Validation of platoon control systems with explicit model of low-level control and communication-awareness

Muhammed, Sajith

Award date:
2017

Department of Electrical Engineering
Electronic Systems Group

Validation of Platoon Control Systems With Explicit Model of Low-Level Control and Communication-Awareness

Master thesis

Sajith Muhammed
1033797
sajith.muhammed@student.tue.nl

Committee members:
Dr. Dip Goswami (Supervisor TU/e)
Dr. Hong Li (Supervisor NXP)
Dr. Tanir Ozcelebi

Abstract

Platoon is chain of vehicles following each other to attain improved traffic and fuel efficiency. Platoon control algorithm aims to maintain short inter-vehicular distance under dynamic driving conditions. Control decision for a vehicle is taken based on the vehicle state information of the preceding vehicle that is received using Vehicle to Vehicle (V2V) communication and from sensors like radar and lidar. However, wireless communication does not guarantee perfect reception of these informations. Therefore it is necessary to have a communication-aware control algorithm that is able to make adequate decisions using previous information when no packets are received from the vehicles in front so that the vehicles avoid collision. This report presents a co-simulation environment for studying dependency of control on communication in vehicle platooning. A hierarchical platoon control algorithm with an explicit model of lower-level controller is validated in different communication network scenarios using the co-simulation environment. The upper level controller of the hierarchical control structure is replaced with a velocity trajectory planner. The proposed multi-rate control scheme integrates the lower-level controller into the platoon control structure. In order to function in the presence of packet drops and delays, predictive schemes are used making it a communication-aware controller.

The platoon vehicles are simulated on a highway with different traffic densities. Highway traffic is generated by using a microscopic traffic simulator called Simulation for Urban Mobility (SUMO). ns-3 sets up a vehicular communication system generating nodes with IEEE 802.11p standard devices. Communication network scenarios based on traffic density are realized in ns-3 to study the impact on the proposed controller.

Acknowledgement

It is with immense pleasure that I am presenting this thesis report. The experience at NXP Semiconductors was challenging and gratifying at the same time. I wish to express my gratitude to all who have supported me during these last 8 months. I would like to thank my university supervisor Dip Goswami, who helped me get this opportunity and guided me throughout my project. My special thanks to my supervisor at NXP, Hong Li, for providing me with ideas and valuable feedbacks. I also thank Tanir Ozcelebi from the Mathematics and Computer Science department, who helped me along with my supervisors with constructive feedback on my Preparation for Graduation Project. I would also like to thank PhD students-Amr Ibrahim and Chetan Belagal-who constantly supported me during the whole period. I would also like to extend my regards to EIT Digital for providing me with the opportunity for pursuing my master degree and being a part of KTH Royal Institute of Technology and Technical University of Eindhoven.

Further I would like to thank my wife, son, parents and my family and friends for their endless support and their faith in me. Most importantly, I would like to thank God Almighty for helping me reach where I am today.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A string of vehicles that follow each other trying to maintain a desired distance between them forms a platoon. The follower vehicle observes the motion of the predecessor and regulates its own motion using a predefined control strategy. The longitudinal dynamics is manually controlled only for the leading vehicle and the remaining vehicles follow the leading vehicle automatically. The main challenges are to reduce and maintain a constant inter-vehicular distance and at the same time not amplify the disturbances through the string. A lot of research have been done on improving the method used to observe the preceding vehicle's motion and developing appropriate platoon control strategy based on it. With advancement in electronic technology in the automotive industry, it is becoming easier to deploy such applications. Intelligent Transportation System (ITS) plays an important role in improving traffic flow, fuel efficiency and safety applications.

Over the past few decades, large number of researches have been carried out to achieve vehicular platooning for improved traffic flow. Adaptive Cruise Control (ACC) has been commercially available since the 90's. ACC has its roots from cruise control technology where the driver sets the velocity of the car and it maintains this velocity without the driver having to use the accelerator pedal. In ACC, a vehicle uses lidar or radar technology to sense the motion of the preceding vehicle. It has a decentralized approach as an individual vehicle has the sensor to measure the distance and apply control strategy to itself to regulate its motion. However, this decentralized behavior has been proven to propagate disturbances in the string of identical vehicles. As a result, it is required to have larger inter-vehicular spacing which leads to high aerodynamic drag on the following vehicles. It improves traffic efficiency to some extent and is considered more of a comfort application rather than for fuel efficiency.

Cooperative adaptive cruise control (CACC) allows additional method for observing the motion of a vehicle that it needs to follow. It makes use of the vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) wireless communication to exchange useful information. Each vehicle is made to broadcast its vehicle state information periodically in a platoon. Now the follower vehicle is able to react faster to the sudden changes in acceleration or velocity of the preceding vehicle. Also, the disturbances are attenuated making the string more stable at shorter inter-vehicular distances. Thus, CACC is able to provide improved

traffic dynamics by damping disturbances, improving highway capacity by reducing inter-vehicular distance and reducing fuel consumption by decreasing the aerodynamic drag force on the vehicles without compromising the safety of the vehicle [3].

CACC has its disadvantage of using wireless communication for exchanging vehicle state information. Exchanged information might be lost or delayed in a congested network and could compromise the safety of the application. While developing control algorithms for platooning, it is essential to consider these details. There should be some predictive scheme to counter the effect of information losses. However, testing the robustness of these control algorithms on real traffic, under various scenarios of communication network is very risky. This report presents a validation tool that co-simulates control strategy and V2V wireless communication to test any proposed control algorithms and predictive scheme for vehicle platooning.

Most of the studies done on control algorithms do not take into consideration the low-level dynamics of the platoon vehicles. The control strategy validated here has a hierarchical 2-layer control structure. This is multi-rate control loop structure that has a lower-level controller taking into account the low-level dynamics of a vehicle such as engine torque and vehicle dynamics. This control structure has been validated in various network congestion scenarios based on different traffic densities in the validation tool.

This report is structured as follows. Chapter 2 presents the current state-of-the-art on CACC and related works that was accomplished by the author during preparation of graduation project. Chapter 3 focuses on modeling the proposed validation tool. The conceptual control structure that is validated is presented in chapter 4. Chapter 5 characterizes the communication network scenarios based on various traffic densities. Also predictive schemes that can be used in the control structure has been proposed. Chapter 6 presents the simulation results of validating the control structure using the co-simulation environment. Finally, chapter 7 concludes the report and presents few potential future works.

Chapter 2

Background

This chapter covers sections which form the background for this thesis project. The current state-of-art in vehicle platooning are discussed. Also, approach for developing co-simulation environment of control and communication and lower-level controller which were proposed during preparation for graduation project has also been discussed.

2.1 Literature survey

Vehicle platooning encompasses controller design, algorithms, V2V and V2I communication, autonomous vehicles, sensor networks and many more fields. Many research papers are published in all these dimensions in order to improve road traffic, fuel efficiency and moreover, safety of the platoon. In this report, aspects related to control and V2V communication are considered. Most popular research for control of vehicle platooning are based on ACC and CACC. In ACC a vehicle follows the velocity of the preceding vehicle to maintain a constant inter-vehicular distance. The velocity of the preceding vehicle is tracked using sensors that are placed in the front of the following vehicle. Due to delay related to sensing and actuating, inter-vehicular distance is set to be a time-gap of 1 s [14]. This is to ensure safety in case of sudden change in acceleration. Since the distance between vehicles is large, aerodynamic (drag) force on the follower vehicle is high leading to lesser fuel efficiency [1].

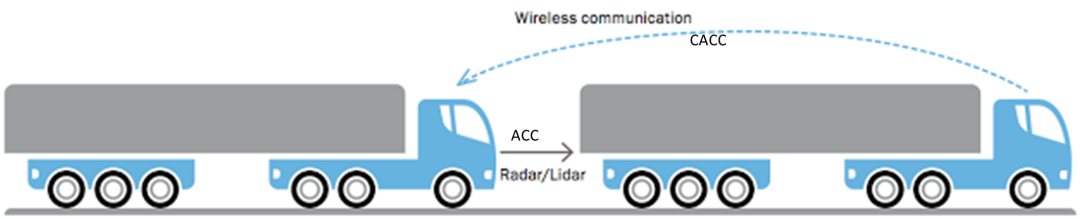


Figure 2.1: CACC and ACC used in vehicle platooning.

For better traffic and fuel efficiency, the time-gap between vehicles should be less. V2V communication used in CACC makes this possible by exchange of vehicle state information. Studies have proved better performance in CACC over ACC [2]. CACC includes concepts of communication enabled vehicle following and speed control [3]. Even though CACC enables lesser inter-vehicular distance compared to ACC [2], reliability of the wireless communication is to be accounted while developing the control algorithm.

Major issues related to wireless communication in CACC based platoons are packet delay, packet drop or packet error. Study on the delay of packets and its effect has been done in [5][6]. Some studies use ACC and CACC together. CACC is used as a feed-forward mechanism and ACC for vehicle following in [7]. In case of consecutive misses in the packets, the control is changed to ACC in [2].

Two layered (hierarchical) control architecture [9][8][11] is used in most of the studies where the upper-level controller decides the desired acceleration of a platoon vehicle with an objective to maintain desired inter-vehicular distance and the lower-level controller takes into consideration the non-linearities related to the vehicle such as engine dynamics, wind velocity, rolling friction. However, non-linearities are not considered in the vehicle model in most studies for simplicity, taking into consideration only a generalized longitudinal vehicle dynamics [6][12]. Numerous studies have been conducted for lower-level control of vehicles in relation to cruise control application [13][1]. Most of the lower-level controllers uses the assumption that the torque converter is locked and there is zero slip for the tires as cruise control is usually used at high gears, except for [11] which also studies the low speed conditions for this purpose. [13] uses a simpler approach where the engine dynamics and longitudinal vehicle dynamics are considered. A proportional-integral-derivative (PID) controller is used to control the traction force developed by the engine and this force is used to drive the vehicle.

In [9], a novel control strategy that utilizes information from multiple vehicles ahead by means of switching between multiple communication topologies is proposed. The different topologies are predecessor-follower (PF), pre-predecessor follower (PPF) and leader follower (LF). Switching mitigates the impact of loss of information from one vehicle, by utilizing the information from other vehicles ahead to obtain a stable and safer operation. This approach allows string stability for shorter inter-vehicular distance under imperfect network conditions.

2.2 Hierarchical platoon control architecture

A 2-layered (hierarchical) architectural style is followed in developing platoon control algorithms that use low-level model of platoon vehicles [8][11]. Layering separates the functionalities of the upper and lower-level controllers, making the implementation easier and scalable. The platoon controller in [9] is implemented using this architectural style, as shown in figure 2.2. The upper level controller uses a proportional-derivative (PD) controller that acts on the vehicle state information of preceding vehicles received via wireless communication and determines the desired acceleration at which the vehicle should

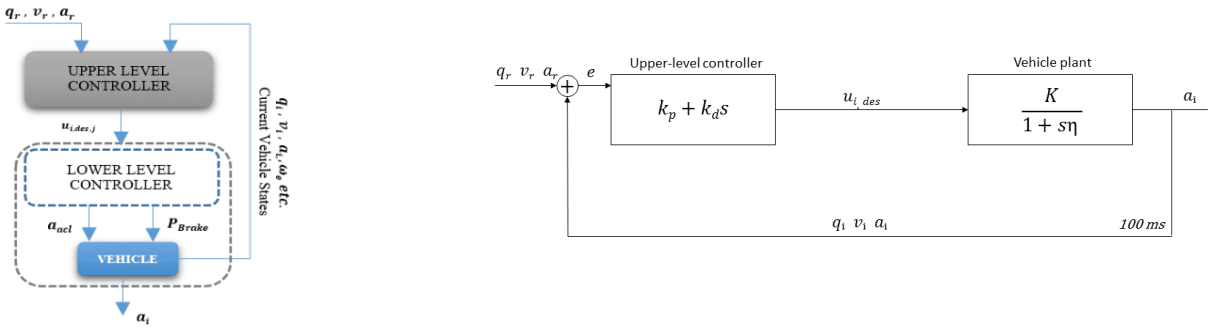


Figure 2.2: Hierarchical platoon control architecture [9]

move. Control objective is to maintain a constant inter-vehicular distance. The position error between two vehicles is given by,

$$e_{j,i} = q_j - q_i - (i-j)(r_i + h_{d,i}\Delta v_i) \tag{2.1}$$

where r_i is constant buffer gap between vehicles, $h_{d,i}$ is the constant headway time-gap constant and $j = i-1, i-2, 0$ for PF, PPF and LF topology respectively. The PD controller uses this error to perform vehicle following objective and is given by,

$$u_{i,des,j} = k_{p,i}e_{j,i} + k_{d,i}\dot{e}_{j,i} \tag{2.2}$$

The lower-level controller has the function of making the vehicle achieve the desired acceleration $u_{i,des,j}$. This is simplified in [9] to form a Generalized Vehicle Longitudinal Dynamics (GVLD) [12]. The plant takes desired acceleration as input from the upper level controller and provides the actual acceleration as output. It is linearized to form a first order system with a lag that is taken as the time constant of the internal actuator of the vehicle. This type of controller structure is used by many researchers as mentioned in the beginning of this chapter. The vehicle state information of the i^{th} vehicle that is sent to a follower vehicle is $x_i = \begin{bmatrix} q_i & v_i & a_i \end{bmatrix}^T$. The state space representation of the vehicle model is given by:

$$\dot{x}_i(t) = A_i x_i(t) + B_{s,i}u_{i,des,j}(t) \tag{2.3}$$

where the system and input matrix is given by,

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\eta_i^{-}1 \end{bmatrix}, B_{s,i} = \begin{bmatrix} 0 & 0 & \eta_i^{-}1 \end{bmatrix}$$

The desired acceleration for correcting the inter-vehicular distance error is the input in the state space equation and is given by,

$$u_{i,des,j} = k_{i,i-1}x_{i-1}(t) + k_{i,i}x_i(t) + k_{c,i} \tag{2.4}$$

The state space equation is lumped with $[\theta_n] = [x_0^T x_1^T ... x_n^T]^T$ together for all the vehicles to form the closed loop platoon system given by the equation:

$$\dot{\theta_n}(t) = A_{\theta_n}\theta_n(t) + B_{\theta_n,j}u^*(t) + f_{\theta_n} \tag{2.5}$$

where $B_{\theta_n,j}$ varies with the topology and f_{θ_n} is the feed-forward gain. A_0 is taken as the system matrix for the virtual reference vehicle in the platoon. It is given by $\dot{x}_0(t) = A_0 x_0(t)$.

$$A_{\theta_n} = \begin{bmatrix} A_0 & 0 & 0 & ... & 0 \\ 0 & A_1 & 0 & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & ... & A_n \end{bmatrix}, where A_0 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix},$$

$$B_{\theta_n,PF} = \begin{bmatrix} 0 & 0 & 0 & ... & 0 & 0 \\ B_{s,1}k_{1,0} & B_{s,1}k_{1,1} & 0 & ... & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & ... & B_{s,n}k_{n,n-1} & B_{s,n}k_{n,n} \end{bmatrix},$$

$$B_{\theta_n,PPF} = \begin{bmatrix} 0 & 0 & 0 & ... & 0 & 0 & \\ B_{s,1}k_{1,0} & B_{s,1}k_{1,1} & 0 & ... & 0 & 0 & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & ... & B_{s,n}k_{n,n-2} & 0 & B_{s,n}k_{n,n} \end{bmatrix},$$

$$B_{\theta_n,LF} = \begin{bmatrix} 0 & 0 & 0 & ... & 0 & 0 \\ B_{s,1}k_{1,0} & B_{s,1}k_{1,1} & 0 & ... & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ B_{s,1}k_{1,0} & 0 & 0 & ... & 0 & B_{s,n}k_{n,n} \end{bmatrix},$$

and $u^* = [\theta_n], f_{\theta_n} = \begin{bmatrix} 0 \\ B_{s,2}k_{c,2} \\ B_{s,3}k_{c,3} \\ . \\ . \\ B_{s,n}k_{c,n} \end{bmatrix}$,

This system is implemented in MATLAB using Zero-order hold (ZOH) mechanism with the sampling period as 100 ms. Every 100 ms, a packet is broadcast by each of the vehicles. As long as a packet is received, the delay in reception of packet is not of concern in [9] because all the vehicles update its vehicle state information at the end of each period.

It was found out in PF topology, that the disturbance propagate through the string in dynamic conditions resulting in maximum deviation between the last 2 vehicles. In LF topology, the deviation from reference distance is maximum between the first two vehicle. The remaining have constant inter-vehicular distance as any disturbance is reflected in all the vehicles. In PPF topology, during dynamic conditions the gap between 1-2 and 3-4 are similar to LF topology since 2 and 4 are dependent on information from reference vehicle at position 0. The gap between 0-1 and 2-3 varies in the same fashion similar to PF topology. The maximum deviation is between 2-3.

In [9], the author considers two situations with regards to sampling, namely, ideal sampling and non-ideal sampling. Ideal sampling means at least one packet is received by the follower vehicle based on any of the topologies. Non-ideal means that no packet is received at the end of a time period. Also received packets were given priority based on the topology in order to form a safe platoon. If multiple packets are received from all the topologies in a time period, priority for vehicle state information is given as PF >PPF >LF. In order to embed real-time communication setting between vehicles, a probabilistic network model was considered. Table 2.1 shows the number of packet misses by using the model for 800 samples at a sampling rate of 10 Hz [9]. This model was taken into consideration for proposing the new control strategy.

Table 2.1: Percentage of consecutive period of packet miss [9].

Consecutive Period of Missed packets	**1**	**2**	**3**	**4**
1	19.80%	4.70%	0.90%	1.50%
2	1.80%	0.10%	0	0
3	0.50%	0	0	0
4	0	0	0	0
5	0.05%	0	0	0
6	0	0	0	0
At least one packet received	77.85%	95.20%	99.10%	98.50%

In case of non-ideal samples, 2 control strategies are followed. The first one is feed-forward control where the input, desired acceleration $u^* = 0$. Only feed-forward is considered in case of no packet is received. The second is predictive control scheme in which a vehicle estimates the position of the vehicle in-front based on its recent buffer history. The vehicle assumes that the vehicle in-front has velocity and acceleration same as its own. Current position of the vehicle in-front is estimated by adding $s = ut + (1/2)at^2$ to its previous position that is 200 ms old. These information are taken as the state of the vehicle in-front to compute the new state of the i^{th} vehicle.

String stability of the closed loop platoon system with the switching strategy was analyzed using Lyapunov stability criterion and found to be asymptotically stable. Analysis of feed-forward control strategy shows a lot of jitter resulting in collision. Maximum deviation is seen between first 2 vehicles. Even though LF does not allow propagation of disturbances in the string, the other two topologies does. However, the predictive control strategy proved to have a better performance with reduced jitter.

This control strategy has been tested in a predefined network scenario shown in table 2.1. Is there any means to prove safe operation of the control strategy in various vehicular communication network scenarios which are dynamic and congested? Will the control strategy prove to be effective taking into considering the non-linearities associated with the platoon vehicles? What is the effect of packet delay in a congested network? These are some of the questions to be considered while implementing a platoon control algorithm.

2.3 Lower-level controller

There are several non-linearities associated with individual vehicles in a platoon such as engine dynamics, wind velocity, rolling friction, slope of the road, etc. In a real platooning scenario, these non-linearities does not allow a vehicle to attain a desired vehicle state. In hierarchical control structure, the lower-level controller has the functionality of providing pedal and brake positions to making the vehicle achieve the desired vehicle state that is requested by the upper-level controller.

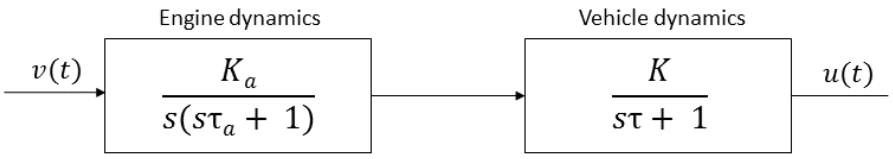


Figure 2.3: Open-loop low level dynamics of vehicle [13].

In this report, a vehicle model is realized that takes into consideration some low-level dynamics such as engine and vehicle dynamics, as shown in figure 2.3. This is based on [13], where a cruise controller is implemented with a PID controller that controls the vehicle model. Since the engine dynamics is related to torque of the motor, the upper-level has to provide reference velocity to the lower-level instead of desired acceleration as in [9]. This is because in an ideal situation (when no other non-linearities exist), engine torque is directly proportional to the longitudinal vehicle velocity [13]. The lower-level controller acts upon this vehicle model to achieve the reference velocity.

State space representation is used to model the low-level dynamics. The state variables of the low-level dynamics include velocity, acceleration and rate-change of acceleration and is represented as $[x] = [u \;\; a \;\; \dot{a}]^T$. The state space equation of the open-loop low-level dynamics of i^{th} vehicle is

$$\dot{x_i}(t) = A_i x_i(t) + B_i u_{i,des}(t) \tag{2.6}$$

where A_i is the system matrix, B_i is the input matrix and $u_{i,des}(t) = v(t)$ represents input to the model.

$$A_i = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -1/\tau_a\tau & -(\tau_a + \tau)/\tau_a\tau \end{bmatrix}, B_i = \begin{bmatrix} 0 \\ 0 \\ K_a K/\tau_a\tau \end{bmatrix}$$

The closed loop low level dynamics is as shown in figure 2.4. Now the input to the system also encompasses the error with the reference value V_{ref}. In the hierarchical control structure, the reference velocity given to the lower-level controller is V_{ref}. Input $u_{i,des}(t) = -K_1 x_i(t) + Fr$ where K_1 is the feedback gain, F is the feed-forward gain of the closed loop system and reference $r = V_{ref}$. Now the state space equation is

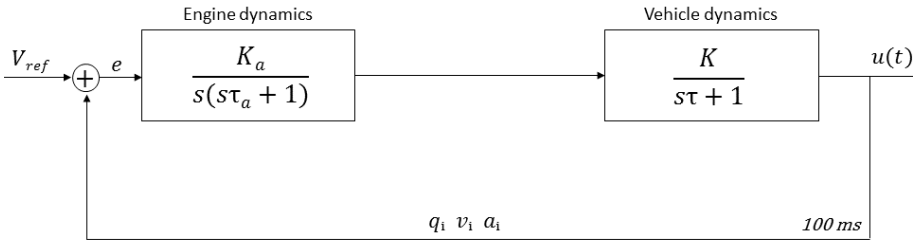


Figure 2.4: Closed-loop low level dynamics of vehicle

$$\dot{x_i}(t) = (A_i - B_i K_1)x_i(t) + Fr \tag{2.7}$$

Table 2.2 shows the values of the constants that are used for the lower-level controller.

Table 2.2: Parameters used in lower-level controller

Parameters	**Value**
τ	0.2 s
τ_a	0.2 s
K	100
K_a	10

The rotational force on engine in a real driving situation changes rapidly. The lower-level controller should act on the vehicle model at the same rate in order to achieve its objectives. Therefore, the execution time for the lower-level controller to compute the next state of the model should be very short. The execution time is assumed to be 2 ms, which is very short compared to the rate at which a vehicle broadcasts vehicle state information. Assuming there is no delay in the received packet from another vehicle and no delay in upper-level computations, the lower-level controller can run 50 times to attain the reference velocity. However, the lower-level controller imposes constraints on the upper-level controller due to the lag it imposes. Also delay of the received packet is not considered in the model. Therefore, appropriate interface is required with the upper-level considering these constraints. Moreover, platoon members do not have an identical low-level dynamics. This makes it necessary to have a multi-rate control, e.g. a velocity trajectory in this work, to maintain a safe distance between vehicles. These topics are discussed in chapter 4 in detail.

2.4 MATLAB - ns-3 Co-simulation Environment

MATLAB is a numerical computing environment that is used for scientific computations. It provides an easy programming platform that enables it to be used for various

applications. The control algorithm for CACC can be simulated using MATLAB. Since CACC involves wireless communication, ns-3 is used to simulate the communication network. It can be used to simulate various communication network scenarios based on network congestion. Simultaneous simulation of both tools can be used to study the impact of implementing CACC control algorithm in a degrading communication network. Server-Client interaction model used for MATLAB-ns3 interface is shown in figure 2.5.

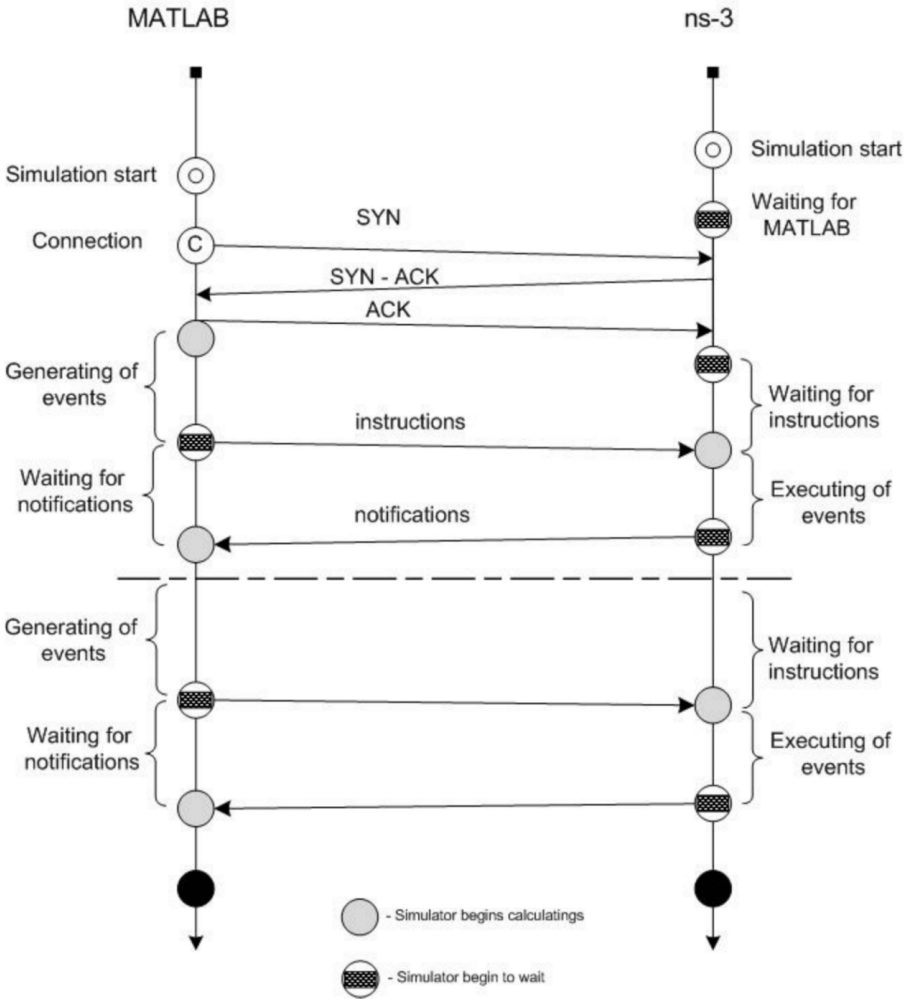


Figure 2.5: Interaction between MATLAB and ns-3 [15].

Integration of MATLAB and ns-3 has been previously done in [15] for ITS application simulation. [16] interfaces Simulation of Urban Mobility (SUMO) [10] for microscopic traffic simulation, in addition to MATLAB-ns3 co-simulation environment. In this, MATLAB functions as a client to the server SUMO. TraCI is an API used by SUMO to establish connection with MATLAB.

In [16], MATLAB runs on a Windows and ns-3 runs on Linux OS. Firewall is configured accordingly to initiate inter-process communication. Sockets programming is an easy ap-

proach for inter-process communication (IPC). Sockets are created in both the applications for bi-directional communication. To make sure that no packets are lost in the communication, transmission control protocol (TCP) is used. ns-3 acts as the TCP server as MATLAB requests for establishing a connection to ns-3. The TCPAcceptor class in ns-3 creates the socket in the server side, listening to incoming requests and bind to the address. Another class called TCPStream is responsible for data exchange after TCPAcceptor established the connection.

Once the connection is setup in [16], MATLAB fetches the current state of the vehicles that is simulated in SUMO. This state information includes the position, velocity and angle in x-y coordinates. This is put in a 1-D matrix and then sent as a string over the TCP to ns-3. ns-3 parses this string, update its mobility model using the parsed values and then broadcasts Cooperative Awareness Message (CAM).

How can this setup be utilized to develop a validation tool for studying the effect of imperfection in V2V communication on control algorithm for CACC? This has been addressed in the following chapter.

Chapter 3

CACC Platooning Problem and Validation Tool

This chapter focuses on problems associated with CACC based vehicle platooning and the need for co-simulation of platoon control algorithm and V2V communication. In order to study the dependency of platoon control on V2V communication, a validation tool has been set up. Modeling of this tool has been discussed.

3.1 CACC platooning problem

In CACC based vehicle platooning, vehicles communicate with each other using standards for V2V communication. IEEE 802.11p is an enhanced version of 802.11 for Intelligent Transportation Systems (ITS), featuring wireless access in vehicular environments (WAVE), a vehicular communication system. Vehicles exchange informations with each other as well as with roads and infrastructures for safety applications. In addition, platoon vehicles exchange information that are useful for taking decisions for merging, splitting, entering or leaving a platoon. This study focuses on the longitudinal control of platoon vehicles and the relevant vehicle state information exchanged are position coordinates, velocity and acceleration. These informations are broadcasted and the security and privacy of this data exchange is out of scope of this project. Here, platoon control is decentralized and decision is made separately for each vehicle depending on the data received from the vehicles in front.

What happens if no data is received due to network congestion? To what extend can the control algorithm tolerate the packet drops providing a safe platoon? These are important questions that needs to be answered while designing a robust control algorithm. A validation tool to study the robustness of control algorithm in various network congestion conditions have been implemented. Implementation of this validation tool is detailed in this chapter.

In this study, the control algorithm is used to control the velocity of a vehicle depending on the state of the vehicle in front. Depending on the information received from the

vehicle in front, each vehicle computes the trajectory of velocity that it should attain in order to maintain a safe inter-vehicular distance and efficient platoon. Details on velocity trajectory planning are detailed in the next chapter. However, control of individual vehicles in the platoon depends on several underlying dynamics. There are several non-linearity that are associated such as engine dynamics, aerodynamic drag, frictional forces, etc. To take these into account, a lower-level controller has to be designed to make sure that the vehicle actually attains the desired velocity that was computed by individual vehicles. Reaching the desired velocity is essential to maintain safe inter-vehicular distance between the platoon members. Design of a lower-level controller that takes into account some of the mentioned non-linearities are detailed in the previous chapter.

3.2 Validation tool

Creating a real scenario of vehicles for platoon purpose is possible in test tracks. Testing it in real traffic is not easy due to safety concerns. Also it is not easy to implement a wireless network congestion to check the dependency of control on communication reliability.

Validating the control for CACC based platooning should consider the imperfections associated with V2V communication. It should also consider an environment where the mobility of the transmitting nodes or vehicles resemble real driving behavior.

Combination of several tools can be used to impart features such as real network simulation, real driving behavior and control algorithm simulation. Such a co-simulation environment will resemble a real scenario of vehicle platooning. ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for modern network research. It can be used to simulate platoon vehicles as well as additional vehicles using WAVE standards to communicate. Simulation of Urban Mobility (SUMO) is an open source, highly portable road traffic simulation package designed to handle large road networks. It can be used to make the nodes simulated in ns-3 to have a real driving behavior in highway or urban traffic conditions. Moreover, it can provide a graphical user interface where users can view the mobility of each of the vehicles in the given traffic. MATLAB can be used as an interface for the co-simulation environment as well as simulate decentralized control algorithm for individual vehicles in the network. Design and implementation of the validation tools is detailed in the following sections.

3.2.1 Architecture of validation tool

The software architecture for the validation tool is based on client-server model. Inter-process communication is established between SUMO, MATLAB and ns-3 to simulate CACC based platoon control algorithm. With 3 different tools, a hierarchical client-server model is realized for inter-process communication.

The three individual applications function at 3 different layers as shown in figure 3.1. SUMO acts as a server in layer 1 and is responsible for simulating all the vehicles and the graphical user interface (GUI) for viewing them. MATLAB is the middle layer and

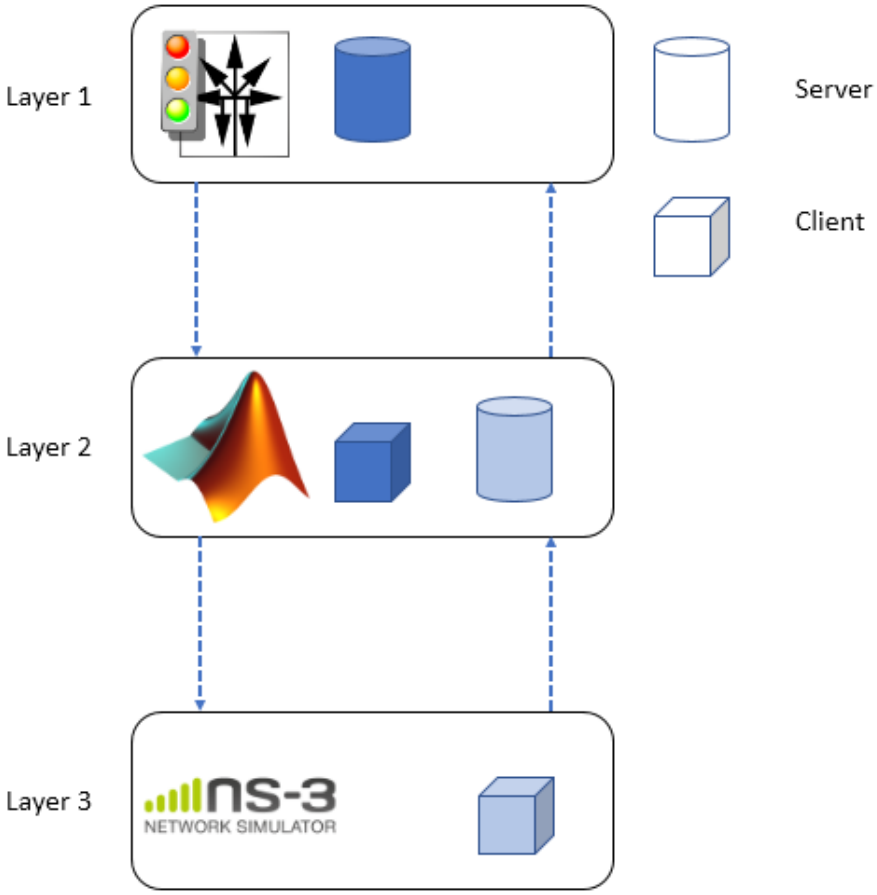


Figure 3.1: Architecture of co-simulation environment.

acts as the connection between SUMO and ns-3 and also computes the control algorithm. MATLAB serves as a client to SUMO and a server to ns-3. In the third layer, ns-3 simulates the V2V communication based on the vehicles state information it receives and informs the MATLAB about packet reception of the platoon members.

Layered architecture is used because each layer produces some output data that is taken as input to the next layer. Changes made in a layer does not affect the functionality of another layer. Each layer has its own role and this makes the tool portable and easy to maintain. More layers could be easily added by using this architectural model.

Client-server interaction model is realized for inter-process communication between each layers. Between Layer 1 and Layer 2, SUMO serves as server while MATLAB acts as client. Between Layer 2 and Layer 3, MATLAB serves as server and ns-3 as client. Layer 2 has both client and server. SUMO and MATLAB are run on Windows 7 operating system. ns-3 is on a virtual machine running Ubuntu 14.04. For reliable data exchange, the client-server architecture used in both cases are TCP based. Details on interaction between different layers are detailed in the following section.

3.2.2 Interaction model

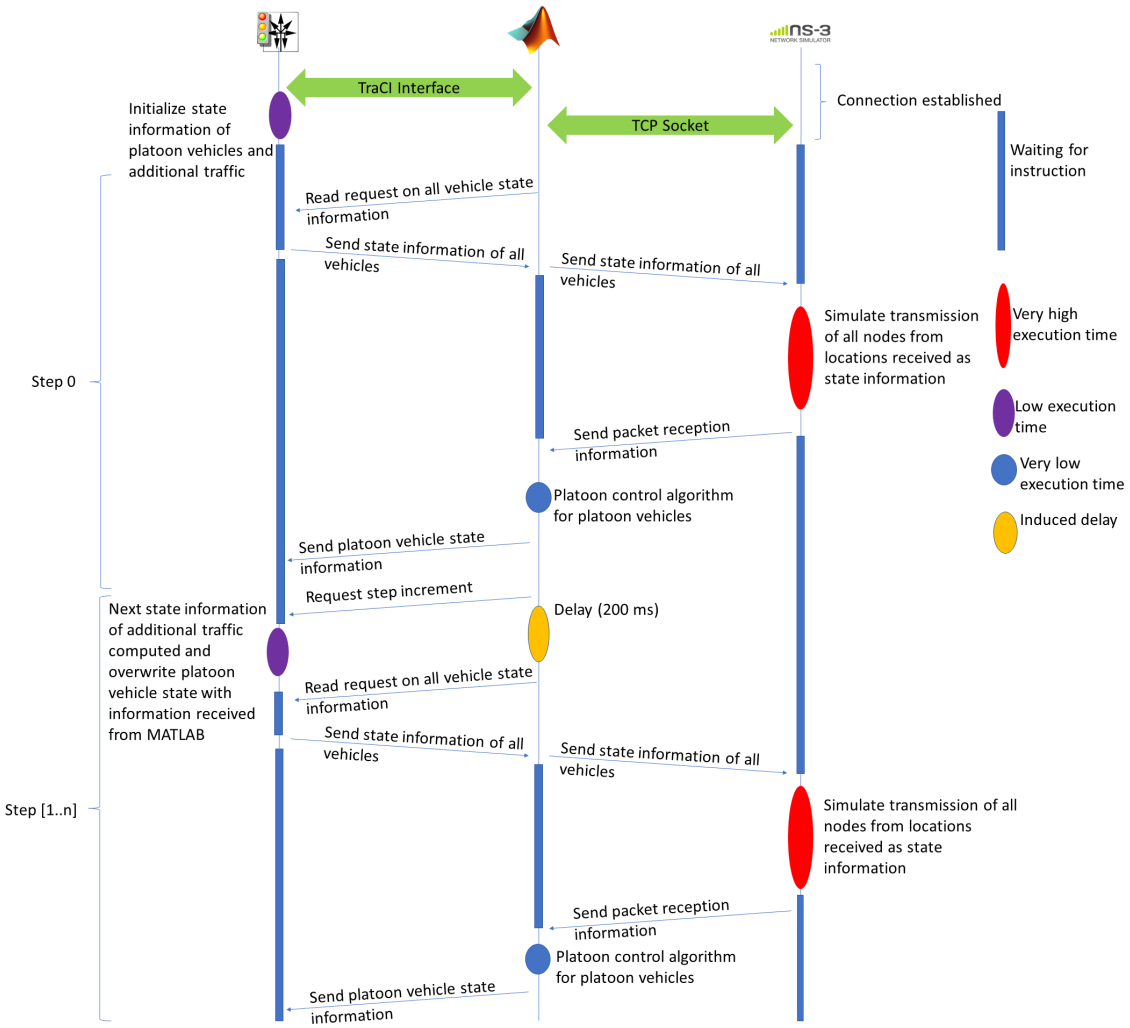


Figure 3.2: Interaction between SUMO-MATLAB-ns-3.

Interaction between all three programs (SUMO-MATLAB-ns-3) happen in steps, each step denoting 100 ms (time interval between consecutive message broadcast by a platoon vehicles). Before step zero starts, TCP connection is established between MATLAB-SUMO using TraCI and between MATLAB-ns-3 using socket programming. Data buffers are defined for exchange of data between them. Interaction model of the co-simulation environment has been shown in figure 3.2.

TCP based client-server interaction between SUMO and MATLAB is realized using Traffic Control Interface (TraCI). As a server, SUMO provides MATLAB access to a running road traffic simulation through TraCI and allows it to retrieve values of simulated objects and to manipulate their behavior “on-line”. Connection between MATLAB and ns-3 is realized using TCP based socket programming. Ports are opened in the operating

system to establish connection with ns-3 running on virtual machine.

Based on control computation for each vehicle, there are two categories of vehicles. The first category of vehicles consist of platoon members and the second category consist of additional traffic vehicles. Step 0 initializes all the tools with the desired initial vehicle states. Initial state information for platoon members are input in MATLAB and for the additional traffic through Extensible Markup Language (XML) file in SUMO. Control algorithm for the first category of vehicles is computed using the user defined platoon control algorithm in MATLAB. Control for additional vehicles are generated by SUMO, imitating real human driving behavior on highways. Control for platoon members are also generated in SUMO as it requires initialization in SUMO also but these are overwritten by TraCI functions using the platoon control information from MATLAB.

Once the control algorithm for first category of vehicles is computed in MATLAB, state information for these vehicles are sent to SUMO over TraCI. SUMO produces state information for the second category of vehicles and displays all the vehicles on SUMO GUI. Acting as a client of SUMO, MATLAB requests SUMO for state information of all vehicles. This request is done once in every step. ns-3 being a client of MATLAB, waits until the requested state information of all the vehicles are sent by MATLAB over socket connection to ns-3. Once ns-3 receives state information of all the vehicles, it simulates packet broadcast for each vehicle. At step 0, all vehicle broadcasts happen uniformly distributed over 100 ms and having uniform velocity mobility model. The same distribution is used for all further steps. After transmission, callback function in ns-3 is used to check and populate packet reception information such as packet delay or drop. When no more packet reception is expected, ns-3 starts sending packet reception details to MATLAB. MATLAB populates a lookup table with the packet reception information. This lookup table is accessed for computing the platoon control algorithm. Interaction model of the n^{th} step in the validation tool has been summarized in Algorithm 1.

Algorithm 1 Interaction model of step 'n' in validation tool

1: SUMO generates vehicle state information for additional traffic vehicles, uses the platoon vehicle state information which was overwritten at end of step 'n-1' from MATLAB and displays all vehicles in SUMO GUI
2: MATLAB starts with getting all vehicle state information from SUMO and forwards it to ns-3
3: ns-3 generates WAVE nodes in designated positions with uniform velocity mobility model and simulates packet broadcast of all nodes, uniformly distributed over 100 ms
4: ns-3 finds packet reception details for all platoon members and sends information to lookup table accessible to MATLAB
5: MATLAB computes platoon control algorithm based on the packet reception details in the lookup table and overwrites the platoon member state information in SUMO using TraCI functions

Chapter 4

Controller for platoon vehicles considering low level dynamics

The hierarchical control structure implemented in [9] uses an upper-level controller and lower-level controller. The upper-level controller computes the desired acceleration based on the vehicle's current state information and the state information received from the vehicle in front, depending on the communication topology. The main constraint on the upper-level controller is the communication network congestion which impacts the packet reception of platoon members. In case of packet drops, a predictive control scheme was taken as control algorithm in [9]. However, consecutive packet drops led to collision. The lower-level controller does not take into consideration any of the non-linearities coming from the engine, aerodynamic drag, gear configuration, road friction, etc. Instead, generalized longitudinal dynamics [12] combining the lower-level controller and a vehicle plant is considered in [9].

In a real-world scenario, platoon vehicles do not follow the reference acceleration provided by the upper-level controller. This is due to the non-linearities arising from the engine, road friction, gear, etc. Moreover, individual platoon members have their own non-linearities associated with it. So a lower-level controller is necessary to take into consideration these non-linearities and control the vehicle so that the vehicle could actually follow the desired acceleration from upper-level controller.

The CACC vehicle model used in [9] has several shortcomings. Firstly, the vehicle dynamics is iterated only once in every 100 ms, same as the V2V communication. The upper-level controller generates the desired acceleration at 100 ms intervals. Since there is no segregation between the upper and lower-level controllers, both run every 100 ms. Secondly, it is assumed that the desired acceleration of the vehicle generated by the upper-level controller is achieved by the vehicle. This is because a perfect lower-level controller is assumed in [9]. This assumption makes the vehicle trace the desired acceleration from the upper-level controller perfectly. The constraints that lower-level controller put on the upper-level controller are not considered. These problems can be addressed by using a lower-level controller that takes into account non-linearities and not limiting its run only once every 100 ms. Also, appropriate strategy has to be used to interface the connection

between upper and lower level controller. This chapter focuses on a decentralized platoon controller with a lower-level controller that takes into account the engine and vehicle dynamics and development of a multi-rate control loop structure.

4.1 Architecture of platoon controller with explicit model of lower-level controller

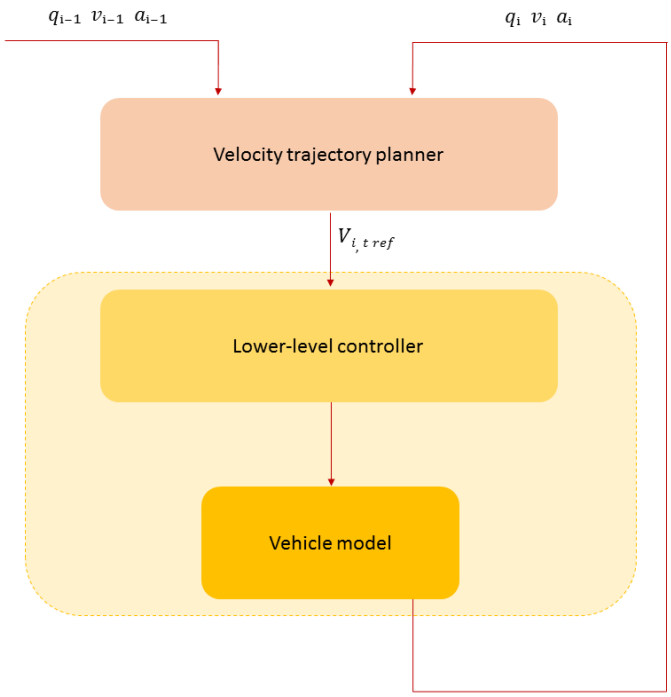


Figure 4.1: Architecture of decentralized platoon controller.

Control for platooning is realized as a decentralized system with each vehicle having its own controller and each controller can control its own non-linearities associated with it. The hierarchical control structure has a layered architecture (figure 4.1) because each layer has its own functionality. The lower-level controller takes reference velocity from its upper layer as input and helps the vehicle to attain the reference velocity (explained in chapter 2). Reference velocity is computed in the upper layer called velocity trajectory planner. This architecture integrates these two layers to perform multi-rate control because the velocity trajectory planner executes every 100 ms and lower-level controller executes every 2 ms.

To understand the necessity of a trajectory generator, let us assume that a platoon has to reach 40 m/s from rest, after which it moves 4 m every 100 ms. Depending on the low-level dynamics of individual vehicles, they may have different acceleration profiles to

attain this velocity. If a reference velocity is provided to the lower-level controller every 100 ms, the objective of maintaining a desired gap between the vehicles is checked only at the end of every 100 ms, i.e., the lower-level controller alone has to be trusted during this time period to avoid collisions. What would happen if a vehicle has very low acceleration profile and its immediate follower has high acceleration profile? So it would be safer to provide multiple desired reference velocities to the lower-level controller within the 100 ms period. Velocity trajectory planner and mechanism for interfacing the two layers are described in the following section.

4.1.1 Velocity trajectory planner

Control objective of this layer is to maintain a desired gap with the vehicle in front. It sets targets to lower-level controller at regular intervals to achieve the desired velocity in order to maintain the desired gap. Functionality of this layer is similar to the upper-level controller in [9]. Here a conceptual controller is depicted that is designed with two functionalities called trajectory generator and extrapolation.

Trajectory is a function of velocity and acceleration over a period of 100 ms and is computed in every step. Initial and final values of these variables are provided to the trajectory generator. Initial values are the velocity (v_{init}) and acceleration (a_{init}) of the vehicle at the beginning of a step. Final values are the velocity (v_{final}) and acceleration (a_{final}) of the predecessor vehicle at the beginning of the step. The function generates a trajectory by solving a third order polynomial of velocity given by

$$v_{i,t} = c_1 + c_2 t + c_3 t^2 + c_4 t^3 \tag{4.1}$$

$$a_{i,t} = c_2 + 2c_3 t + 3c_4 t^2 \tag{4.2}$$

where $t = (t_{init}, t_{final})$ and $c = [c_1 c_2 c_3 c_4]^T$. Co-efficient c is computed using $c = A.B^{-1}$ with

$$A = \begin{bmatrix} 1 & t_{init} & t_{init}^2 & t_{init}^3 \\ 1 & t_{final} & t_{final}^2 & t_{final}^3 \\ 0 & 1 & 2t_{init} & 3t_{init}^2 \\ 0 & 1 & 2t_{final} & 3t_{final}^2 \end{bmatrix}, B = \begin{bmatrix} v_{init} \\ v_{final} \\ a_{init} \\ a_{final} \end{bmatrix},$$

Extrapolation functions as an offset on the generated trajectory such that the vehicle maintains the desired gap. Velocity offset (δv_i) is generated by finding the difference between the actual inter-vehicular gap and desired inter-vehicular gap between two consecutive vehicles and making the vehicle reduce this difference to zero over some time span (T_s). The difference in gap is reduced to zero by adding velocity offset to the trajectory that was generated. The position error between two vehicles is given by,

$$e_i = q_{i-1} - q_i - (r_i + h_{d,i}\Delta v) \tag{4.3}$$

$$\delta v_i = \frac{e_i}{T_s} \tag{4.4}$$

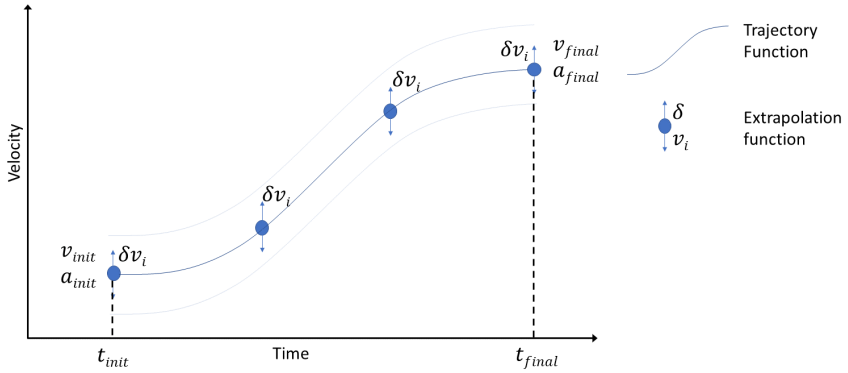


Figure 4.2: Trajectory plan for i^{th} vehicle

where r_i is constant buffer gap (considered as 3 m for reduced aerodynamics drag force) between vehicles, $h_{d,i}$ is the constant headway time-gap (considered as 0.5 s for safety reasons) and Δv is the difference in velocity with predecessor vehicle. Both functionalities are combined (figure 4.2) to form the final velocity trajectory ($V_{i,t}$) over a time period of 100 ms from t_{init} to t_{final} are given by

$$V_{i,t} = v_{i,t} + \delta v_i \tag{4.5}$$

If the time span (T_s) is very short, the vehicle experiences more jerk. However, if time span is increased to improve comfortableness, the reactiveness of the platoon control to abrupt changes in velocity reduces. This further demands an increase in constant buffer gap (leading to more aerodynamic drag force) to avoid collision. By trial and error method on highway driving conditions, the time span was taken as 1 second for comfortable driving experience (jerk $< \pm 0.35$ m/s^3) without compromising safety at 3 m constant buffer gap. This value is subject to changes depending upon the driving and road conditions.

4.1.2 Constraints of lower-level controller and Multi-rate loop

The lower-level controller receives sampled values of the trajectory that was generated by velocity trajectory planner. Design of lower-level controller has already been described in chapter 2. This section discusses multi-rate loop, sampling of the trajectory to provide reference velocity to the lower-level controller and solution to the constraints that the lower layer has on the upper layer.

The lower-level controller runs 50 loops in 100 ms. If a reference velocity is provided to the lower-level controller only once in every 100 ms, it runs 50 cycles trying to reach this velocity. To assure that the lower-level controller attains the desired velocity at the end of each 100 ms, sampling of the velocity trajectory for a series of reference velocities are required during each 100 ms. Selection of sampling rate is based on trade-off between jerk or driving comfort and safety. If the sampling rate is high, the lower-level controller has to run with new reference velocities frequently which in turn changes acceleration leading

to jerks. If sampling rate is less, there will be fewer reference velocities for the lower-level controller resulting in a smoother acceleration profile.

In order to attain a smooth and safe platoon, the velocity trajectory is sampled thrice in 100 ms for simulation purposes. Assuming the vehicle is traveling at 30 m/s, the vehicle travels only 1 m in the new sampling period. This makes the inter-vehicular distance uncertain by of only 1 m which is less compared to the constant buffer gap (r_i) between the vehicles.

Adding a lower-level controller that induces its own lag into actuation of the desired acceleration will lead to uncertainties that needs to be addressed. Lag has its effect on the inter-vehicular distance. For instance, if the platoon leader is accelerating, lag in the lower-level controller of the follower will impact the inter-vehicular distance between them. The lag creates deviation from desired gap by increasing it in this case. Lag in lower-level controller has adverse effect if the leader applies sudden braking because it might lead to collision. This lag in acceleration induced by the lower-level controller is a constraint on the upper-level controller. Adequate measure needs to be taken in designing the interface between the layers, taking into consideration this issue in order to meet the platoon controller objective of maintaining a desired inter-vehicular distance.

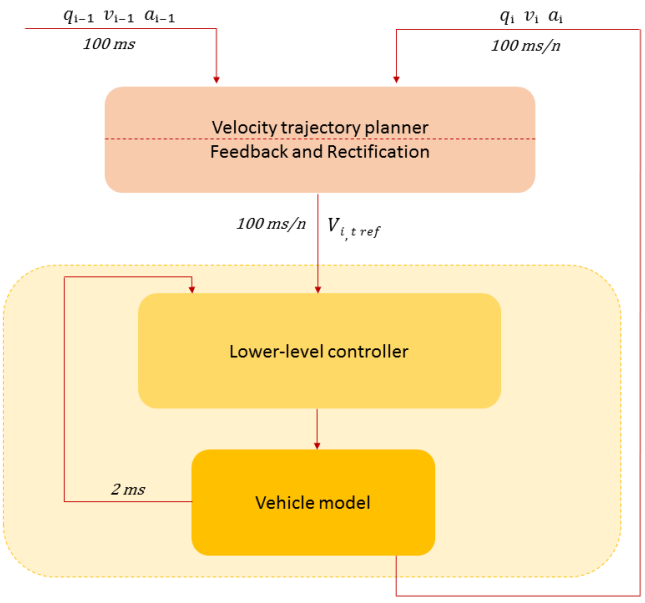


Figure 4.3: Multi-rate control loop.

In order to overcome the constraints that the lower level imposes on the upper-level controller, feedback from lower-layer and rectification of reference velocity to lower-layer is required. This is a concept proposed, that helps in rectifying the lag problem and maintaining lesser deviation from the desired inter-vehicular distance between the vehicles. For a sampling rate of n of the velocity trajectory within each 100 ms, feedback is taken $n-1$ times from the lower-level controller to upper-layer. Feedback is taken to check if the

vehicle reached the reference velocity. However due to lag, in most cases, the vehicle does not actually reach the reference velocity. Rectification is done by varying the reference velocity for the next 100/n ms. An offset is equal to the difference of reference velocity and the actual velocity the vehicle attained during the previous 100/n ms and this is added to reference velocity for the next 100/n ms. The architecture for conceptual platoon control structure in i^{th} vehicle with different control rates is shown in figure 4.3.

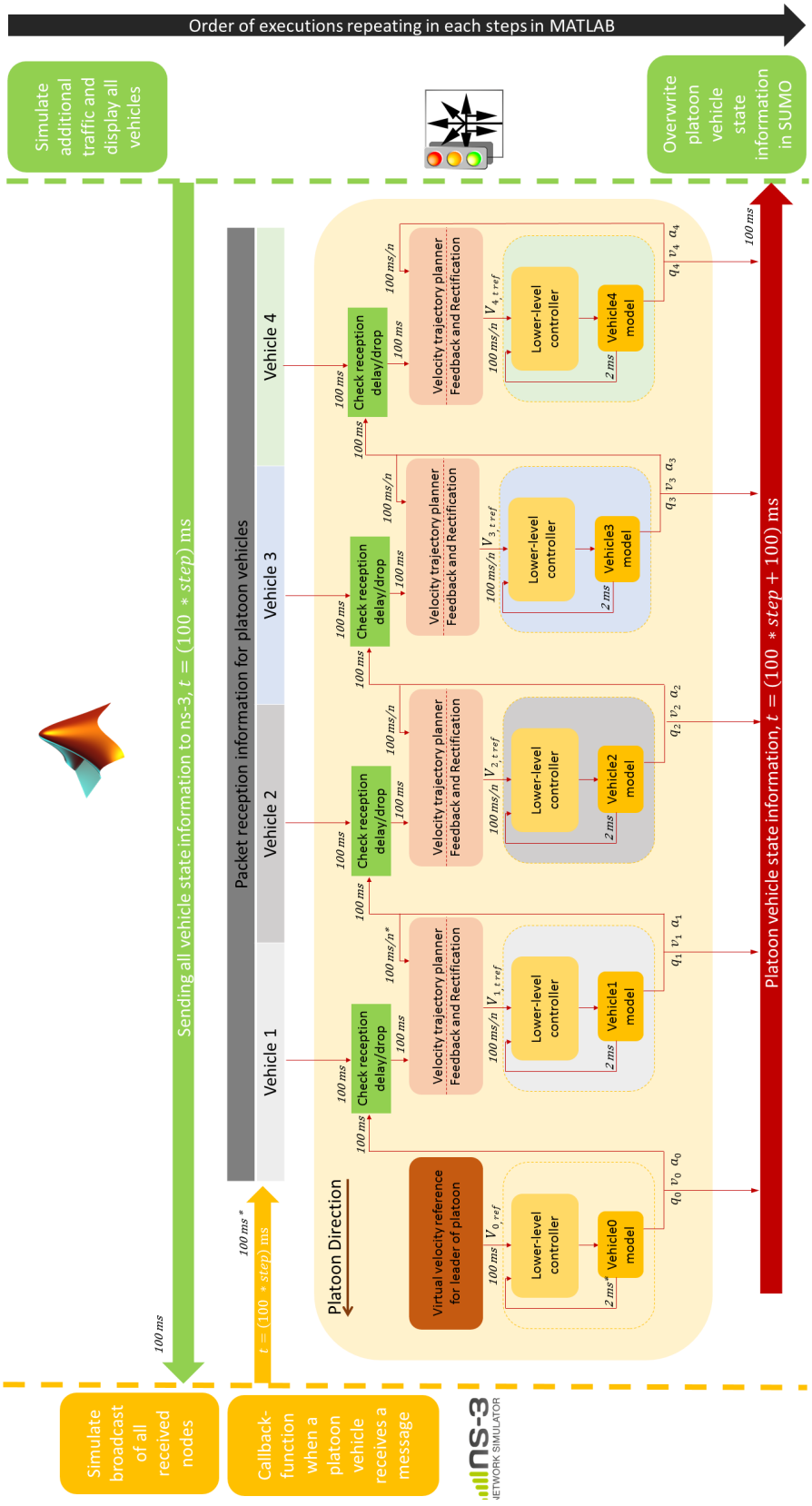
As previously explained, lower-level controller deals with the low-level (engine and vehicle) dynamics of the vehicle. It is therefore necessary while designing the lower-level controller that each cycle or loop should run in a very small time period. For instance, if the vehicle is running at constant 30 m/s, it would cover a distance of 6 cm in 2 ms. A 2 ms loop on the lower layer enables it to control velocity for small fractions of distance. This results in a very short time period per loop for lower-level controller compared to the time period of the upper layer. As seen in figure 4.3, there are now 3 different rates; the inter-vehicular communication running at 100 ms, feedback and rectification of reference velocity running at 100/n ms and the lower-level controller at 2 ms. Since there are multiple loops at different rates within the same control structure, the term multi-rate control loop was coined.

4.2 Validation of proposed platoon control algorithm

In chapter 3, the validation tool for validating any proposed control algorithms was presented. This co-simulation environment used SUMO for microscopic traffic simulation, MATLAB for platoon control simulation and ns-3 for simulation of various network scenarios. The platoon control algorithm with multi-rate loop proposed in this chapter will be validated in various communication network scenarios which are presented in chapter 5.

Decentralized platoon control algorithm for 5 vehicles are integrated into the co-simulation environment as shown in figure 4.4. The leader (vehicle 0) uses a simple control structure with no velocity trajectory generator. Virtual velocity reference is provided to the lower-level controller of leader vehicle every 100 ms (see chapter 6). Control for the remaining follower vehicles are implemented with multi-rate control loops.

At the beginning of each simulation step (corresponding to 100 ms), vehicle state information of all vehicles are sent from SUMO to ns-3 through MATLAB. ns-3 then simulates wireless communication of nodes using the received vehicle state informations. Information regarding packet drops and delays, obtained from ns-3 are populated in MATLAB table every 100 ms. Each vehicle looks up this table to check if it received vehicle state information from the vehicle in front. If a packet is received, velocity trajectory is planned to maintain the desired inter-vehicular distance. If the packet is delayed or dropped, predictive schemes are used (see communication-aware controller in chapter 5) to estimate information in the expected packet. At the end of each simulation step in MATLAB, vehicle state information of all platoon vehicles are sent to SUMO. These vehicle state informations are overwritten to the platoon vehicles in SUMO in the next step.



* 2 ms, 100 ms/n and 100 ms denotes time intervals at which an input is required

Figure 4.4: Integration of platoon control algorithm with multi-rate loop of 5 vehicles into co-simulation environment.

Chapter 5

Communication characterization and Communication aware-controller

Reliability of the control algorithm depends on packet reception. With increasing deployment of V2V and V2I communications, there is more traffic in communication channels. To verify the robustness of the control algorithm using the validation tool, it has to be validated in various communication network scenarios. These scenarios are based on the congestion on the channel used for vehicle platooning. For simulation purpose, we assume that all vehicles on highway have devices communicating by IEEE 802.11p standard in the same channel. Hence, the network congestion is varied by varying the vehicle traffic density.

Various communication network scenarios are characterized in this section. In order to attain a safe platoon, prediction schemes are used to formulate a communication-aware controller.

5.1 Communication network scenarios

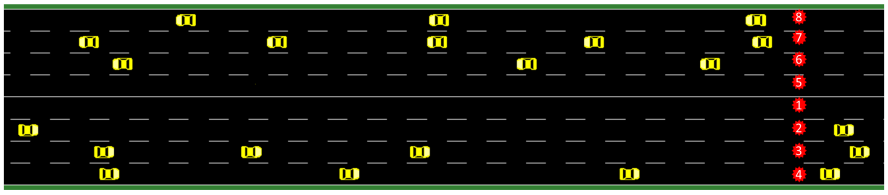


Figure 5.1: 8-lane (bidirectional) highway traffic.

The communication network simulation is done on an 8-lane (bidirectional) highway traffic. Length of the highway is 3 km, after which the vehicles takes a U-turn. Platoon vehicles and additional traffic vehicles are 4 m and 3 m long, respectively. All vehicles can go up to a maximum velocity of 30 m/s on the highway. Each vehicle can have acceleration between -3 and 2 m/s^2. Vehicles in lane 1, 2, 3 and 4 are moving towards east and 5, 6,

Table 5.1: Communication network scenarios based on vehicle traffic density

#	Scenario I	Scenario II	Scenario III	Scenario IV	Scenario V	Scenario VI	Scenario VII
Platoon vehicles	5	5	5	5	5	5	5
Additional traffic vehicles	0	95	195	295	395	495	595
Total vehicles on highway	5	100	200	300	400	500	600

7 and 8 are moving towards west, as shown in figure 5.1. Lane 1 and 5 are dedicated for platoon vehicles whereas the additional traffic vehicles move through the remaining lanes.

Various communication scenarios are realized based on traffic density over the 3 km highway. In order to have communication network congestion, the density of vehicles in the highway are increased. Table 5.1 shows various scenarios depending on total number of vehicles on the highway.

The least number of additional traffic vehicles is 0, in scenario I. This means that either there are no additional traffic vehicles on the highway or the platoon vehicles use a dedicated channel for vehicle platooning. In the latter case, congestion could happen only if there are too many platoons using the same channel. We assume that there is only 1 platoon using the channel. Additional traffic vehicles are increased by 100 in each scenarios. Considering safe trailing distance to be 1 second in highways, the maximum number of additional traffic vehicles that could be contained on the designed highway is approximately 545. Scenario VII represents our worst case of congestion that is possible on the highway, with total of 600 vehicles. More lanes and vehicles can be added to have a more congested network.

5.2 Characterization of Communication Network

Wireless Access in Vehicular Environment (WAVE) standards by IEEE makes vehicular communication very inter-operable. The IEEE 1609 family of standards for WAVE makes it possible to have a homogeneous communication interface for different car manufacturers. The WAVE protocol stack available in ns-3 is used for simulation of the communication network scenarios. Vehicles are simulated as nodes in ns-3 with each node having WAVE network devices installed in them. Settings for the physical and MAC layers are shown in table 5.2. One UDP (User Datagram Protocol) socket is associated with each device and is used for broadcasting information. Each node are configured to periodically broadcast a packet of 500 bytes. Broadcast of each node are uniformly (random offset) distributed over the first 100 ms. All the following broadcasts are done periodically after every 100 ms.

After the vehicle state information of all vehicles are received in ns-3 (see chapter 3), it simulates them with a constant velocity mobility model before they start broadcasting. Since Line-of-Sight (LOS) is not always achieved in vehicular networks, shadowing effect is also considered. Propagation loss models such as Nakagami and Three log distance loss model are used along with constant speed propagation delay model.

Characterization of communication for vehicle platooning includes determining packet

Table 5.2: Communication simulation parameters

Parameters	Value
Data rate	6 Mbps
Message rate	10 Hz
Tx power level	23 dBm
Channel BW	10 MHz
Energy detection threshold	-82 dBm
Maximum Delay for packet reception	100 ms

reception information such as packet drops, average delay and number of consecutive packet drops in each link over a period of time. It is important to characterize these information. Platoon control algorithms should be robust to work in different communication network scenarios. However, beyond certain amount of traffic congestion, undesired behavior could occur if there are no means for communication. Characterizing the communication network scenarios helps in determining to what extend a proposed platoon control algorithm can operate without displaying undesired behavior.

Packet reception in each communication links are checked in ns-3 using callback functions. Each packet's header contains the node number and the time at which it is transmitted. When a node receives a packet, it performs callback to a function that keeps track of packet reception of all platoon members and also calculates the time delay for reception. All these informations are sent at the end of a step to MATLAB (see chapter 3). These informations are populated in an EXCEL sheet for each steps and subsequently the consecutive number of packet drops are found after 1250 steps (representing 125 s). Since a destination node is expected to receive only 1 packet from a source node in each 100 ms, packet reception ratio (PRR) for a link (source vehicle -> destination vehicle) is computed as the percentage of number of packets received by the destination node to the number of transmissions by the source node in 1250 steps.

Communication characterization of the scenarios in Table 5.1 is shown in Table 5.3. It is observed that there are no packet drops in scenario I. This is because there are no additional traffic vehicles causing congestion in the network. The drops and average delays increases with each scenario because with more nodes the channel becomes busier. Scenario VII represents a very highly congested network with PRR as low as 44.275%. The number of consecutive periods of packet misses are also highest for scenario VII, with 17 misses for the links vehicle $1->$ vehicle 2 and vehicle $1->$ vehicle 3. In almost all cases, PRR reduces with increase in distance between the vehicles. This is because of propagation losses in the network which increases with distance. Also fading due to shadowing and multi-path propagation losses plays a role in this. In some cases, PRR improves with increase in distance between vehicle (PRR for the link vehicle $0->$ vehicle 1 is less than PRR for vehicle $0->$ vehicle 2 in scenario II). This is because of change in environment around the nodes. Since the nodes are moving, there are chances for interference and exposed terminal situations.

Table 5.3: Communication characterization of 7 scenarios over 1250 steps

	Communication Characterization		0 ->1	0 ->2	0 ->3	0 ->4	1 ->2	1 ->3	1 ->4	2 ->3	2 ->4	3 ->4
Scenario I	PRR (%)		100.000	100.000	100.000	100.000	100.000	100.000	100.000	100.000	100.000	100.000
	Average delay (ms)		4.585	4.585	4.585	4.585	3.486	3.486	3.486	2.594	2.594	2.531
Scenario II	PRR (%)		98.164	98.484	98.244	98.005	93.216	92.099	92.099	93.695	92.658	98.404
	Average delay (ms)		4.594	4.594	4.594	4.594	3.481	3.481	3.481	2.619	2.621	3.051
	Consecutive number of periods of packet misses	1	1.197	1.357	1.437	1.676	5.906	6.544	6.145	5.267	6.305	1.437
		2	0.638	0.160	0.319	0.319	0.638	0.638	1.756	0.798	0.798	0.160
		3	0	0	0	0	0.239	0.718	0	0.239	0.239	0
Scenario III	PRR (%)		94.436	94.201	93.260	93.652	91.458	90.517	89.969	94.671	93.966	95.768
	Average delay (ms)		4.833	4.824	4.839	4.827	4.315	4.308	4.315	3.043	3.059	3.218
	Consecutive number of periods of packet misses	1	4.545	4.624	5.408	4.859	7.210	7.680	7.915	3.918	5.016	3.448
		2	0.784	0.940	1.097	0.940	1.097	1.097	1.881	1.097	0.784	0.784
		3	0.235	0.235	0.235	0.235	0.235	0.705	0.235	0	0.235	0.000
		4	0	0	0	0.313	0	0	0	0.313	0	0
Scenario IV	PRR (%)		82.226	80.945	78.783	78.463	87.030	85.508	85.188	89.752	88.311	90.552
	Average delay (ms)		4.766	4.752	4.752	4.758	5.335	5.342	5.322	3.523	3.507	3.775
	Consecutive number of periods of packet misses	1	7.446	7.926	8.807	9.528	9.367	8.487	9.127	7.046	8.167	7.206
		2	3.523	4.964	4.323	4.644	2.082	2.882	2.082	1.922	1.121	1.761
		3	2.162	1.922	3.122	1.441	1.201	1.922	2.162	0.961	1.681	0.480
		4	1.601	2.242	2.882	3.523	0.320	0.320	0.641	0.320	0.320	0
		5	2.002	0.801	0.400	1.201	0	0.400	0.801	0	0.400	0
		6	0.480	0.480	0.961	0.480	0	0.480	0	0	0	0
		7	0.560	0	0	0	0	0	0	0	0	0
		8	0	0	0	0	0	0	0	0	0	0
		9	0	0.721	0.721	0.721	0	0	0	0	0	0
Scenario V	PRR (%)		80.220	79.199	77.237	78.022	84.301	82.025	81.947	84.694	82.025	86.421
	Average delay (ms)		8.095	8.101	8.182	8.189	7.430	7.468	7.504	4.912	4.875	6.799
	Consecutive number of periods of packet misses	1	7.143	6.750	7.143	7.614	9.890	11.774	11.695	11.146	11.538	8.163
		2	4.553	5.651	5.651	4.553	4.396	5.495	5.338	2.512	4.082	3.925
		3	4.003	3.061	3.532	3.297	1.413	0.706	0.706	0.942	0.471	0.235
		4	1.256	1.256	0.942	1.884	0	0	0.314	0.314	0.942	0
		5	0	0.392	0.785	0	0	0	0	0.392	0.392	0.785
		6	0.471	0.471	1.413	1.413	0	0	0	0	0	0.471
		7	0	0	0	0	0	0	0	0	0.549	0
		8	1.256	0.628	0.628	0.628	0	0	0	0	0	0
		9	0	0.706	0.706	0.706	0	0	0	0	0	0
		10	0	0	0	0	0	0	0	0	0	0
		11	0	0	0	0.863	0	0	0	0	0	0
		12	0	1.884	0.942	0	0	0	0	0	0	0
		13	1.020	0	1.020	1.020	0	0	0	0	0	0
Scenario VI	PRR (%)		68.425	66.747	64.748	64.668	82.814	81.535	78.817	80.655	79.536	81.375
	Average delay (ms)		9.277	9.318	9.165	9.095	10.325	10.421	10.294	7.788	7.535	11.112
	Consecutive number of periods of packet misses	1	14.149	15.108	14.468	15.028	12.470	11.671	14.548	11.910	12.310	13.030
		2	8.633	10.711	10.232	9.113	4.157	5.116	3.038	4.796	4.956	5.116
		3	4.077	4.317	6.235	6.235	0.240	1.679	2.158	1.679	2.878	0.480
		4	1.599	1.279	2.238	2.878	0.320	0	0.959	0.959	0.320	0
		5	1.599	1.199	0.799	0.799	0	0	0	0	0	0
		6	0.959	0	0	0	0	0	0.480	0	0	0
		7	0	0	0	0	0	0	0	0	0	0
		8	0.639	0.639	1.279	1.279	0	0	0	0	0	0
Scenario VII	PRR (%)		48.038	46.918	45.957	44.275	72.138	70.296	67.654	72.698	69.576	74.620
	Average delay (ms)		10.458	10.534	10.147	10.000	13.220	13.373	13.083	10.664	10.195	15.425
	Consecutive number of periods of packet misses	1	10.568	10.649	10.408	10.729	11.369	15.132	14.732	13.531	14.331	13.771
		2	10.729	11.209	10.568	11.369	8.006	8.006	5.604	5.604	7.686	6.405
		3	7.446	7.446	7.686	9.367	3.122	2.882	3.363	3.363	4.083	2.402
		4	7.366	6.965	7.366	6.725	0.961	0.961	1.922	1.922	1.601	0.641
		5	5.604	5.604	4.804	6.005	0.400	0.400	1.201	1.201		
		6	3.363	2.402	3.363	2.882		0.480	0.961	0.961	1.922	1.441
		7	1.681	2.242	2.242	3.363	1.121	0.560	0	0	0	0
		8	1.922	3.203	2.562	1.922	0.641	0	0	0	0	0
		9	0	0	0.721	0.721	0	0	0	0.721	0	0.721
		10	2.402	2.082	2.402	1.601	0	0	0	0	0.801	0
		11	0.881	0.881	0	0	0	0	0	0	0	0
		12	0	0.961	0.961	0	0.961	0	0	0	0	0
		13	0	0	1.041	0	0	0	1.041	0	0	0
		14	0	1.121		1.121	0	0	0	0	0	0
		15	0	0	0	0	0	0	0	0	0	0
		16	0	0	0	0	0	0	0	0	0	0
		17	0	0	0	0	1.361	1.361	0	0	0	0

5.3 Communication-aware controller

So far, the controller has been designed for an ideal condition where state information of the vehicle in front is readily available (platoon control algorithm in chapter 4). However, packet drops and delay are inevitable in wireless communication and is evident from the communication characterization in previous section.

5.3.1 Topology for V2V communication in platooning

With the introduction of lower-level controller, switching between topologies as in [9] will lead to jerk and less travel comfort. This is because individual vehicles induce their own lag and it accumulates towards the end of the platoon. If a platoon member switches from PF to LF (while all vehicles are experiencing acceleration), the follower vehicle will experience jerk because velocity of transmitting vehicle in LF topology is more than that in PF topology. The condition worsens when topology continuously switches in congested traffics. To avoid this situation, only one topology is followed here. As observed in Table 5.3, PF topology has the best values for PRR and the consecutive periods of packet drops in most cases. This is because LOS is possible and also the distance is shortest in PF topology. However, this behavior might change in dense traffic condition due to interference. Therefore, PF topology is used for receiving vehicle state informations.

5.3.2 Predictive control scheme

At the beginning of each period of 100 ms, each platoon member has to check if it has received state information from the vehicle in front. Only if information is received, a platoon member can compute reference velocity for safe platooning. However, as seen in Table 5.3, due to packet drops or delay, state information of the vehicle in front is not readily available. Therefore it is necessary to have some predictive scheme that estimates useful information within a dropped or delayed packet to counter such cases. Figure 5.2 depicts a period of 0.5 seconds from a congested network simulation to understand how the predictive control scheme has been implemented.

Predictive scheme during packet drop. This scheme is concerning how estimation is done when a vehicle misses a packet and delay is not considered here. Predictive scheme with delay is explained in the next section. In this scheme, when a vehicle drops a packet, trajectory planning is done based on the last available information received from the vehicle in front. This information is at least 100 ms old. A vehicle estimates state information of the vehicle in front based on the assumption that, the vehicle in front maintains a constant acceleration during the period a packet was dropped. This scheme is explained using example from figure 5.2.

- All vehicles receive packet from the vehicle in front which were transmitted at 0.1 s.

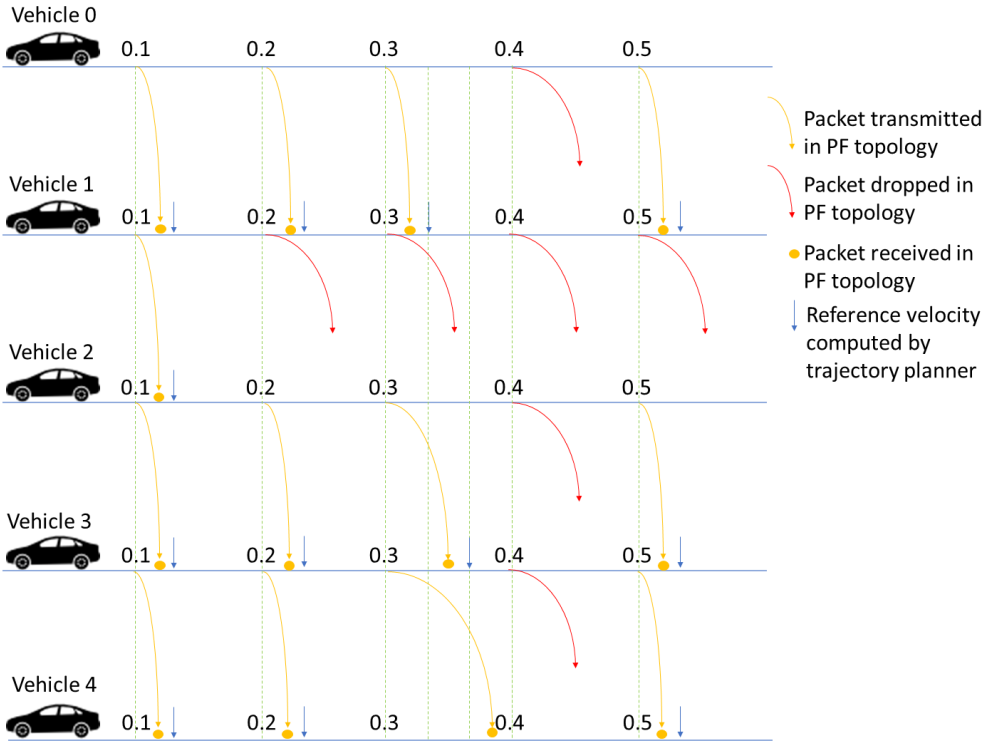


Figure 5.2: Predictive control scheme.

- Vehicle 2 does not receive the packet which was transmitted by vehicle 1 at 0.2 s. Vehicle 2 uses the vehicle state information that was transmitted by vehicle 1 at 0.1 s.

- The last received packet by vehicle 2 contained the state information of vehicle 1, $x_{1,0.1} = [q_{1,0.1}; v_{1,0.1}; a_{1,0.1}]$. Vehicle 2 assumes that vehicle 1 maintained a constant acceleration of $a_{1,0.1}$ from 0.1 s to 0.2 s.

- Vehicle 2 makes an estimation of vehicle 1's current position and velocity based on the assumption. The estimated displacement of vehicle 1 is given by $\hat{s} = ut + \frac{1}{2}at^2$ and estimated velocity as $\hat{v}_{1,0.2} = u + at$, where $u = v_{1,0.1}, a = a_{1,0.1}, t = 100ms$. The estimated state information of vehicle 1 at 0.2 s is given as $\hat{x}_{1,0.2} = [q_{1,0.1} + \hat{s}; \hat{v}_{1,0.2}; a_{1,0.1}]$.

This estimated state information is used to generate velocity trajectory. Since a platoon member anticipates reception of packets, how will the trajectory planner work until a packet is received? When to provide the new reference velocity to lower-level controller if a packet is received? These questions are addressed in the next section, taking into consideration delay.

Predictive scheme during packet delay. The previous section assumes that a packet is either received or not received and no delay is considered. However, table 5.3 shows that there is always some delay associated with the received packet. In scenario I, even when there are no additional vehicles, there is a maximum delay of 4.585 ms. The delay increases with more additional vehicles. An assumption is made to neglect all delays less than 5 ms. In other words, the velocity trajectory is always generated after a delay of 5 ms. Prediction scheme for delayed packets is required only if the packet delay is more than 5 ms.

In order to study the implementation of predictive scheme for delayed packets, the period between 0.3 s and 0.4 s in figure 5.2 is considered. The 2 dashed lines depict the intervals at which the lower-level controller receives new reference velocity from the velocity trajectory planner (i.e., sampling of velocity trajectory is done every $\frac{100}{3}$ ms) due to multi-rate interface between the layers.

- Vehicle 1 receives the packet from vehicle 0 sent at 0.3 s, with a delay in the range $(5, \frac{100}{3}]$ ms. Since vehicle 1 did not receive this packet within the first 5 ms, it generates velocity trajectory at 0.305 s by estimating vehicle state information of vehicle 0 and provide the sampled reference velocity to lower-level controller.

- Velocity trajectory was generated for the entire period until 0.4 s. This is done because vehicle 1 does not know whether it will receive any packet from vehicle 0 during this whole period.

- After vehicle 1 receives the delayed packet from vehicle 0 (delayed by $(5, \frac{100}{3}]$ ms), a new velocity trajectory is planned by vehicle 1 using the latest vehicle state information of vehicle 0.

- The new velocity trajectory is sampled at 0.3 s + $\frac{100}{3}$ ms to update the lower-level controller with new reference velocity. The same trajectory is sampled at 0.3 s + $\frac{200}{3}$ ms to provide reference velocity to lower-level controller.

Similarly, vehicle 3 receives the packet sent at 0.3 s by vehicle 2, with a delay in the range $(\frac{100}{3}, \frac{200}{3}]$ ms. After vehicle 3 receives the delayed packet from vehicle 2, a new velocity trajectory is planned by vehicle 3 using the new vehicle state information of vehicle 2. However, the lower-level controller of vehicle 3 is updated with the new reference velocity at 0.3 s + $\frac{200}{3}$ ms.

Vehicle 4 receives the packet sent at 0.3 s by vehicle 3, with a delay in the range $(\frac{200}{3}, 100]$ ms. However, this packet is of no use in the current 100 ms (from 0.3 s to 0.4 s) because vehicle 4 uses the same state information which was sent by vehicle 3 at 0.2 s. The delayed packet might be useful in the period starting from 0.4 s if the expected packet is dropped or delayed. In order to make use of the received packet in the current period, a possible approach would be to increase the number of loops interfacing the velocity trajectory planner layer and lower-level controller.

Chapter 6

Simulation and Results

In chapter 3, a co-simulation environment for validating platoon control algorithm was developed. Chapter 4 proposes and integrates a multi-rate loop control algorithm into the simulation environment. Several communication network scenarios based on traffic density on a highway were characterized in Chapter 5. In this chapter, the validation tool is simulated to see how the communication-aware platoon control algorithm performs in various communication network scenarios.

Acceleration profile, velocity profile and inter-vehicular distance of the platoon vehicles are analyzed to understand the behavior (reactiveness and comfortableness) of the proposed control algorithm in various scenarios. High reactiveness is required to respond to abrupt changes in the leader's motion and avoid any collision. Comfortableness is achieved when jerk is not very often. The simulation results are analyzed based on these to factors.

Each simulation is run for 1250 steps (corresponding to 125 s). Virtual reference velocity is provided to the lower-level controller of the vehicle 0 (leader). The virtual reference velocity as shown in figure 6.1 simulates acceleration, deceleration, constant velocity and sudden braking for vehicles (mini trucks). The virtual reference velocity has 3 different phases. The first phase (0 s to 94.5 s) represents increasing and decreasing velocity (between 10 m/s and 25 m/s) that resembles oscillating acceleration. Constant velocity (25 m/s), i.e. 0 acceleration is maintained in the second phase from 94.5 s to 105 s. The third phase, after 105 s, represents sudden braking (velocity decreases from 25 m/s to 0 m/s in 13 s) condition for trucks.

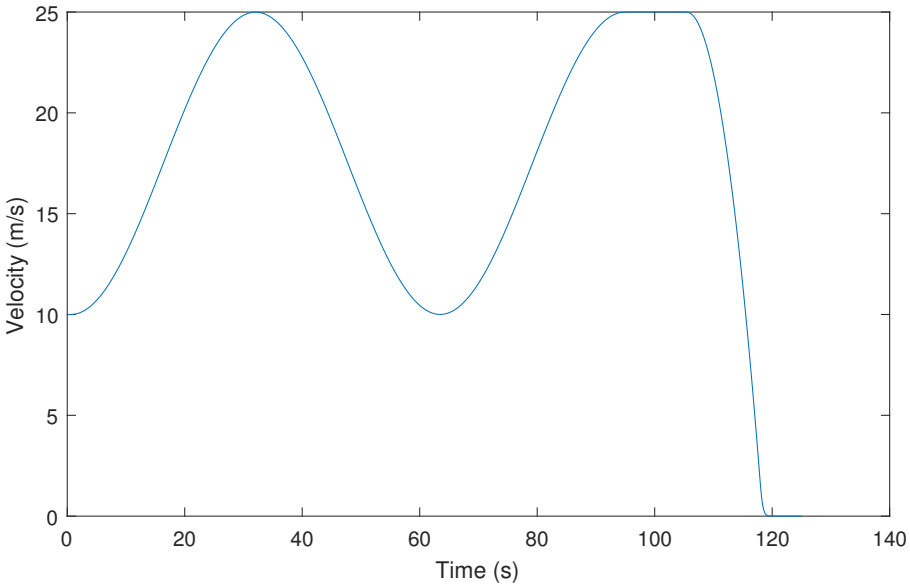
Table 6.1: Initial vehicle state information of platoon vehicles

Vehicle	**Position (m)**	**Velocity (m/s)**	**Acceleration (m/s^2)**
0	329	10	0
1	322	10	0
2	315	10	0
3	308	10	0
4	301	10	0

Table 6.2: Assumed parameters for platoon vehicles

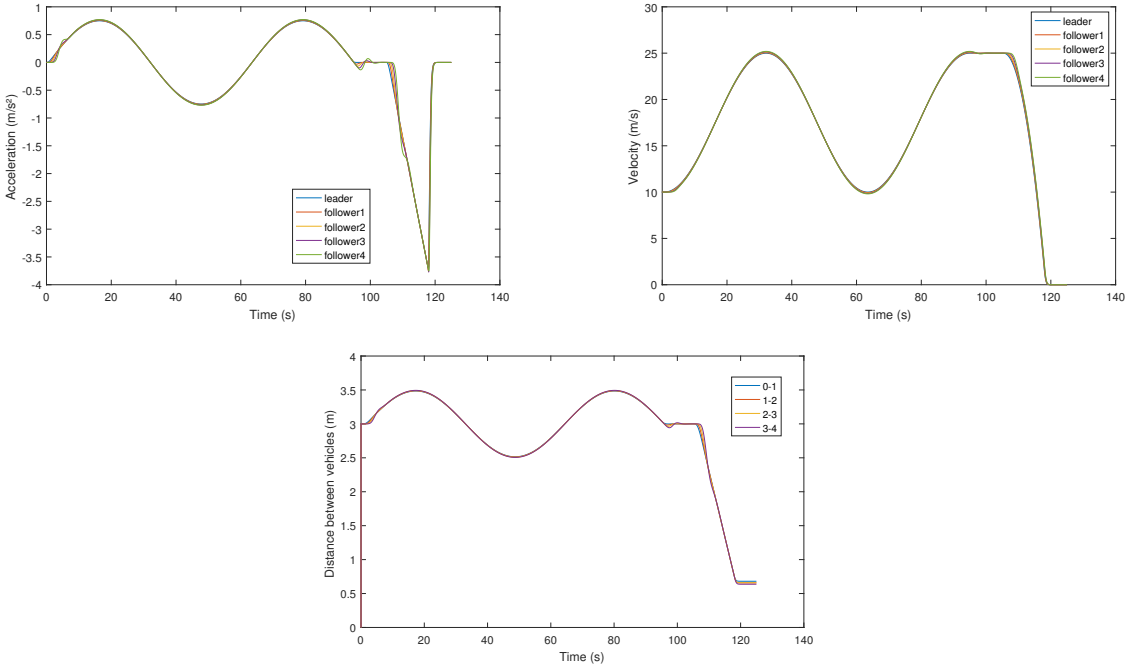
Parameters	**Assumptions**
Numbers of vehicles in platoon	5 m
Constant Buffer Distance	3 m
Headway time gap constant	0.5 s
Engine Dynamics time constant	0.3
Internal Dynamics Actuator time constant	0.3
Traffic type	Highway
Maximum velocity in highway for trucks	25 m/s
Maximum deceleration during braking	-3.5 m/s^2
Time taken for truck from maximum velocity to 0 m/s	13.5 s
Time period for V2V communication	100 ms
Worst case jerk realizable by platoon members	$\pm$0.35 m/s^3

Figure 6.1: Virtual reference velocity of vehicle 0



In this setting of simulation, each vehicle travels for an approximate distance of 2.2 km in the 3 km long highway. Traffic density is less towards the beginning and end of the highway. So the platoon is made to start from 300 m in order to have the worst case impact of communication imperfection on control algorithm. Initial conditions and assumptions are shown in table 6.1 and 6.2 respectively. Simulation results for all communication network scenarios have been shown in figure 6.2-6.10.

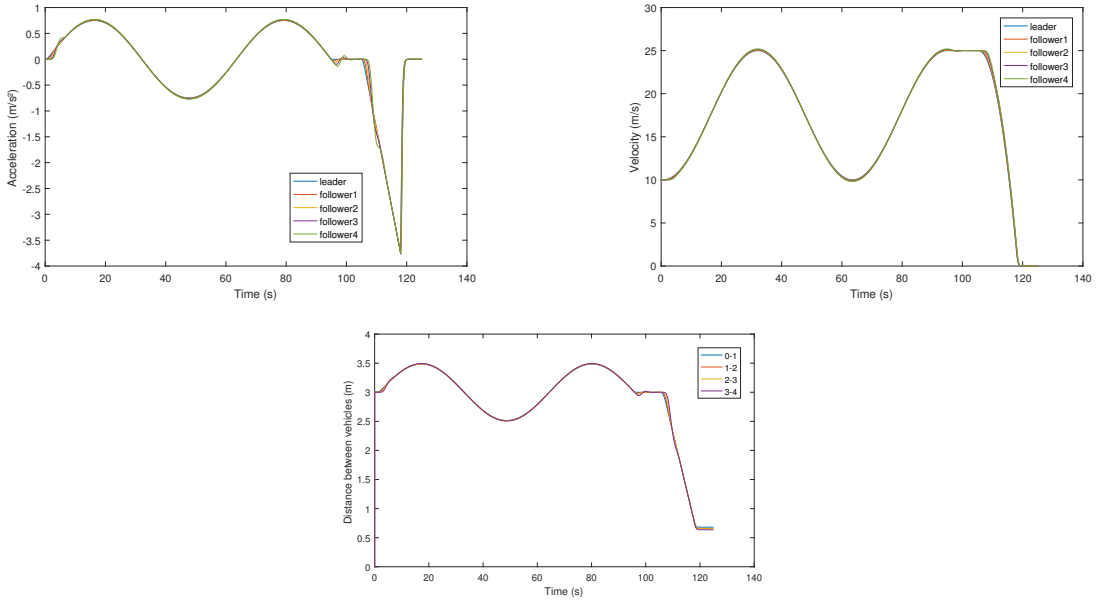
Figure 6.2: a) Acceleration profile b) Velocity profile, c) Inter-vehicular gap in scenario I.



Scenario I represents the situation when there are no additional traffic vehicles or when a separate channel is used for vehicle platooning. The channel is least congested in this scenario with no packet drops and minimum delay (table 5.3). Figure 6.2 shows the best case performance of the platoon control algorithm because no predictions are made. Few observations can be made from results of scenario I that are common in all other scenarios. All the vehicles were initialized with 0 acceleration and then suddenly the leader is made to increase velocity. During the first 7 s, vehicles experience some jerk. Maximum jerk is experienced by vehicle 4 is 0.214 m/s^3 whereas vehicle 1 experiences only 0.086 m/s^3. The lag induced by the low-level dynamics of the vehicle propagates through the platoon vehicles and affects the last vehicle the most. However, the effect of lag is taken care by feedback and rectification of multi-rate control loops which results in jerk. Similar behavior can be noted during 94 s - 102 s and 104 s - 112 s when the vehicles suddenly change velocity.

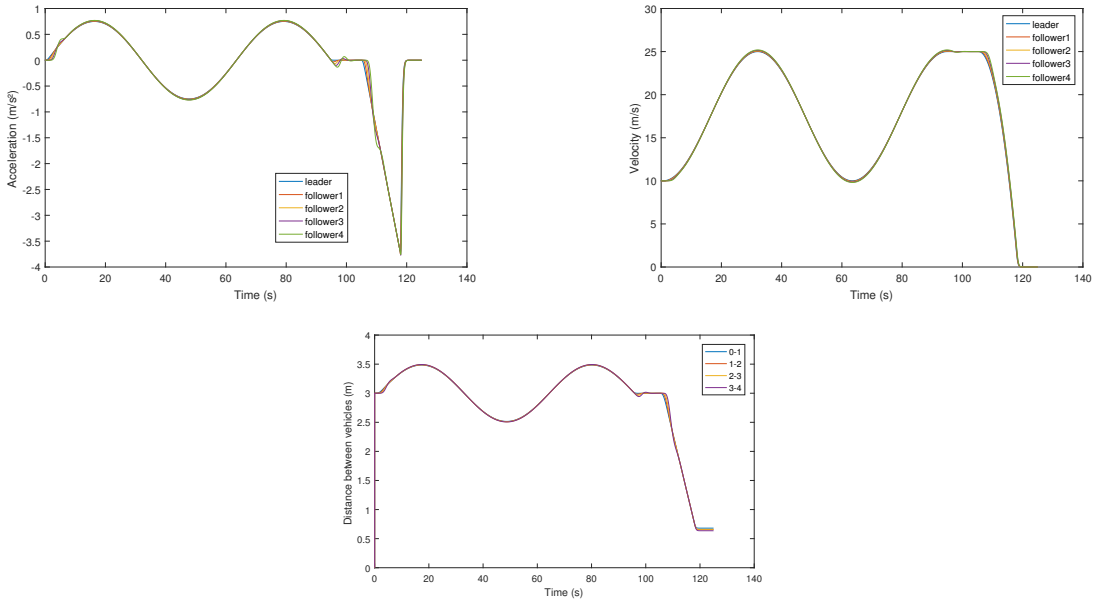
It can be seen that in all scenarios (figure 6.2-6.9), the inter-vehicular distance remains to be a positive value. This means that there is no collision detected between any of the vehicles in all scenarios. Until 94.5 s in scenario I, when oscillating input is given as the virtual reference velocity to vehicle 0, a deviation of ±48.5 cm and ±49.5 cm from constant distance buffer gap (3 m) is noticed between vehicle 0 - vehicle 1 and vehicle 3 - vehicle 4, respectively. Between 100 s and 105 s, inter-vehicular distance between the vehicles converge to 3 m. Largest deviation from constant buffer gap is experienced after sudden brakes are applied. Inter-vehicular distance reduces to 68.1 cm and 63.7 cm between vehicle 0 - vehicle 1 and vehicle 3 - vehicle 4, respectively.

Figure 6.3: a) Acceleration profile b) Velocity profile c) Inter-vehicular gap in scenario II.



Simulation results for scenario II with 5 platoon vehicles and 95 additional traffic vehicles is shown in figure 6.3. Behavior does not deviate from scenario I because maximum of only 3 consecutive packet misses are experienced (table 5.3).

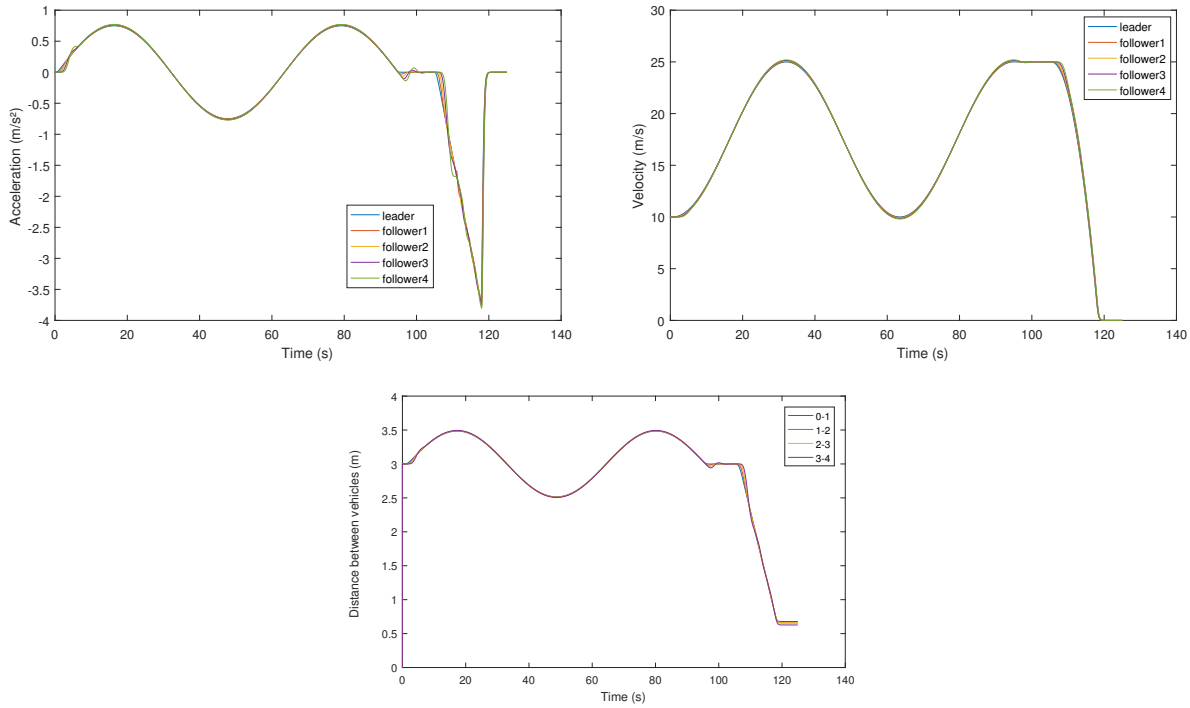
Figure 6.4: a) Acceleration profile b) Velocity profile, c) Inter-vehicular gap in scenario III.



Simulation results for scenario III with 5 platoon vehicles and 195 additional traffic

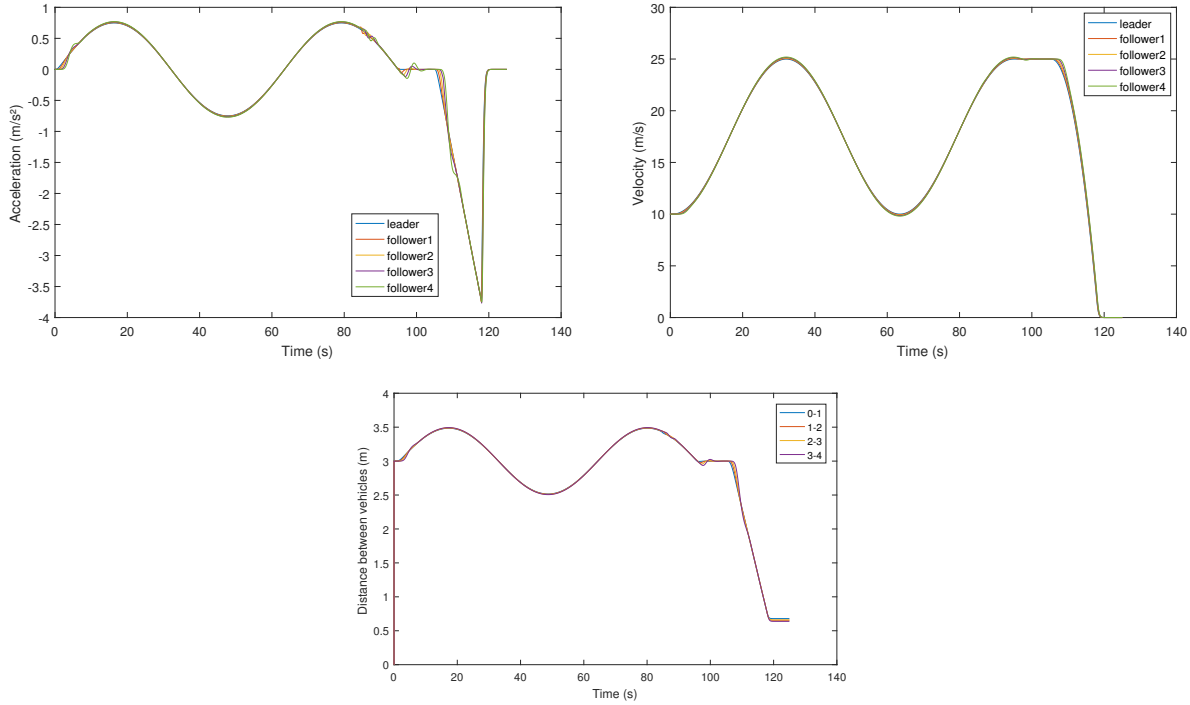
vehicles is shown in figure 6.4. Behavior remains to be consistent with scenario I because maximum of only 4 consecutive packet misses are experienced (table 5.3).

Figure 6.5: a) Acceleration profile b) Velocity profile, c) Inter-vehicular gap in scenario IV.



Simulation results for scenario IV with 5 platoon vehicles and 295 additional traffic vehicles is shown in figure 6.5. Behavior remains to be consistent with scenario I. However, a maximum of 7 consecutive packet misses are experienced in the link vehicle $0 -> $ vehicle 1 (table 5.3). Its effect is not visible prominently because the episode of misses occur from 110.5 s, during which all the vehicles are already decelerating (sudden braking). During this episode of 700 ms, the predictive scheme enables the platoon vehicles to assume that the vehicle in front is maintaining a constant deceleration. Also, an episode of 6 consecutive misses occur at 102.7 s. It does not produce any jitter because the platoon vehicles were moving with 0 acceleration and the predictive scheme enables them to maintain the same velocity. When vehicle state information is received after 6 misses, the acceleration is still 0. Therefore there is no jerk in this case.

Figure 6.6: a) Acceleration profile b) Velocity profile, c) Inter-vehicular gap in scenario V.



Simulation results for scenario V with 5 platoon vehicles and 395 additional traffic vehicles is shown in figure 6.6. There are variations from common behavior in the result. It can be seen that there is jitter in acceleration profile. From 83.4 s onwards, vehicle 1 experiences an episode of 13 consecutive periods of packet drops. Vehicle 1 predicts vehicle 0's vehicle state information during this period. At 84.7 s when vehicle 1 receives a packet from vehicle 0, jerk is induced into vehicle 1 due to sudden change in acceleration to avoid any collisions. This jerk is propagated along the platoon (as seen in figure 6.7) resulting in jitter in the acceleration profile of the platoon. A maximum jerk of 0.196 m/s^3 is experienced in this case by vehicle 4.

Figure 6.7: Jitter in acceleration profile in scenario V.

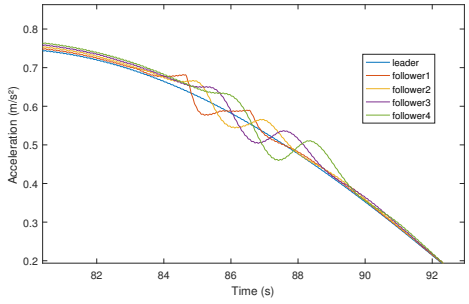
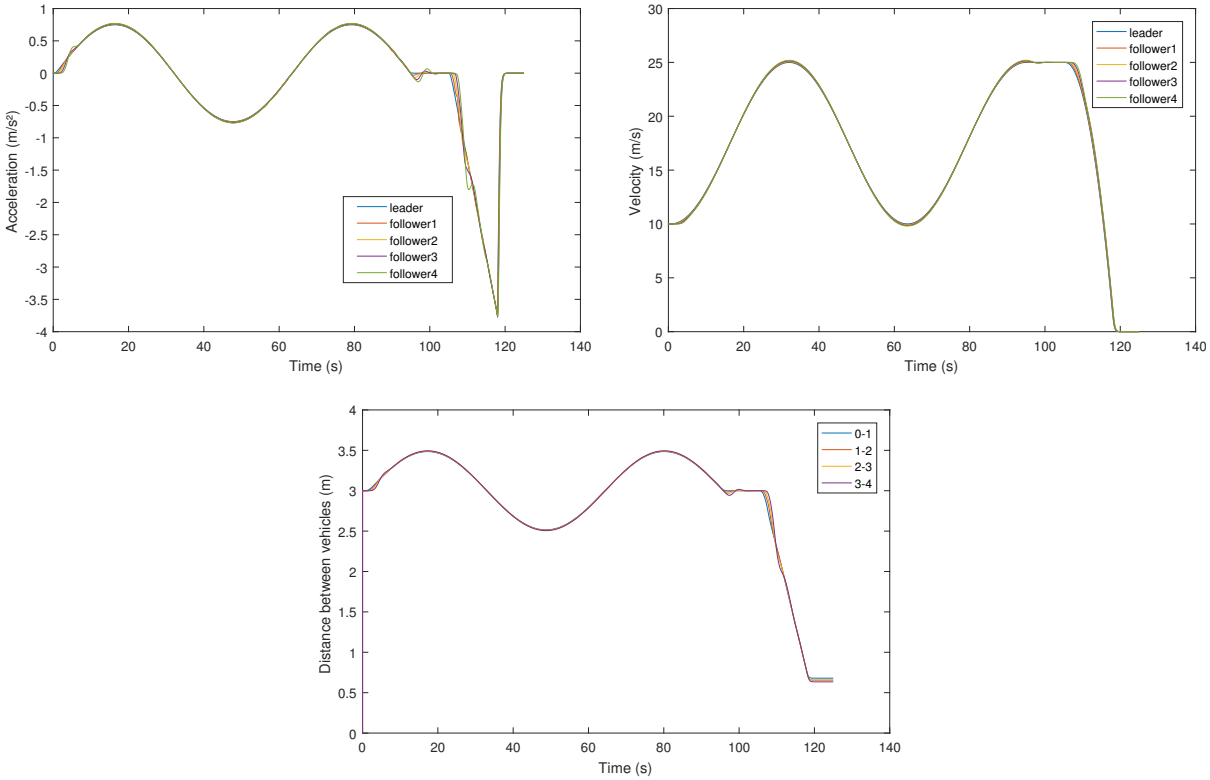
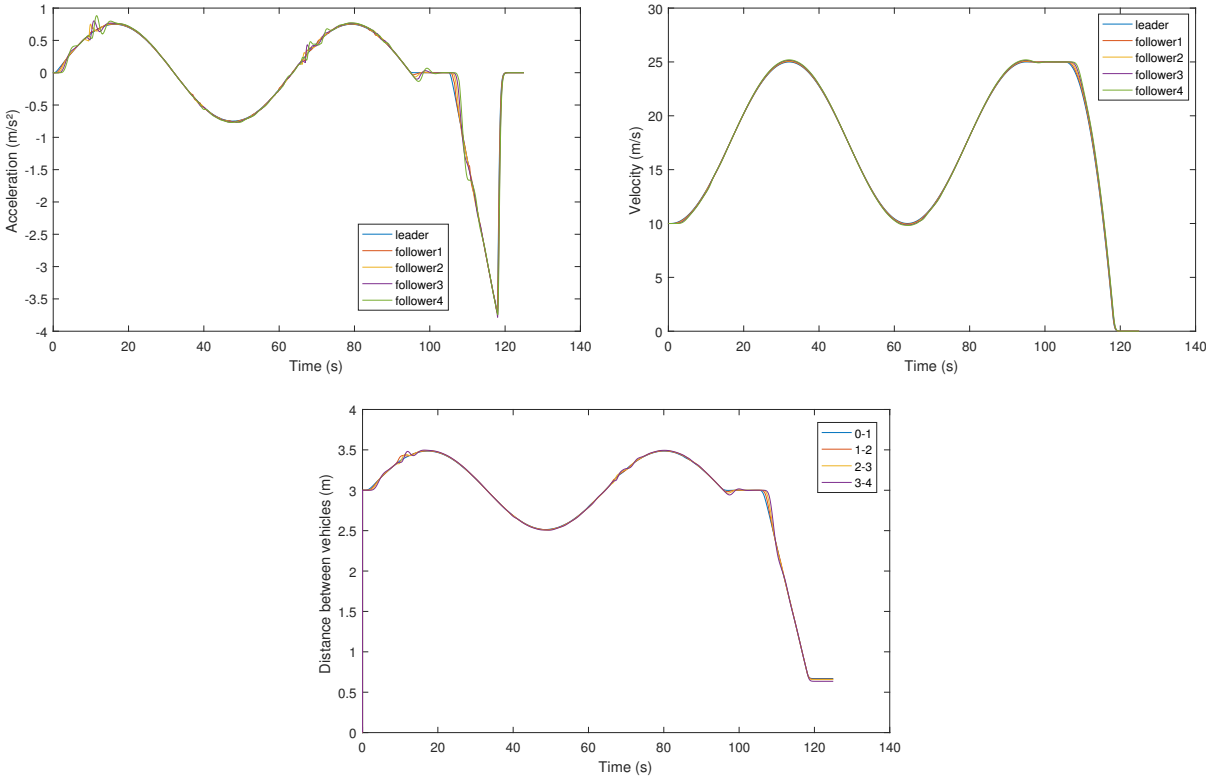


Figure 6.8: a) Acceleration profile b) Velocity profile, c) Inter-vehicular gap in scenario VI.



Simulation results for scenario VI with 5 platoon vehicles and 495 additional traffic vehicles is shown in figure 6.8. As seen in table 5.3, even though PRR is as low as 68.4% (vehicle $0 - >$ vehicle 1), jitter is not prominent in acceleration profile as compared to scenario V. This is because the maximum number of consecutive packet misses is only 8 in scenario VI. Moreover, this occurs at 106.6 s after all the vehicles applied sudden braking.

Figure 6.9: a) Acceleration profile b) Velocity profile, c) Inter-vehicular gap in scenario VII.



Simulation results for scenario VII with 5 platoon vehicles and 595 additional traffic vehicles is shown in figure 6.9. It can be seen that there is jitter in acceleration profile and noticeably more than in previous scenarios. Two episodes of jitter is observed in scenario VII. The first one is due to 17 consecutive periods of packet drops by vehicle 2 (from vehicle 1) from 7.8 s onwards. The second jitter is due to multiple vehicles having episodes of consecutive periods of packet drops simultaneously. From 65.8 s until 66.9 s, vehicle 1 experiences 11 consecutive periods of packet drops from vehicle 0. During the same interval, vehicle 3 and vehicle 4 experiences 9 and 6 consecutive periods of packet drops from vehicle 2 and vehicle 3, respectively. Thus, more than one vehicle contributes to the jitter in the second case.

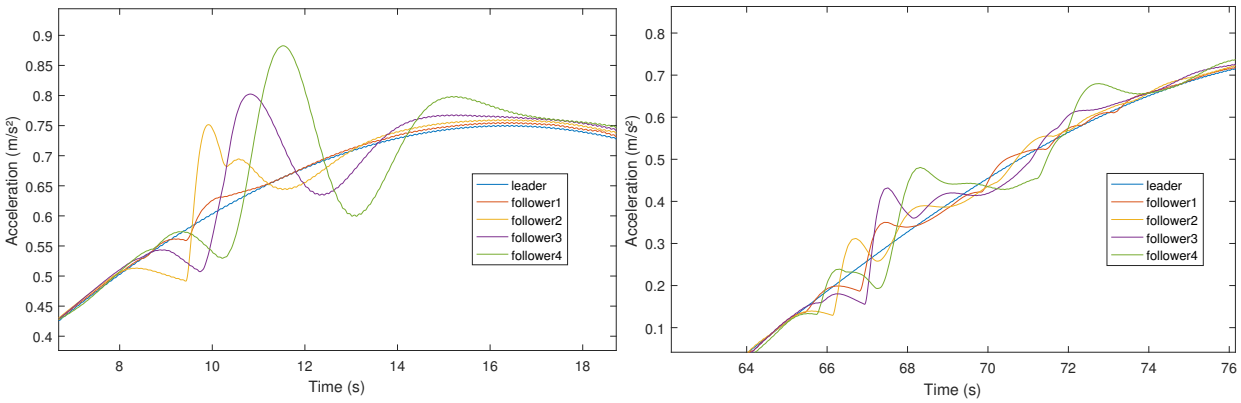


Figure 6.10: Multiple jitters in acceleration profile of scenario VII.

The worst jerk realized using the virtual reference velocity (figure 6.1) of vehicle 0 is during the sudden braking situation (-0.35 m/s^3). Jerks make transportation uncomfortable and loaded vehicles cannot realize very high jerks. Here, the worst case jerk that is realizable by a vehicle in the platoon is assumed to be $\pm$0.35 m/s^3 (table 6.2). From figure 6.7, the worst jerk experienced during jitter in scenario V is 0.196 m/s^3. However, in scenario VII, jerk experienced in the first and second cases shown in figure 6.10 are 0.486 m/s^3 and 0.495 m/s^3 respectively. These values are too large to be realizable. Even though the platoon control algorithm maintains the desired inter-vehicular distance between platoon members (reactiveness) in all scenarios, scenario VII is too congested and makes driving very uncomfortable. Hence, the proposed communication-aware platoon control algorithm with multi-rate control loop is validated in the first 6 communication network scenarios.

Chapter 7

Conclusion and Future Work

A validation tool has been developed to study the impact of communication imperfection on a proposed platoon control algorithm. The proposed platoon control algorithm uses an explicit model of lower-level control that takes into consideration low-level dynamics (engine and vehicle dynamics). The constraints of the lower-level controller and the problem of integrating it to the upper-level were investigated. A multi-rate control loop interface was used to integrate the upper and lower-level of the hierarchical control structure. Communication network scenarios on highway traffic were characterized to study PRR, packet delay and consecutive period of packet drops. Predictive schemes were also used in the platoon control algorithm to work in case packets are delayed or dropped.

The upper-layer of the hierarchical control structure uses a trajectory planner to control the inter-vehicular distance between platoon members. This can be replaced with sophisticated control techniques such as Model Predictive Controller (MPC) which can generate optimized trajectories for the vehicles. MPC can efficiently handle situations of packet drops and delays. Due to portability of the validation tool, MPC based platoon control algorithm can be easily validated using the developed tool. Furthermore, lower-level controller can be made more complex by including other factors relating to low-level dynamics of the vehicle such as gear ratio, weight, slope of the road, etc.

The hierarchical controller is limited to longitudinal control only. This can be extended for lateral control and other platoon functions such as merging, splitting and leaving. Control algorithm needs to be tested in rural and urban road conditions to ensure its robustness. Also crucial conditions such as uphill-downhill and round-abouts need to be satisfied. SUMO has limitations with respect to precision of decision and its graphical interface. Commercial traffic simulator such as VISSIM can replace SUMO in the validation tool to implement complex conditions. The future scope of this thesis is not limited to the implementations done so far. The validation tool is not limited to validating control algorithm for vehicle platooning alone. Other vehicular applications can also be simulated using the tool by changing the settings accordingly.

Bibliography

[1] R. Rajamani, "Longitudinal Control for Vehicle Platoons," in Vehicle dynamics and control, USA: Springer Publications, 2016, ch. 7, pp. 187-216. 3, 4

[2] T. Esbensen, B. T. Jensen, M. O. Niss, and C. Sloth, "Comparison of Longitudinal Control of Vehicles with and without Inter-vehicle Communication," Section for Automation and Control, Department of Electronic Systems, Aalborg University, December 2007. 4

[3] S. Shladover, C. Nowakowski, X. Lu, R. Ferlis, "Cooperative Adaptive Cruise Control (CACC) Definition and Operating Concepts ," Transportation Research Record: Journal of the Transportation Research Board, Vol. 2489, DOI: 10.3141/2489-17. 2, 4

[4] https://www.nsnam.org/docs/tutorial/html/

[5] M. Bernardo, P. Falcone, A. Salvi, S. Santini, "Design, Analysis, and Experimental Validation of a Distributed Protocol for Platooning in the Presence of Time-Varying Heterogeneous Delays," IEEE Transactions on Control Systems Technology, Vol. 24, No. 2, March 2016. 4

[6] S. Oncu, " String stability of interconnected vehicles : network-aware modelling, analysis and experiments," Eindhoven: Technische Universiteit Eindhoven DOI: 10.6100/IR762788, 2014. 4

[7] S. Oncu, N. Wouw, H. Nijmeijer, "Cooperative Adaptive Cruise Control: Trade-offs Between Control and Network Specifications," 14th International IEEE Conference on Intelligent Transportation Systems Washington, DC, USA. October 5-7, 2011. 4

[8] O. Gehring, H. Fritz, "Practical Results of a Longitudinal Control Concept for Truck Platooning with Vehicle to Vehicle Communication," IEEE Conference on Intelligent Transportation System, Boston, MA, USA, 1997. 4

[9] A. Saxena, "Design and Analysis of Control Strategies for Vehicle Platooning," 19th International IEEE Conference on Intelligent Transportation Systems (ITSC 2016), Rio de Janeiro, Brazil, November, 2016. 4, 5, 6, 7, 8, 17, 19, 28

[10] http://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-Wiki 10

[11] R. Rajamani, S. Choi, J. K. Hedrick, R. Prohaska, P. Kretz, "Design and Experimental Implementation of Longitudinal Control for a Platoon of Automated Vehicles," J. Dyn. Sys., Meas., Control 122(3), Vol. 122, pp. 470-476, September 2000. 4

[12] S. Li, K. Li, R. Rajamani, J. Wang, " Model Predictive Multi-Objective Vehicular Adaptive Cruise Control," IEEE Transactions on Control Systems Technology, Vol. 19, No. 3, May 2011. 4, 5, 17

[13] A. G. Ulsoy, H.Peng, M. Cakmakci, "Cruise and Headway Control," in Automotive Control Systems, Cambridge University Press, 2012, ch. 12, pp 213-231. 4, 8

[14] "Adaptive Cruise Control System Overview," 5th Meeting of the U.S. Software System Safety Working Group April 12th-14th 2005 @ Anaheim, California USA. 3

[15] D. Kachan, "Integration of NS-3 with MATLAB/Simulink," Sweden, Lund University, 2010. 10

[16] G. Pathak, "Investigate various VANET Communication and TraCI Simulator Interface Solutions," Eindhoven: Eindhoven University of Technology, 2016. 10, 11

[17] S. Y. Han, Y. H. Chen, A. Abraham, "Decentralized Longitudinal Tracking Control for Cooperative Adaptive Cruise Control Systems in a Platoon," IEEE International Conference on Systems, Man, and Cybernetics, 2013.