second-order ODEs. A system of second-order ODEs is required for transformation into a system of first-order ODEs. Thus, the system of first-order equations using IDM can be represented as follows:

$$\begin{aligned}
&\dot{y}_1 = y_{N+1} \\
&\dot{y}_2 = y_{N+2} \\
&\dot{y}_3 = y_{N+3} \\
&.......... \\
&\dot{y}_N = y_{2N}
\end{aligned}$$

$$\begin{aligned}
&\dot{y}_{N+1} = a\left[1-\left(\frac{y_{N+1}}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*\left(y_{N+1}, \Delta y_{N+1}\right)}{s_1}\right)^2\right] \\
&\dot{y}_{N+2} = a\left[1-\left(\frac{y_{N+2}}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*\left(y_{N+2}, \Delta y_{N+2}\right)}{s_2}\right)^2\right] \\
&.. \\
&\dot{y}_{2N} = a\left[1-\left(\frac{y_{2N}}{\dot{x}_0}\right)^{\delta} - \left(\frac{s^*\left(y_{2N}, \Delta y_{2N}\right)}{s_N}\right)^2\right]
\end{aligned}$$

The system of first-order equations is of the form, $\dot{\boldsymbol{y}} = g\left(\boldsymbol{y}, \boldsymbol{p}\right)$, where $\boldsymbol{y}$ is the combined position and speed vectors of a platoon of vehicles as explained for the OVM model in the previous sub-section 5.4.2.1.

5.4.3 Simulation of a CACC Platoon of N Number of Vehicles

The author coded the system of ODEs for a CACC platoon in MATLAB. Total number of vehicles for forming a CACC platoon, total simulation time, initial position and speed of the follower vehicles, position and speed profile of the platoon leader and car-following model parameter values are the input parameters in the simulation. Table 5-I summarizes input parameters and simulation requirement. Parameter values for both driver car-following behavior models are adopted from the literature (Gunawan, 2012).

The author simulated three different traffic states: i) uniform speed state, ii) speed with constant acceleration state, and iii) speed with constant deceleration state and three different simulation scenarios: 3600s, 240s and 120s based on different acceleration and deceleration rates to assess the driver car-following behavior for CACC vehicles. In the simulation, the number of follower vehicles of vehicle was five. The initial position of the leader vehicle was 100 m from the origin, and five follower vehicles were 0m, 20 m, 40 m, 60 m and 80 m from the leader vehicle. The initial speed of all follower vehicles was 15 m/s, which was same as the leader vehicle's speed. The speed of all the CACC vehicles varies between 15m/s (~35mph) to 25m/s (~ 55 mph), which represents the traffic flow characteristics of the urban roadway (Gunawan, 2012). The author divides the total simulation durations into four periods: i) uniform speed state (15 m/s), ii) constant acceleration state (speed changes from 15 m/s to 25 m/s), iii) uniform speed state (25 m/s), and iv) constant deceleration state (speed changes from 25 m/s to 15 m/s).

The author solved the system of ODEs using the ode45 solver. However, the author tested three non-stiff (ode45, ode23 and ode113) and three stiff (ode15s, ode23s and ode23t) ODE solvers to find the reliable and efficient solver for the system of ODEs. ode45 solver provided the most reliable result, based on the area under the curve of SSE (i.e., SSE for IDM=1167 and OVM=975) for the speed profiles of follower vehicles with respect to the leader vehicle without considering any communication delays than other solvers.

Table 5-I Simulation Scenarios

Input Parameters	**Simulation Requirement**
Total Vehicle number	One leader and five follower vehicles
Initial position of the leader vehicle	100 m from the origin
Initial position of follower Vehicles (1, 2, 3, 4, and 5)	0 m, 20 m, 40 m, 60 m, and 80 m respectively from the origin
Initial speed of the leader vehicle	15 m/s
Initial speed of follower Vehicles' (1, 2, 3, 4, and 5)	15 m/s for all follower vehicles
Simulation scenarios	3600 s, 240 s, and 120 s
Traffic states defined by the leader vehicle	• **Uniform speed state** (15 m/s and 25 m/s) • **Speed with constant acceleration state** (Speed changes from 15 m/s to 25 m/s) • **Speed with constant deceleration state** (Speed changes from 25 m/s to 15 m/s)

5.4.4 Analysis of Simulation Results

5.4.4.1 *Evaluation of Car-Following Model without Communication Delay*

As described in the previous section, the author simulated a CACC platoon with six vehicles to assess the driver's car-following behavior with different acceleration and deceleration rates. Platoon vehicles' position, speed, and acceleration profiles were developed using MATLAB simulation outputs to evaluate the follower vehicles' occupants comfort level in terms of vehicle dynamics and string stability. Figure 5.1(a) presents the position profile of a leader and five follower vehicles using OVM and IDM for different traffic states for 120s simulation scenario. It is evident that the position profile of following vehicles maintains a uniform distance between vehicles. As none of the two following vehicle's position profile overlapped, there is no collision risk between vehicles using OVM and IDM. Similar behavior of position profiles for simulation scenarios 240s and 3600s was observed. Figures 5.1(b), 5.1(c), and 5.1(d) represent speed profiles of leader and following vehicles in a CACC platoon for 3600s, 240s and 120s simulation scenarios, respectively, using OVM and IDM. The acceleration/deceleration rates followed by the leader were ±0.01 m/s^2, ±0.17 m/s^2, and ±0.33 m/s^2 for the 3600s, 240s and 120s simulation scenarios, respectively. Simulation scenario 1 (3600s) as illustrated in Figure 5.1(b) shows that the CACC platoon was formed in first 20s, where the vehicles were 20m away from each other at the beginning of simulation. After 20s, all follower vehicles follow the leader vehicle closely in a CACC platoon. Same space-headway between vehicles was maintained (as shown in Figure 5.1(a)), and the follower vehicles adjusted the speed to maintain comfortable vehicle dynamics for the occupants in the vehicle. The detail findings from the simulation analysis are presented in the following three sub-sections. Analysis of user acceptance in terms of vehicle dynamics and string stability of the CACC platoon are described in the sub-sections 5.4.1.1 and 5.4.1.2, respectively. Summary evaluation matrix of the user acceptance for the driver car-following models has been presented in sub-section 5.4.1.3.

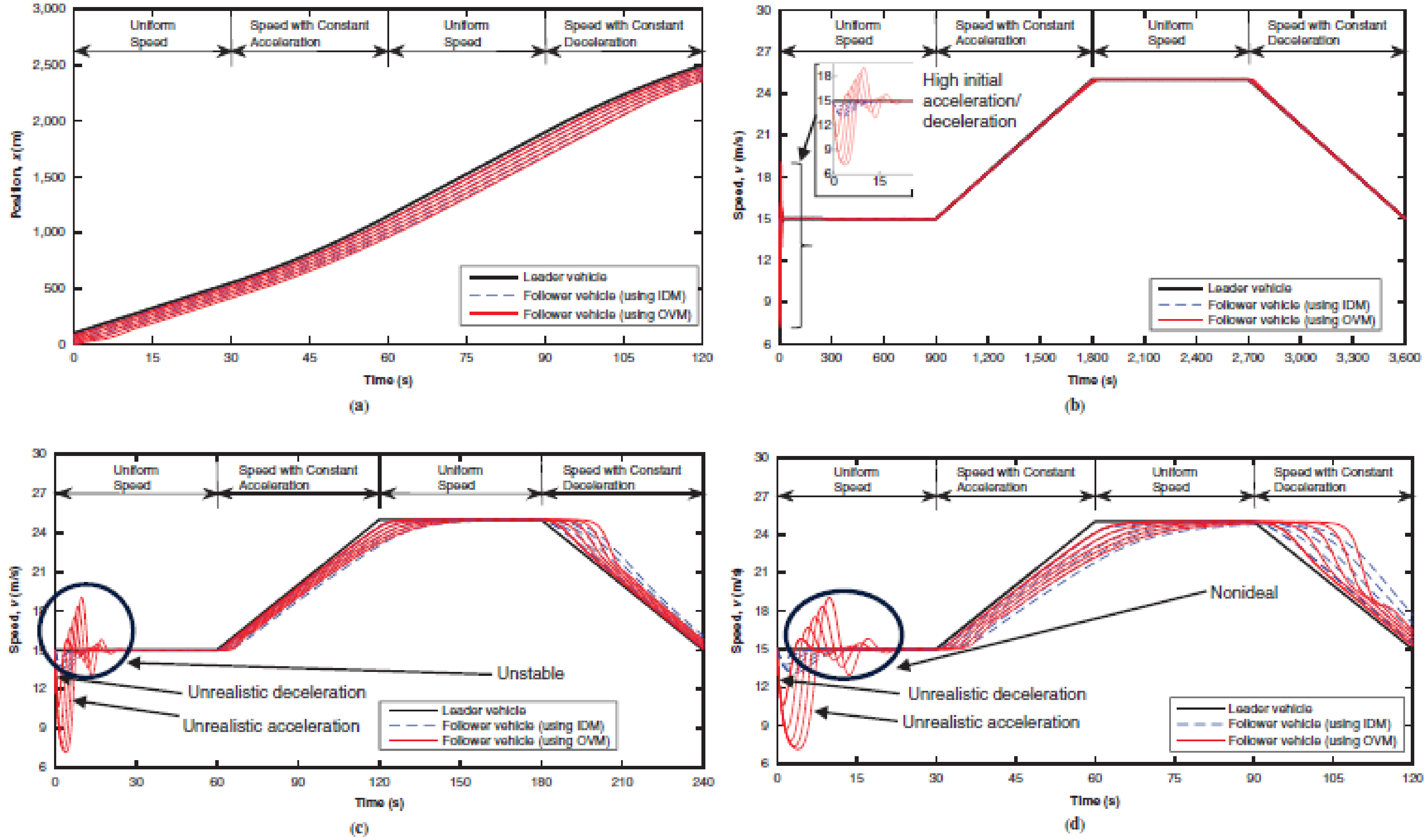


Fig. 5.1. For platoon of six vehicles under OVM and IDM: (a) position profile for 120-s simulation scenario and speed profiles for (b) 3,600-s, (c) 240-s, and (d) 120-s simulation scenarios

5.4.4.1.1 Analysis of User Acceptance in terms of Vehicle Dynamics

Figures 5.2(a) and 5.2(b) shows the acceleration/deceleration profiles of the follower vehicles in a CACC platoon for the 120s simulation scenario using the OVM and IDM models, respectively. It is observed that the maximum acceleration of five following vehicles in the platoon forming state (i.e., 20 s) exceeds the maximum comfortable acceleration/deceleration threshold, ±2 m/s2 for regular traffic using OVM (Gunawan, 2012), shown in Figure 5.2(a). In contrary, Figure 5.2(b) shows that using the IDM model the observed maximum acceleration/deceleration is less than the maximum comfortable/acceptable threshold in all traffic states for all CACC vehicles. As the optimum velocity of the OVM model is only vary with respect to space headway and does not consider relative speed between the subject vehicle and the vehicle immediately in front of the subject vehicle, it has a significant effect on the following vehicle's acceleration (Treiber and Kesting, 2013). In contrast, IDM considers relative speed, space-headway and desired maximum acceleration/deceleration constraints in the acceleration calculation. However, both IDM and OVM model show comfortable acceleration and deceleration in terms of vehicle dynamics after the first 20 s of the simulation period.

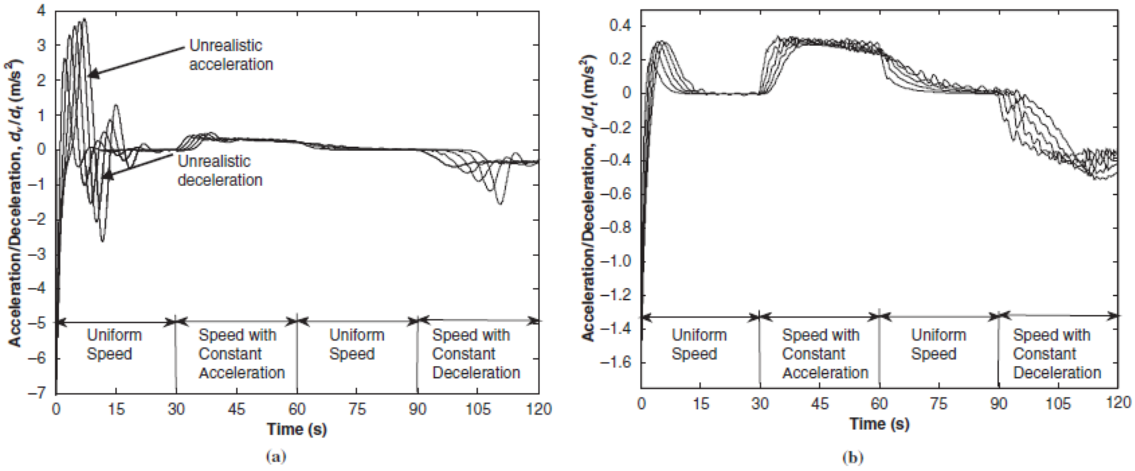


Fig. 5.2. Acceleration and deceleration profiles of five follower vehicles under (a) OVM and (b) IDM for 120-s simulation scenario (d = derivative; v = velocity; t = time).

5.4.4.1.2 Analysis of User Acceptance in terms of String Stability

Figures 5.3(a), 5.3(b), and 5.3(c) represent SSE profiles in terms of speed of the CACC follower vehicles using OVM and IDM for simulation scenarios 3600s, 240s and 120s, respectively. The time window (tw) of 0.001s used in calculating these error profiles. Speed profiles show higher SSE value for the first 20s of the simulation period using the OVM compared to the IDM for all simulation scenarios, but the SSE for speed was zero after first 20s only in 3600s simulation scenario. The author did not consider first 20s of the simulation to calculate area under the SSE of the speed profile. The first 20s considered as a platoon forming state. Thus, there was no speed fluctuation and the follower vehicles followed the same speed as the leader vehicle in acceleration and deceleration states, and vehicle occupants were comfortable in 3600s simulation scenario.

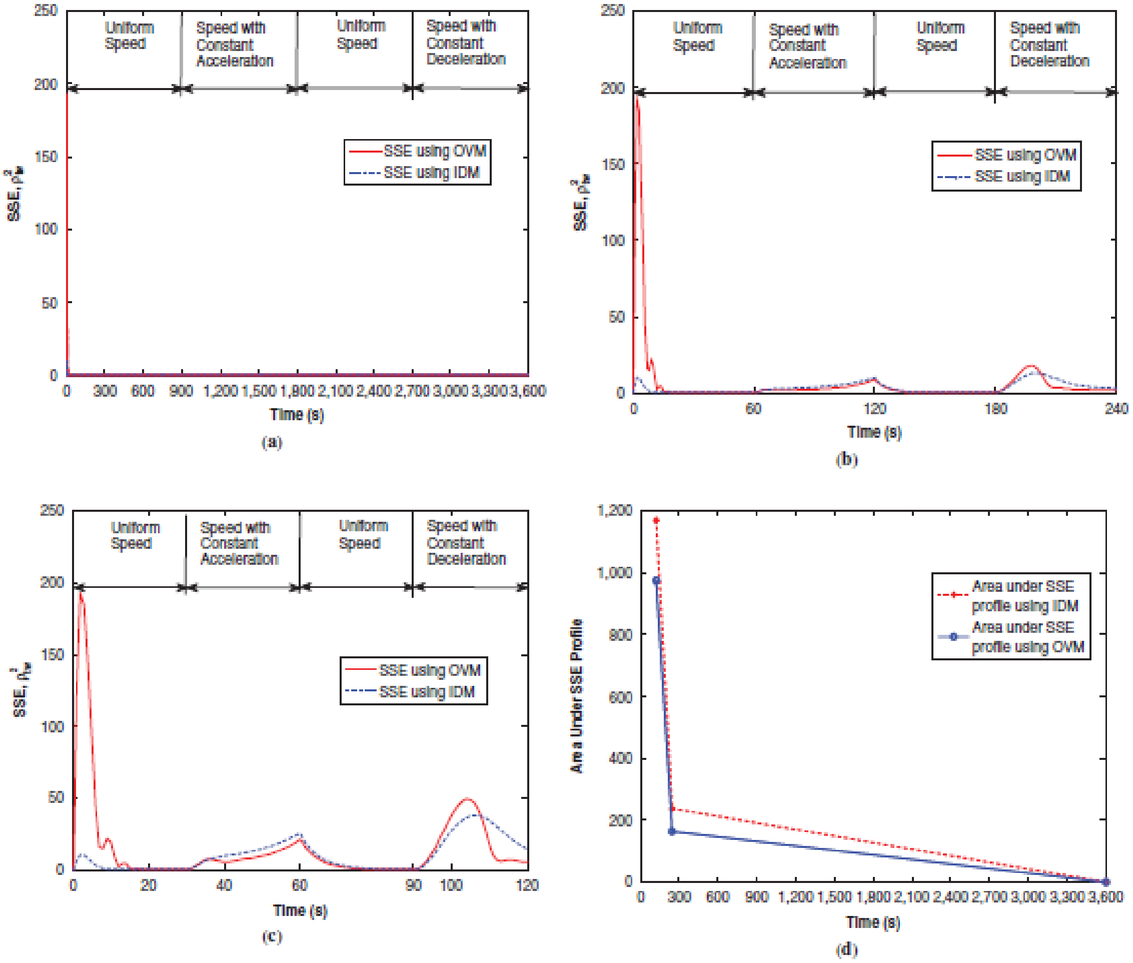


Fig. 5.3. SSE profile of follower vehicle speed with respect to leader vehicle in terms of speed fluctuation for (a) 3,600-s, (b) 240-s, and (c) 120-s simulation scenarios and (d) comparison of area under curve of SSE profile, excluding first 20 s of simulation.

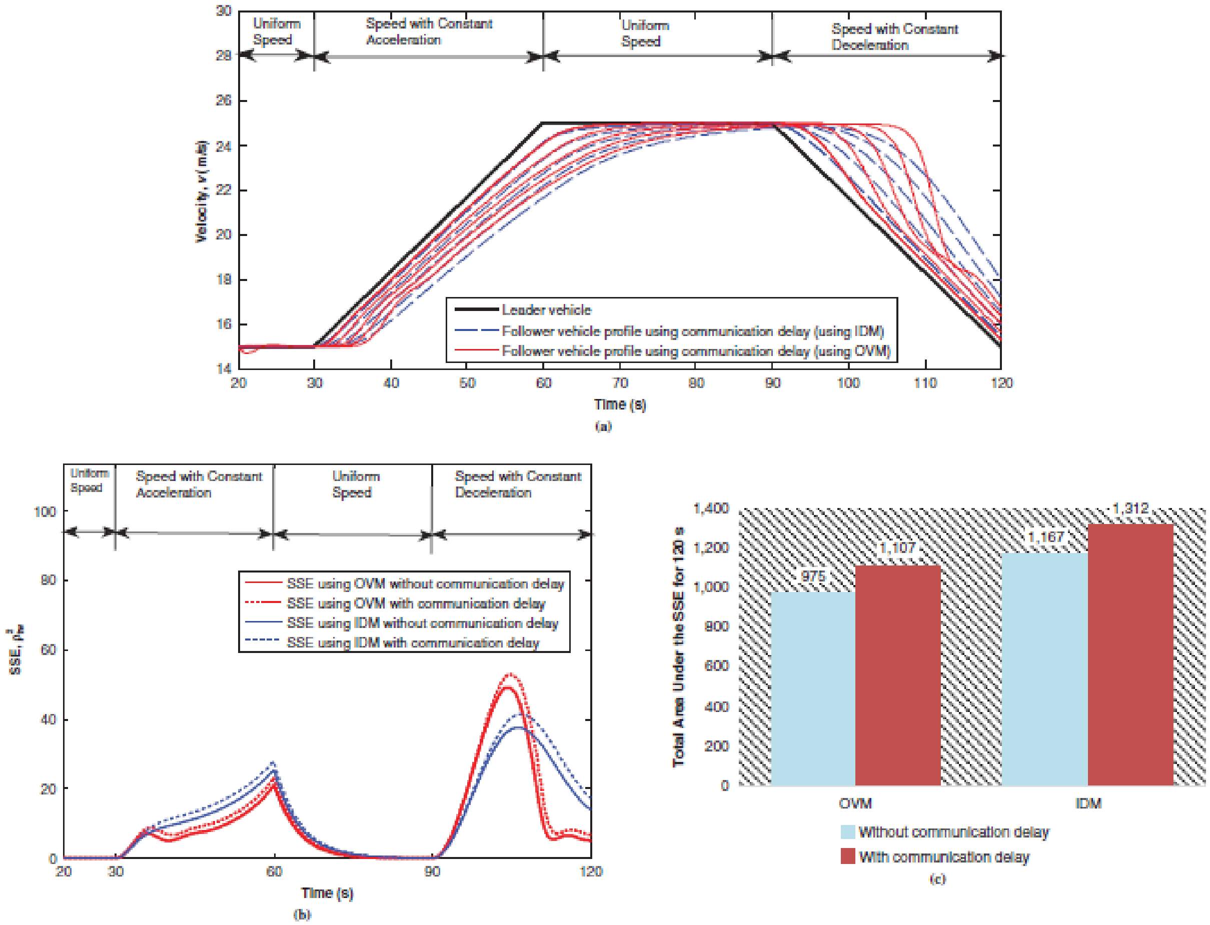
5.4.4.1.3 Analysis Summary of the CACC Controller design using Car-following Model

A decision matrix is developed summarizing the key observations from the analysis of speed profiles and acceleration/deceleration profiles of the following vehicles and presented in Table 5-II. As discussed in sections 5.4.4.1.1 and 5.4.4.1.2, the decision matrix reveals that if the CACC controller of the follower vehicles use IDM, vehicle occupants will be comfortable in all traffic states in terms of string stability and vehicle dynamics. On the other hand, occupants in the platoon vehicles will be uncomfortable in the CACC platoon forming states, if CACC controller of the follower vehicles is designed using OVM.

Table 5-II Comparison of OVM and IDM for 120s Simulation Scenario

Traffic State	**OVM Maximum Acceleration/ Deceleration (m/s^2)**	**IDM Maximum Acceleration/ Deceleration (m/s^2)**	**Maximum Acceleration/ Deceleration Threshold (m/s^2) *(12)***	**Decision Matrix**			
				In terms of Vehicle Dynamics		**In terms of String Stability**	
				OVM	**IDM**	**OVM**	**IDM**
Uniform speed state (15 m/s)	-6.65	-1.49	± (1 to 2)	UC	C	UC	C
Constant acceleration state (Speed changes from 15 m/s to 25 m/s)	+0.49	+0.28	± (1 to 2)	C	C	C	C
Constant deceleration state (Speed changes from 25 m/s to 15 m/s)	-1.57	-0.51	± (1 to 2)	C	C	C	C

Note: Comfortable (C), Uncomfortable (UC)



(*a*) Speed profile for a platoon of six vehicles under OVM and IDM from 20 to 120 s in 120-s simulation scenario with communication delay, (*b*) comparison of SSE profile of follower vehicle speed from 20 to 120 s with and without communication delay in 120-s simulation scenario, and (*c*) comparison of area under curve of SSE profile.

Fig. 5.4. Simulation results with communication delay.

5.4.4.2 Evaluation of Car-Following Model with Communication Delay

The author considered communication delay in transferring vehicle position, speed, and acceleration information between vehicles for simulating realistic CACC scenario. As the author simulated urban driving scenarios, 125ms communication delay between vehicles was used for Dedicated Short Range Communication (DSRC) at 50 mph (Dey et al., 2016). Thus, each following vehicle in the CACC platoon receives vehicle trajectory information (i.e., position, speed, acceleration) from the immediate front vehicle with 125ms delay. In MATLAB, driver car-following behavior model was used this delayed immediate front vehicle trajectory information to calculate the subject vehicle's position, speed and acceleration. Figures 5.4(a) represents speed profiles of leader and following vehicles in a CACC platoon for 120s simulation scenario using OVM and IDM with communication delay. As illustrated in Figure 5.4(a), the author found that the first 20s is required to form the CACC platoon (also shown in Figure 5.1(d)). After the first 20s of the simulation period, all follower vehicles closely follow each other and form a CACC platoon. Figures 5.4(b) represents SSE (ρ_{tw}^2) profiles in terms of speed of the CACC follower vehicles using OVM and IDM for simulation scenarios 120s with and without communication delay. The time window (*tw*) of 0.001s is used for calculating these error profiles. In Figure 5.4(c), the author compared the total area under the curve of SSE profiles with and without communication delay for IDM and OVM speed profiles. It shows that IDM has higher area under the curve of SSE profiles with and without communication delay than the OVM model. It reveals that OVM is more string stable with and without communication delay than IDM. The analysis of user acceptance in terms of vehicle dynamics with communication delay shows similar vehicle dynamics (i.e., comfortable acceleration and deceleration) without communication delay.

5.5 Summary and Conclusions

The primary focus of this study was to develop a framework for the evaluation of the user acceptance of different driver car-following behavior models for a CACC controller, based on their performance in maintaining acceptable vehicle dynamics and string stability. Two most widely used car-following models, OVM and IDM, were considered for the CACC controller design and evaluated using the developed evaluation framework in this research. In order to evaluate these models, a system of first-order ODEs for OVM and IDM were derived for a platoon of vehicles and were extended from a single vehicle to N number of vehicles using a system of ODEs. A platoon of six vehicles was then simulated for different traffic flow scenarios (i.e., uniform speed, speed with constant acceleration, and speed with constant deceleration). Analysis of position, speed and acceleration profiles of each vehicle in the platoon were performed to determine vehicle dynamics and string stability that measure driver comfort as well as the associated collision risks between vehicles. It was found that OVM is more suitable for a CACC controller design due to its capability to incorporate different traffic states while providing acceptable vehicle dynamics and string stability compared to IDM for a given set of model parameters, with both with and without communication delays between platooned CACC vehicles. It is important to note that OVM and IDM model parameters can change if the traffic condition changes. Vehicle dynamics and string stability of a CACC platoon in a connected vehicle environment also depends on incorporation of driving behaviors or factors, and design of the autonomous control.

5.6 Acknowledgment

The author would like to thank Aniqa Chowdhury for editing the draft.

5.7 References

Ames, W. F. *Numerical Methods for Partial Differential Equations,* 3rd ed. Academic Press, Cambridge, Mass., 2014.

Basak, K., S. N. Hetu, Z. Li, C. Lima, H. Azevedo, T. Loganathan, T. Toledo, R. Xu, Y. Xu, L. Peh, & M. Ben-Akiva (2013). "Modeling Reaction Time Within a Traffic Simulation Model," In *Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems,* IEEE, New York, pp. 302–309.

Bando, M., K. Hasebe, A. Nakayama, A. Shibata, & Y. Sugiyama (1995). "Dynamical Model of Traffic Congestion and Numerical Simulation," *Physical Review E,* Vol. 51, No. 2, pp. 1035–1042. https://doi.org/10.1103/PhysRevE.51.1035.

Bando, M., K. Hasebe, K. Nakanishi, & A. Nakayama (1998). "Analysis of Optimal Velocity Model with Explicit Delay," *Physical Review E,* Vol. 58, No. 5, pp. 5429–5435. https://doi.org/10.1103/PhysRevE.58.5429.

Bose, A., & P. Ioannou. *Analysis of Traffic Flow with Mixed, Manual and Intelligent Cruise Control Vehicles: Theory and Experiments.* California Partners for Advanced Transit and Highways, University of California, Berkeley, 2001.

Brackstone, M., & M. McDonald (1999). "Car-Following: A Historical Review," *Transportation Research Part F: Traffic Psychology and Behaviour,* Vol. 2, No. 4, 1999, pp. 181–196. https://doi.org/10.1016/S1369-8478(00)00005-X.

Burnham, G. O., J. Seo, & G. A. Bekey (1974). "Identification of Human Driver Models in Car-Following," *IEEE Transactions on Automatic Control,* Vol. 19, No. 6, 1974, pp. 911–915. https://doi.org/10.1109/TAC.1974.1100740.

CVRIA. Connected Vehicle Reference Implementation Architecture. http://www .iteris.com/cvria/html/applications/applications.html. Accessed July 30, 2016.

Davis, L. C (2002). "Comment on Analysis of Optimal Velocity Model with Explicit Delay," *Physical Review E,* Vol. 66, No. 3, p. 038101.https://doi.org/10.1103/PhysRevE.66.038101.

Dey, K. C., L. Yan, X. Wang, Y. Wang, H. Shen, M. Chowdhury, L. Yu, C. Qiu, & V. Soundararaj (2016). "A Review of Communication, Driver Characteristics, and Controls Aspects of Cooperative Adaptive Cruise Control (CACC)," *IEEE Transactions on Intelligent Transportation Systems,* Vol. 17, No. 2, 2016, pp. 491–509. https://doi.org/10.1109/TITS.2015.2483063.

Dey, K. C., A. Rayamajhi, M. Chowdhury, P. Bhavsar, J. Martin, A. Rayamajhi, M. Chowdhury, P. Bhavsar, & J. Martin (2016). "Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) Communication in a Heterogeneous Wireless Network Performance Evaluation," *Transportation Research Part C: Emerging Technologies,* Vol. 68, pp. 168–184.https://doi.org/10.1016/j.trc.2016.03.008.

Fritzsche, H (1994). "A Model for Traffic Simulation," *Traffic Engineering and Control,* Vol. 35, No. 5, pp. 317–321.

Gunawan, F. E (2012). "Two-Vehicle Dynamics of the Car-Following Models on Realistic Driving Condition," *Journal of Transportation Systems Engineering and Information Technology,* Vol. 12, No. 2, pp. 67–75. https://doi.org/10.1016/S1570-6672(11)60194-3.

Jones, S (2016). *Cooperative Adaptive Cruise Control: Human Factors Analysis.* Publication FHWA-HRT-13-045. FHWA, U.S. Department of Transportation, McLean, Va. https://www.fhwa.dot.gov/publications/research/safety/13045/13045.pdf. Accessed July 30, 2016.

Krauss, S (1997). "Towards a Unified View of Microscopic Traffic Flow Theories," In *Proceedings of the IFAC Transportation System Conference,* International Federation of Automatic Control, Laxenburg, Austria, pp. 941–945.

Lei, C., E. M. van Eenennaam, W. Klein Wolterink, J. Ploeg, G. Karagiannis, & G. Heijenk (2012). "Evaluation of CACC String Stability using SUMO, Simulink, and OMNeT," *EURASIP Journal on Wireless Communications and Networking,* Vol. 2012, No. 1, p. 116. https://doi.org/10.1186/1687-1499-2012-116.

Ma, Y., M. Chowdhury, A. Sadek, & M. Jeihani (2012). "Integrated Traffic and Communication Performance Evaluation of an Intelligent Vehicle Infrastructure Integration (VII) System for Online Travel-Time Prediction," *IEEE Transactions on Intelligent Transportation Systems,* Vol. 13, No. 3, pp. 1369–1382. https://doi.org/10.1109/TITS.2012.2198644.

Ma, Y., M. Chowdhury, A. Sadek, & M. Jeihani (2009). “Real-Time Highway Traffic Condition Assessment Framework Using Vehicle Infrastructure Integration (VII) with Artificial Intelligence (AI),”. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 10, No. 4, pp. 615–627.

Mehmood, A., & S. M. Easa (2009). “Modeling Reaction Time in Car-Following Behaviour Based on Human Factors,” *International Journal of Engineering and Applied Science,* Vol. 5, No. 2, pp. 93–101.

Milanés, V., & S. E. Shladover (2014). “Modeling Cooperative and Autonomous Adaptive Cruise Control Dynamic Responses Using Experimental Data,” *Transportation Research Part C: Emerging Technologies,* Vol. 48, pp. 285–300. https://doi.org/10.1016/j.trc.2014.09.001.

Newell, G. F (1961). “Nonlinear Effects in the Dynamics of Car-Following,” *Operations Research,* Vol. 9, No. 2, pp. 209–229. https://doi.org /10.1287/opre.9.2.209.

Pueboobpaphan, R., & B. van Arem (2010). “Driver and Vehicle Characteristics and Platoon and Traffic Flow Stability: Understanding the Relationship for Design and Assessment of Cooperative Adaptive Cruise Control,” *Transportation Research Record: Journal of the Transportation Research Board,* No. 2189, pp. 89–97. https://doi.org/10.3141/2189-10.

Panwai, S., & H. F. Dia (2004). “Comparative Evaluation of Car Following Models. Presented at Workshop on Traffic Simulation,” *Bridging Theory and Practice*, Brisbane, Australia.

PTV VISSIM 5.40-03 User Manual. Planung Transport Verkehr, Karlsruhe, Germany, 2012.

Schakel, W. J., J. Wouter, B. van Arem, & B. D. Netten (2010). “Effects of Cooperative Adaptive Cruise Control on Traffic Flow Stability,” In *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems*, IEEE, New York, pp. 759–764.

Sheikholeslam, S., & C. A. Desoer (1993). “Longitudinal Control of a Platoon of Vehicles with No Communication of Lead Vehicle Information: A System Level Study,” *IEEE Transactions on Vehicular Technology,* Vol. 42, No. 4, pp. 546–554. https://doi.org/10.1109/25.260756.

Tadaki, S., M. Kikuchi, Y. Sugiyama, & S. Yukawa (1998). “Coupled Map Traffic Flow Simulator Based on Optimal Velocity Functions,” *Journal of the Physical Society of Japan,* Vol. 67, No. 7, pp. 2270–2276. https://doi.org/10.1143/JPSJ.67.2270.

Taieb-Maimon, M., & D. Shinar (2001). "Minimum and Comfortable Driving Headways: Reality Versus Perception," *Human Factors: The Journal of the Human Factors and Ergonomics Society,* Vol. 43, No. 1, pp. 159–172. https://doi.org/10.1518/001872001775992543.

Treiber, M., A. Hennecke, & D. Helbing (2000). "Congested Traffic States in Empirical Observations and Microscopic Simulations," *Physical Review E,* Vol. 62, No. 2, pp. 1805–1824. https://doi.org/10.1103/PhysRevE.62.1805.

Treiber, M., A. Hennecke, & D. Helbing (2000). "Microscopic Simulation of Congested Traffic," *Traffic and Granular Flow '99*. Springer, Berlin, pp. 365–376.

Treiber, M., & A. Kesting (2013). *Traffic Flow Dynamics: Data, Models and Simulation.* Springer, Berlin. https://doi.org/10.1007/978-3-642-32460-4.

Van Arem, B., C. J. van Driel, & R. Visser (2006). "The Impact of Cooperative Adaptive Cruise Control on Traffic Flow Characteristics," *IEEE Transactions on Intelligent Transportation Systems,* Vol. 7, No. 4, pp. 429–436.

Wang, X., & Y. Wang (2015). "Human-Aware Autonomous Control for Cooperative Adaptive Cruise Control (CACC) Systems," In *Proceedings of ASME 2015 Dynamic Systems and Control Conference, Volume 2,* p. V002T31A001, American Society of Mechanical Engineers, Washington, D.C.

Wiedemann, R (1974). "*Simulation Des Verkehrsflusses,"* Institute of Traffic Engineering, University of Karlsruhe, Germany, 1974.

Zhao, L., & J. Sun (2013). "Simulation Framework for Vehicle Platooning and Car-Following Behaviors Under Connected-Vehicle Environment," *Procedia—Social and Behavioral Sciences*, Vol. 96, 2013, pp. 914–924.

CHAPTER 6

CONCLUSIONS

The author developed and evaluated four connected and autonomous vehicles applications in an edge-centric TCPS. Through this evaluation, the author demonstrated the efficacy of these applications with four distinctive objectives. In the second Chapter of this dissertation, an adaptive queue prediction algorithm was developed using a machine-learning algorithm with a real-time feedback loop. The algorithm was evaluated using Simulation of Urban Mobility (SUMO) and Network Simulator 3 (NS3) to illustrate the effectiveness of the algorithm on a roadway network in Clemson, South Carolina, USA. The simulation analyses revealed that the adaptive queue prediction algorithm using the feedback loop has higher accuracy compared to the queue prediction without a feedback loop. The feedback loop was developed using a fixed sliding time window and dynamic sliding time window. Using a dynamic time window allows queue prediction data sets with unique patterns to be added from the evolving data streams in real-time depending on dynamic traffic conditions. Analyses demonstrated that the accuracy for the dynamic sliding time window is higher than the fixed sliding time window for a CV penetration level less than 50%. When the CV penetration is 50% or above, the performance of both fixed and dynamic sliding scenarios were very similar.

The third chapter of this dissertation investigated noise reduction models, the standard Kalman filter and Kalman filter based Rauch–Tung–Striebel (RTS) data smoothing techniques, to reduce the noise from the traffic data generated form the BSMs and the performance of the LSTM prediction model was evaluated for predicting traffic data (i.e., speed and space headway) using the resulting filtered data. The traffic data for speed and space headway used in this study were generated from the Enhanced Next Generation Simulation (NGSIM) dataset, which contains vehicle trajectory data for every one-tenth of a second. The analyses revealed that the model developed here for traffic data prediction can predict speed and space headway for different penetrations of connected vehicles with no significant difference from the ground truth data. Specifically, LSTM combined with an RTS noise reduction model reduced MAPE from 19% to 5% for speed prediction and from 27% to 9% for space headway prediction at a 5% penetration of CVs compared to LSTM alone. The reduction of MAPE value ranges from 1% to 14% for speed and 2% to 18% for space headway prediction with penetration of CVs ranges from 5% to 100% compared to LSTM alone. The statistical significance test with a 95% confidence interval confirmed that the predicted speed and space headway using LSTM combined with RTS is not significantly different from the ground truth for 5% to 100% CV penetration. A comparison of the three noise reduction models (Moving Average, standard Kalman filter and RTS filter) suggests that LSTM combined with RTS can achieve the best prediction performance in terms of RMSE, MAE and MAPE.

The fourth chapter of this dissertation developed an application for dynamic routing to reduce recovery time for major incidents in a connected environment which would result in increased safety, improved traffic operations, and reduced negative environmental effects. Simulation experiments were conducted on a roadway segment of I-26 using a traffic micro-simulator, VISSIM. The study confirmed that dynamic routing in a cooperative vehicle environment could prove to be a powerful tool in reducing traffic incident induced impacts on the network. The case study demonstrated that incidents that are not cleared quickly could affect freeway traffic operations. The results also demonstrated that the dynamic routing strategy reduced recovery time by 52% with a reduced verification time and an average reduction of 61% in arrival time of a tow truck at the incident scene. Recovery time was also reduced by 5% per minute reduction of verification time with the dynamic routing strategy.

The author developed an evaluation framework for the application of a driver car-following behavior model in CACC system design. in the fifth chapter of this dissertation, which considers user acceptance in terms of vehicle dynamics and string stability. The author adopted two widely used driver car-following behavior models, (a) the optimum velocity model (OVM) and (b) the intelligent driver model (IDM), to prove the efficacy of the evaluation framework developed in this research for CACC system design. A platoon of six vehicles was simulated for three traffic flow states with different acceleration and deceleration rates (uniform speed, the speed with constant acceleration, and speed with constant deceleration). The maximum acceleration or deceleration and the sum of the squares of the errors of the follower vehicle speed were measured to evaluate user acceptance in terms of vehicle dynamics and string stability. Analysis of the simulation results revealed that the OVM performed better at modeling a CACC system than did the IDM in terms of acceptable vehicle dynamics and string stability.

These connected and autonomous vehicle related research presented in this dissertation demonstrated the benefits of CAVs in improving traffic mobility. This dissertation would lead to the real-world deployment of these applications in an edge-centric TCPS as the dissertation focuses on the edge-centric deployment strategy of CAV applications. Edge centric architecture has proven effective in managing such applications by decoupling the dependency and distributing the computation based on the application requirements. To support different type of CAV applications with different levels of computation and memory requirements, DSRC devices are coupled with a computation unit capable of larger memory management, multiprocessor operations and user-friendly application development. The computation unit also permits the integration of other communication mediums, such as Bluetooth, Wi-Fi and LTE to support different type of CAV applications. As multiple CAV applications as presented in this dissertation could be supported simultaneously by the same CAV infrastructure with a TCPS and public investments will only include infrastructure investments (such as investments in roadside units and backend computing infrastructure), connected and autonomous vehicle applications can potentially provide significant economic benefits compared to its cost.