Technical Report

Fail-Operational and Fail-Safe Vehicle Platooning in the Presence of Transient Communication Errors

Shahriar Hasan

October 2, 2020

Revised March 21, 2022

School of Innovation, Design and Engineering
Mälardalen University
Västerås, Sweden

ISBN 978-91-7485-515-9
Published by Mälardalen University, Västerås, Sweden
Date of publication October 2, 2020 (as licentiate thesis); date of revised version March 21, 2022 (as technical report)

Table 1: Overview of the revisions

Date	Content of the revision
2020-10-02	Published as a licentiate thesis with ISBN numbers 978-91-785-477-0, ISSN 1651-9256, MDH thesis no 294.
2020-10-23	The thesis was withdrawn and the licentiate seminar was postponed since the Ethics committee at Mälardalen University wanted to investigate a statement from an external researcher regarding citations of related work. By 2021-03-12, the Ethics committee had finalized their investigation and the legal officer decided to leave the case without further action. Due to the long delay, it was decided to not move further with the licentiate seminar and instead focus on the doctoral thesis.
2022-03-21	It became apparent that the thesis was already cited by other researchers while it was available in 2020. Hence it was decided to revise the thesis to be a technical report with ISBN numbers 978-91-7485-515-9. To this end, the statement “in this thesis” was replaced by “in this report”, or “in this research work”, or “in this study” throughout the text. In addition, the Acknowledgments were revised to fit a technical report rather than a thesis and some revisions were made according to: Updated the captions in Figure 2.1 (page 10) and Figure 5.1 (page 42) to highlight that an updated version of the state machine will be made available soon (submitted for reviewing in Feb 2022). In addition, on page 26, the statement “proposed by Rajamani in [19]” is replaced by “presented by Rajamani in [19]”.

Abstract

Recent advances in wireless technology facilitating Vehicle-to-Vehicle (V2V) communication has paved the way towards a more connected and Cooperative-Intelligent Transportation System (C-ITS). It has unveiled the possibility of many services which are anticipated to make the road transport ecosystem safer, cleaner, and more sustainable. Platooning is one such application that is expected to soon appear on the roads. In platooning, a group of connected and highly automated vehicles follow a lead vehicle with short inter-vehicle distances. They adapt their speed, acceleration, steering angle, etc., with the help of on-board sensors and inter-vehicle communications. Due to the highly automated driving and the very short inter-vehicle distances required to achieve fuel-efficiency, platooning is a complex and safety-critical system of systems. As a result, the consequences of component or system failure can endanger human life, cause damage to property, or the environment. Given that V2V communication is subject to packet losses due to interference, path loss, fading and shadowing, it is usually desirable to maintain a sufficient level of platooning functionality without compromising safety also during periods of transient errors. Moreover, a platoon can experience different sensor failures, permanent hardware/software failures, or a suddenly appearing road hazard, e.g., a moose. The platoon should, therefore, also be capable of dissolving and transitioning into a fail-safe state by performing emergency braking, safe stop, or manual handover without causing any harm to the equipment, people, or to the environment. This research work focuses on incorporating fail-operational mechanisms in platooning in a fuel-efficient and safe way, even in the presence of transient errors and enable transition into a fail-safe state in the event of an emergency. To this end, a platoon runtime manager is proposed, which monitors the channel quality and keeps the platoon operational in cases of temporal failures by degrading the platoon performance to the level at which it will remain acceptably safe. Simulation results demonstrate that the runtime manager can avoid collisions in the platoon and still maintain fair performance in terms of fuel-efficiency by either adjusting the inter-vehicle distances or switching to a different controller during runtime. Furthermore, two emergency braking strategies, namely Synchronized Braking and Adaptive Emergency Braking, are proposed to address the emergency events that can arise while platooning. These braking strategies are compared to several state-of-the-art braking strategies in terms of their ability to avoid collisions, and the distance traversed by the lead vehicle. Simulation results show that Synchronized Braking and Adaptive Emergency Braking strategies can ensure fail-safe platooning while the other braking strategies fail to do so. Moreover, a simulation tool named PlatoonSAFE has been developed to facilitate the evaluation of fail-operational and fail-safe mechanisms in platooning under realistic traffic, vehicle dynamics, and communication scenarios.

Sammanfattning

De senaste framstegen inom uppkopplade automatiserade fordon har stor potential att tillhandahålla ett säkrare, renare och mer hållbart intelligent transportekosystem. En grupp trådlöst sammankopplade självkörande fordon kan följa tätt efter ett ledande fordon och tillsammans bilda ett fordonståg. Att ingå i ett fordonståg möjliggör för de efterföljande fordonen att spara upp till 20% bränsle då det aerodynamiska motståndet minskar. Eftersom 90% av alla dödsfall som sker i trafiken kan härledas till mänskliga brister, har automatiserad körning också potential att minska antalet trafikolyckor avsevärt genom att i olika grad avlasta föraren. Trots dessa enorma samhälleliga, miljömässiga och ekonomiska fördelarna med fordonståg dröjer automatiseringen på grund av risken för säkerhetsproblem. Detta beror på att fordonståg är ett komplext system av sammankopplade säkerhetskritiska system som kan äventyra människoliv om det fallerar. Denna rapport fokuserar därför på att integrera feltoleranta och felsäkra lösningar i fordonståg för att tillhandahålla höga säkerhetsnivåer både för fordonstågen och för omgivande medtrafikanter även då kommunikationsproblem uppstår. Om ett kommunikationsfel eller en störning upptäcks kan säkerhetsavståndet mellan fordonen automatiskt ökas eller så kan det sätt på vilket fordonen kommunicerar ändras och anpassas. Vid större systemfel orsakade av maskin- och programvarufel eller någon annan fara aktiveras olika nödbromsningstekniker som undviker kollision mellan fordonen. Exempelvis så kan det sista fordonet uppmanas att bromsa först varefter de andra följer en i taget, eller så kan fordnomen kommunicera med varandra så att alla börjar bromsa samtidigt med full kraft. Resultatet av arbetet innebär att fordonståg kan användas på ett säkert sätt trots kommunikationsproblem, och också att det finns möjlighet att stoppa fordonståget om allvarligare problem uppstår eller om det plötsligt finns något hinder på vägen.

Acknowledgments

The work presented in this report has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska - Curie grant agreement No 764951[1].

[1] https://immersafe-itn.eu/

Contents

Chapter 1

Introduction

1.1 Motivation

In the last few decades, Intelligent Transportation System (ITS) has undergone revolutionary changes dictated by road safety and environmental concerns. Currently, 27% of EU's total carbon emission is caused by the transport industry, of which 25% comes from trucks and lorries, and 44% comes from the cars [1, 2]. The newly approved regulation by the European Parliament agrees upon a 30% carbon reduction target for heavy-duty vehicles by 2030 [3]. *Platoon*-based driving has the potential to help to attain this goal by minimizing fuel consumption. In platooning, a group of highly automated vehicles with common goals gather behind a Lead Vehicle (LV) and obey its speed, acceleration, position, steering angle, etc., with the aid of on-board sensors and wireless vehicular communications, Figure 1.1. The advantages are fuel-efficiency, driver offload, road throughput enhancement, improved safety, etc. Bonnet *et al.* [4] reported 21% fuel consumption reduction for the following truck in a two-vehicle platoon with an inter-vehicle gap of 10 meters, thanks to the reduction of aerodynamic drag. Furthermore, in the literature [5, 6], it has been shown that platooning can significantly improve safety by minimizing brake lag, avoiding front- and rear-end collisions, preventing human errors, and restricting non-platooning vehicles from cutting in; hence reducing the possibility of collisions with them.

As appealing as the benefits of platooning might sound, its challenges are equally complex. A platoon is a tightly coupled system of computing, communication, and control technologies. Jia *et al.* characterizes a platoon as a Vehicular Cyber Physical System (VCPS) [5], and divides VCPS into two planes, namely cyber plane and physical plane. The physical plane is described by the physical dynamics of the platoon, whereas the cyber plane is the vehicular network. The performance of vehicular networks in terms of communication latency, packet loss, etc., significantly affects the platooning actions such as platoon forming, maintenance, merging, splitting, and braking. On the other

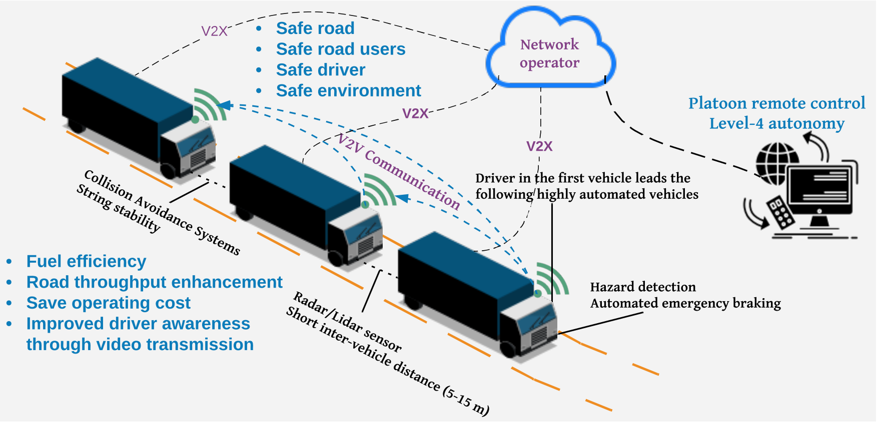


Figure 1.1: A platooning scenario: its benefits, challenges, and the underlying technologies.

hand, these platooning actions affect the communication aspects as well. It is therefore equally important to consider both planes as well as the tight coupling between them while analyzing the performance of a platoon. The work presented in this report considers this dependency while addressing the identified problems.

In pursuance of accomplishing the benefits of platooning, a short inter-vehicle gap is of the essence, and this is what makes platooning a critical system because the response times of the involved systems need to meet the real-time requirements to safeguard lives, nature, the fate of a business model etc. Axelsson [7] says, the degree to which the platooning benefits can be attained is directly proportional to the inter-vehicle distance maintained between the vehicles. The reduction of inter-vehicle gap raises the issues related to platoon safety. This challenge is further aggravated by transient errors caused by the unreliable wireless communication and component failures, e.g., sensor and actuator failure, as well as the in-vehicle communication network failure. This research work aims at handling the hazardous situations that can arise in a platoon due to the need to keep a short inter-vehicle gap.

In platooning, a significant form of hazard is when a platooning vehicle runs into its preceding vehicle due to sudden emergency braking by the lead vehicle [8, 9]; from this point onward, this is denoted as rear-end collision. A rear-end collision can also happen without emergency braking. As an illustration, let us consider that a platoon is maintaining 5 m gap in the motorway and the leader slightly decelerates due to the change in speed limit. A few consecutive packet losses experienced by any of the Following Vehicles (FVs) can cause rear-end collisions as the inter-vehicle distance is short and the on-board radar sensor has substantial delay. Although the platooning vehicles can have backup drivers, they will not have enough time to react in the events of emergency while driving with short gaps. Several studies in the literature [9, 10, 11, 12] suggest that the backup driver can take over the control of the vehicle and perform emergency braking, or steer out to avoid collision with the front vehicle. However, the typical reaction time of a human driver is much higher than the required time gap in platooning applications [13, 14]. When the driver has no driving task and needs to react to a sudden event, the reaction time can go up to 10 s [15, 16]. Hence, automated emergency braking is required to bring the platoon into a complete standstill without rear-end collisions in such hazardous situations. There can be other types of platoon-related hazards such as driving off the road by steering to avoid collisions [9], cut-in situations, etc. From a broader perspective, failing to meet the objectives of a system can also be considered as hazard [7]. In the context of platooning, failing to reduce fuel consumption and ensure safety due to software and/or hardware failure can be deemed as hazard [17].

A platoon is regulated by an intelligent system that is called a controller. A controller ensures that a group of vehicles move at consensual speed while maintaining the desired inter-vehicle distance between the adjacent vehicles [18]. Radar and sensor-based vehicle controllers such as Cruise Control (CC) and Adaptive Cruise Control (ACC) are already available in modern cars. In CC, the drivers are required to set the desired speed which the vehicle can maintain until interrupted by the driver manually. In ACC, a vehicle can follow its preceding vehicle by looking at the relative speed, and distance measured by a radar or Lidar sensor. In addition, ACC can act as CC also when there are no vehicles ahead. The desired distance to the front vehicle usually grows proportionally with the increase of speed to ensure safety and stability in ACC [19]. These sensor-based systems are less fuel-efficient as the vehicles are required to maintain a high inter-vehicle distance. The limitations of such systems triggered the necessity of Cooperative Adaptive Cruise Control (CACC) where Connected and Automated Vehicles (CAVs) can share vehicle kinematic parameters to form a string of vehicles, referred to as *CACC vehicle string* in this report. Both ACC and CACC act on the Constant Time Gap (CTG) policy in which the time gap between two vehicles changes with the change of speed. The California PATH project [20, 19] is one of the first of its kind to incorporate V2V communications to demonstrate the benefits of CACC in terms of safety, fuel efficiency, and string stability. Within this project, a comparative analysis between ACC and CACC was conducted through simulations [21]. In the PATH CACC study, Nowakowski *et al.* [22] reported that CACC could reduce the following mean gap from 1.4 seconds to 0.6 seconds in comparison to manual driving while the vehicles are driving at a speed of 104 kmh^{-1}. Like the CACC vehicle string, a platoon also requires a controller and V2V communications. A CACC vehicle string offers longitudinal control only, i.e., the driver still requires to steer the wheel and monitor the surrounding environment, whereas, platooning offers both lateral and longitudinal control [23]. Moreover, a platoon controller utilises the Constant Distance Gap (CDG) policy in which the vehicles maintain the same gap regardless of the change in speed.

The information required for lateral and longitudinal control in platooning is disseminated through Platoon Control Messages (PCMs). The European Telecommunications Standards Institute (ETSI) and the European Committee for Standardization (CEN) delivered the first set of standards for C-ITS in 2014. The C-ITS standards define a set of day-one applications that include cooperative awareness applications and road hazard applications. To ensure situational awareness, the LV and/or the other platooning vehicles periodically broadcast Cooperative Awareness Messages (CAMs) [24] that contain necessary parameters for lateral and longitudinal control. When a hazard of common interest occurs, the LV broadcasts Decentralized Environmental Notification Message (DENM) [25] for the duration of the event instructing the FVs to react to the emergency, e.g., by performing emergency braking, manual handover, platoon dissolution etc. A more detailed description of the C-ITS standards and PCMs are provided in Section 3.3.

1.2 Scope of the Study

John C. Knight defines a *safety-critical system* from the *consequences of a failure* point of view [26]. If the consequences of a failure in a system are considered unacceptable because they could endanger human life, lead to property damage or environmental damage, then it is a safety-critical system. From this point of view, platooning is a safety-critical system of systems because the harm posed by platoon related hazards goes beyond one single vehicle, and it could cause all possible safety-related dangers. Furthermore, any safety-critical system should have fault-tolerant, fail-safe, and fail-operational features to mitigate the effects of the hazards or failures [7]. In the context of a safety-critical system, a *fault* is a defect within the system that leads to an *error*. When a system cannot deliver the correct service, this is called a *failure*. The presence of an error can cause a system failure. Should any hardware or software faults take place, a platoon should be *fault-tolerant* to mitigate the effects of the fault and prevent the fault from leading to possible failures. As an illustration, a platoon can change its communication topology when faults happen in the communication system by continuously monitoring packet losses.

This study focuses on devising solutions towards fail-safe and fail-operational vehicle platooning by attempting to tackle the platoon-related hazards. A *fail-safe* state in this context implies that the platoon will transition into a known condition that is safe/secure, e.g., safe stop by performing emergency braking, in the event of an irrecoverable failure, or emergency. A fail-safe state should prevent any harm to the equipment, people, or the environment. The intention is to keep the platoon in safe mode even after the failure. In a *fail-operational* state, the platoon should keep certain critical functionalities operational in the event of a failure. For instance, the system degrades platoon performance in terms of fuel-efficiency during temporary communication failure. Performance degradation means increasing the inter-vehicle gap, and/or transitioning from the PLATOON mode to the CACC or ACC mode. Fail-safe and fail-operational states are intended to ensure that a system behaves predictably as a consequence of failures by taking proactive measures [27]. Both fail-safe and fail-operational features are indispensable for the design of platooning system architecture. Still many platoon-related studies in the literature often emphasize on one and ignore the other [28].

This study accounts the temporary communication errors that the vehicles can experience when cruising in a platoon or performing emergency braking. To this end, a quantitative analysis of the communication latency under various network loads has been carried out. Moreover, the state-of-the-art ACC and CACC controllers have been studied to investigate their efficacy in driving a platoon. The controllers have been simulated with varying inter-vehicle distances, deceleration rates, and network loads to understand the correlation between these parameters.

To tackle the fail-safe platooning problem, two emergency braking techniques, namely *Synchronized Braking (SB)* and *Adaptive Emergency Braking (AEB)*, have been proposed. The SB strategy allows the platooning vehicles to brake at a very high deceleration rate using communication to coordinate the braking. The AEB strategy instead leverages the communication latency incurred by the platooning vehicles to perform deceleration at a lower rate upon reception of a DENM, and later brakes at full-deceleration once the vehicles receive a DENM Acknowledgement (ACK). These braking strategies are compared with the state-of-the-art emergency braking protocols in terms of collision avoidance and the distance traversed by the lead vehicle before a complete stop. The rationale behind choosing these two metrics is that in a fail-safe state the platooning vehicles are required to avoid rear-end collisions, and the lead vehicle should stop fast to circumvent the obstacle that triggered the emergency braking.

As the platooning vehicles are vulnerable to failures such as temporary communication failure, sensor failure, etc., the Runtime Manager (RM) concept [29] is used that monitors the packet losses within the platoon during runtime, and degrades the platoon performance by increasing the gap and/or switching to the ACC controller that does not require wireless communication; the degradation is done based on several safety contracts that is defined and implemented in the RM. Thus, the RM ensures fail-operational platooning by keeping certain functionalities operational so that the platoon can either recover from the failure if it is transient or make sufficient inter-vehicle gaps to facilitate safe stop by dissolving the platoon in case of permanent failures or road hazards. In case of a fault in the communication system due to a few consecutive packet losses, the RM switches between PLATOON and a set of CACC controllers that have different communication topologies and finally, ACC in case of total failure in the wireless communication.

All the studies performed in this report have been validated through computer simulations. A platooning simulator was required that simulates realistic traffic model, vehicle dynamics, vehicle control system, and wireless vehicular networks to support cooperative driving. Fortunately, the Plexe [30] simulator provides all these features. The simulator has further been extended to implement various Cooperative Emergency Braking (CEB) techniques and the Runtime Manager concept. The simulation tool is referred to as the *Platoon-SAFE* simulator.

1.3 Outline of the Report

This report comprises ten chapters. A brief overview of each chapter is provided below.

In Chapter 2, the research problem is formulated, and the goals of the study are defined. A state machine is proposed that shows the transition between different fail-operational and fail-safe states based on the communication quality. Based on the problem formulation, the research questions are determined. Then the contributions of this report have been briefly described that answer the research questions. Finally, the research process that is followed to attain the research goals is presented.

The background required for assessing the recent advances in vehicular communications and automated driving is provided in Chapter 3. In this chapter, the reader is first familiarized with the SAE's levels of driving automation. This is followed by a comprehensive discussion on the CACC and PLATOON concepts. Then a short overview of the Vehicle-to-Everything (V2X) communication standards in light of the ETSI ITS-G5 protocol stack is given. Moreover, the existing Medium Access Control (MAC) protocols available for Vehicular Ad hoc Network (VANET) applications are discussed.

Chapter 4 presents state-of-the-art works. First, the concept of PLATOON, CACC, and ACC controllers are described that is used as benchmarks throughout this report. In addition, simulation results are presented to compare and analyze these state-of-the-art controllers. This is followed by a discussion on the Collision Avoidance Systems (CASs) proposed in recent works. Then the runtime manager concept and the contemporary works related to contract-based safety assurance methods are described. Chapter 4 ends with an overview of existing VANET simulation tools.

Chapter 5 presents the safety applications that are proposed in this report to achieve fail-operational and fail-safe states in platooning. The platoon safety applications comprise the RM and CEB modules. In this chapter, the concept of the Runtime Manager and the emergency braking strategies are described. In addition, a state-machine is presented that demonstrates how a platooning vehicle maintains the fail-operational state depending on the perceived communication quality to the preceding vehicle and the LV, and fail-safe state when it comes across a road hazard.

The PlatoonSAFE simulator is presented in Chapter 6. In this chapter, the safety contracts are elaborately described that the runtime manager uses to keep the platoon fail-operational in the events of transient communication errors. Moreover, the implementation details of the cooperative emergency braking techniques are given. Chapter 6 also works as a tutorial for a user who intends to use the PlatoonSAFE tool. The aim is to facilitate future extensions and induce the reproducibility of the results presented in this report.

In Chapter 7, the runtime manager is evaluated to assess its efficacy in maintaining fail-operational platooning. First, a detailed description is provided on how to classify a *good*, *fair*, and *poor* communication quality based on the packet losses perceived by the platooning vehicles. Later on, the research findings obtained through simulation analysis are presented to demonstrate how the runtime manager can degrade the platoon performance temporarily to keep it fail-operational.

The SB and AEB strategies are compared with the state-of-the-art braking strategies, and the findings are presented in Chapter 8. Rigorous simulations have been performed to understand which of the braking methods are more suitable for transitioning a platoon into the fail-safe state in the events of hazards.

Chapter 9 presents the practical aspects in terms of speeds, inter-vehicle distances, and stopping distances that are required to be considered in platooning. These practical aspects are extracted based on the evaluation results presented in Chapter 7 and 8.

Chapter 10 finally concludes the report and provides possible directions for future work.

Chapter 2

Research Synopsis

"Self-driving trucks are anticipated to appear on the roads in the next couple of years due to the straightforward nature of hub to hub good transport on freeways", says Waymo's CEO [31]. This would take the practical realisation of vehicle platooning a step closer to the public highways. The platooning vehicles are required to maintain short inter-vehicle distances to reduce fuel consumption by minimising aerodynamic drag and enhance road throughput. This raises the issue of the platoon safety and that of other road users. The challenge is further amplified due to the transient errors caused by the unreliable wireless communication, which is the basis for platoon coordination; and the error can happen both while the platoon is cruising, or in the events of emergency braking. Fail-operational and fail-safe measures are proposed in this report to tackle these challenges.

2.1 On Fuel-efficient and Safe Platooning

Figure 2.1 presents the states of platooning operations proposed in this report. The description of the state machine demonstrates the transitions between fail-operational and fail-safe states under various levels of transient communication errors. The purpose is to tackle the challenges caused by connectivity errors and the appearance of sudden road hazards. This research work aims to determine when to transition between the states, what controllers apply to the states that regulate the inter-vehicle gaps and communication topology, and finally, how to decide whether the experienced communication latency is sufficient or not. To do this, a framework to evaluate the platoon performance is also considered.

The state machine comprises five platooning states. A description of the states is provided below:

- *Fuel-efficient and safe platooning*: This is the platooning state in which the vehicles maintain short inter-vehicle distances using the CDG strategy to enable sufficient fuel-efficiency and provide the required level of

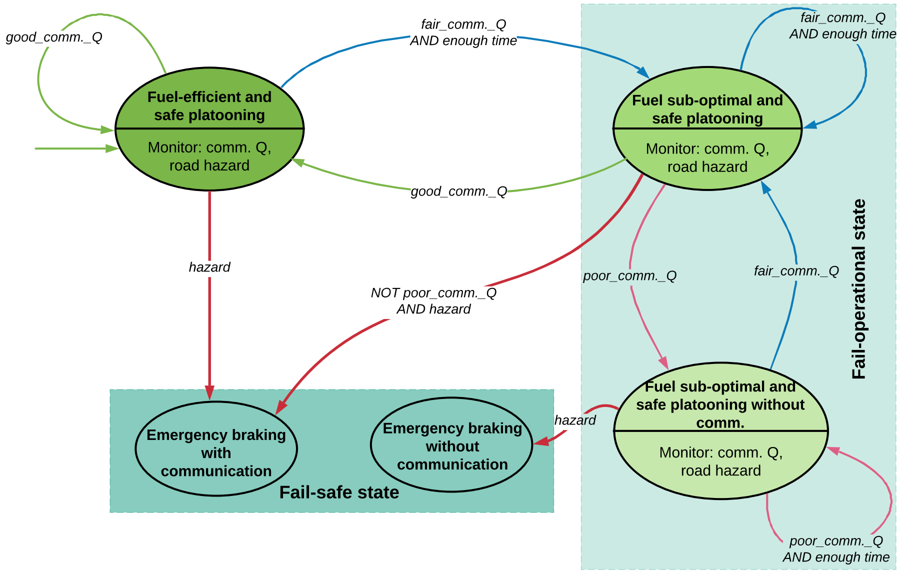


Figure 2.1: State machine representing the switching between fail-operational and fail-safe states. Update [2022-03-10]: an updated version of the state machine will be made available soon (submitted for reviewing in Feb 2022).

safety. The platoon controller monitors the communication quality and the presence of road hazards. In this state, the driver in the FVs have no task because the platoon controller administers the lateral and longitudinal control of the individual vehicles. The maintenance of this state relies on the timely delivery of the CAMs and the absence of road hazards.

- *Fuel sub-optimal and safe platooning*: In this state, the fuel efficiency degrades due to the increase of inter-vehicle distances triggered by the controller or the change of controller when detecting a communication problem. The vehicles no more act on the CDG strategy; instead, they adopt the CTG policy. The platoon transitions into this state when the communication quality deteriorates. Therefore, the communication topology and/or the inter-vehicle gaps are adjusted to keep the platoon still safe and operational.

- *Fuel sub-optimal and safe platooning without communication*: This is the state in which communication quality is poor, and it is unsafe to operate the platoon by relying on V2V communications. On-board sensors such as Lidar, Radar, and camera are used to maintain platooning but with large inter-vehicle gaps. This state uses ACC only, but still, it can be called platooning since the goals of the vehicles are the same, and they try to restore the connection with the LV.

- *Emergency braking with communication*: The platoon transitions into this state when a hazard is encountered. In this state, the platoon initiates a safe stop and ensures that no harm is done to people, environment, or equipment. The emergency braking in this state is performed in a coordinated manner with the aid of V2V communications and other on-board sensors.
- *Emergency braking without communication*: In this state also the platoon performs emergency braking to bring the vehicles into complete standstill safely. The difference is however, that V2V communication is lost in this state, and the braking is required to be performed solely with the aid of Lidar, Radar, and camera sensors.

The *fuel sub-optimal and safe platooning* state and the *fuel sub-optimal and safe platooning without communication* state constitute the *fail-operational* states in which certain platoon functionalities are kept operational but with degraded performance in terms of fuel-efficiency. The *emergency braking with communication* and *emergency braking without communication* states form the *fail-safe* state in which the platoon makes a safe stop. The goal of this study is to ensure platoon safety by defining the fail-operational and fail-safe states, and envelop the actions to be taken within the states as well as when to transition between them.

When the platoon perceives the communication quality as *good*, it performs *fuel-efficient and safe* platooning. The inter-vehicle distances are short, and a high level of vehicle autonomy is maintained in this state. If the communication quality deteriorates to *fair* and there is still enough time to switch to a degraded platoon mode such that certain platoon functionalities can be kept operational, then the platoon transitions into the *fuel sub-optimal and safe* state. From this state, the platoon can switch back to the *fuel-efficient* state once the communication quality becomes *good* again. In *fuel sub-optimal* state the PLATOON controller degrades to CACC controller that has a different communication topology and relies on the CTG strategy. If the communication quality improves, the vehicles switch back to the PLATOON controller that uses CDG strategy, and again adopts the *fuel-efficient* state. However, if the communication quality further worsens to the *poor*, then the platoon adopts the ACC controller to maintain minimal platooning functionalities. This is the *fuel sub-optimal and safe platooning without communication* state, and in this state, the inter-vehicle gaps are further increased according to the CTG approach to ensure safety. Fuel-efficiency is not the primary goal at this stage; instead, the platoon monitors the V2V communication link to see if a *fair* communication can be reestablished. In that case, the platoon again switches back to the *fuel sub-optimal* state. From the *fuel-efficient* state, a platoon cannot directly transition into the *fuel sub-optimal and safe platooning without communication* state, or vice versa. This is because, in wireless communication, the number of packet losses increase or decrease sequentially. When the packet loss threshold for transitioning into the *fuel sub-optimal and safe platooning* state is reached, the platoon makes the switching accordingly. The packet loss threshold for transitioning to the *fuel sub-optimal state without communication* appears only afterwards when the communication is declared to be lost entirely.

A hazard might be encountered in any of the *fuel-efficient*, *fuel sub-optimal*, or *fuel sub-optimal without communication* state. If the platoon is in the *fuel sub-optimal and safe platooning without communication* state in which the communication quality is *poor* when it encounters a hazard, it will transition into the *emergency braking without communication* state. In this state, the platoon needs to do the braking manoeuvre without any coordination. Nevertheless, this can still be considered sufficiently safe because the platooning vehicles should already have increased the gap due to *poor* communication quality in the fuel sub-optimal state. If a hazard is detected in the *fuel-efficient and safe platooning* state, then the platoon performs emergency braking by transitioning into the *emergency braking with communication* state. Emergency braking in this state is less challenging as the communication link is *good*. Finally, if the platoon is in the *fuel sub-optimal and safe platooning* state in which the communication quality is *fair*, when a hazard is detected, the platoon initiate coordinated braking by transitioning into the *emergency braking with communication* state. In this particular case, what kind of coordinated braking will be performed depends on the quality of the connection (number of packet losses), and emergency braking from this state is more challenging but still acceptable as the gap has been increased compared to the *fuel-efficient* state. To this end, two emergency braking strategies are proposed in this report that can ensure fail-safe braking despite having *fair* or *poor* communication quality. Another challenge is to determine the thresholds of *good*, *fair*, and *poor* communication quality based on which the platoon would perform the state transitions. This problem is also investigated in this study. Finally, the performance of different braking strategies as well as the different communication strategies are evaluated given certain channel load profiles.

2.2 Research Questions

Different platooning vehicles experience varying levels of communication latency depending on the distance between the sender and the receiver, channel congestion level, communication strategy, etc. In this study, it is intended to understand what kind of fail-operational and fail-safe means are to be applied for what extent of latency. To this end, the following research questions are addressed:

- RQ1: How to enable and evaluate fail-operational and fail-safe mechanisms in platooning under realistic wireless communication and vehicle dynamics scenarios?

- RQ2: How can fail-operational platooning be maintained in the presence of transient communication errors and with the constraints of using short inter-vehicle distances, high vehicle speed, and maintained string stability?

- RQ3: In case of emergencies caused by permanent hardware/software failures, and/or road hazards, how should the platoon coordinate to perform its emergency braking manoeuvre to transition into a fail-safe state?

- RQ4: What is the number of packet losses that determine *good*, *fair*, and *poor* communication quality such that a platoon can use it to change state and take proactive safety measures?

- RQ5: What is the communication delay incurred by the platooning vehicles under various network loads, and how does it affect platoon safety?

2.3 Contributions

In this section, an overview of the contributions of this research work is given. The contributions are mapped to the research questions. Also, the relevant sections or chapters that contain the answers to the research questions are illustrated in Table 2.1.

- *Research Contribution 1 (RC1) answers RQ1*: A runtime manager responsible for controlling the platoon in the fail-operational states has been proposed and evaluated. In addition, two emergency braking techniques have been proposed for use in the fail-safe states. The details of the state machine, the runtime manager and the emergency braking strategies are presented in *Chapter 5*. In order to evaluate the performance of the runtime manager and the braking strategies, several platoon specific performance metrics are defined. Further, a simulation tool named PlatoonSAFE has been developed to facilitate the simulation and evaluation of fail-operational and fail-safe mechanisms in platooning. The tool is designed as an extension of the Plexe simulation framework. In the PlatoonSAFE tool, four emergency braking techniques have been implemented. In addition, the runtime manager is implemented that keeps the platoon fail-operational by temporarily degrading the performance based on the proposed safety contracts. The implementation details of the RM and CEB modules in the PlatoonSAFE tool, how to use the simulator, and some use cases are presented in *Chapter 6*.

- *Research Contribution 2 (RC2) answers RQ2*: In order to answer the RQ2, the runtime manager concept is introduced that is explained in the *Section 5.1* of *Chapter 5*. Extensive simulations have been performed using the RM module of PlatoonSAFE tool under a simulation scenario in which the lead vehicle in the platoon changes its speed with an amplitude of 10 kmph and at a frequency of 0.2 Hz (sinusoidal scenario). The FVs try to track the LV in the presence of transient communication errors. The results demonstrate that the runtime manager can adjust the inter-vehicle gap and/or switch between the PLATOON, CACC, and ACC controllers to keep the platooning functionalities operational and avoid rear-end collisions. The simulation results concerning the RC2 can be found in the *Section 7.4* of *Chapter 7*.

- *Research Contribution 3 (RC3) answers RQ3*: RC3 focuses on the emergency events that can arise due to platoon related hazards and proposes solutions to circumvent the emergency. An automated emergency braking strategy entitled *Synchronized Braking* is proposed that takes into account the latency incurred in the DENM dissemination and can facilitate braking at a very high deceleration rate. With SB strategy, the platooning vehicles can avoid collisions in the events of emergency and transition the platoon into the fail-safe state fast. Moreover, another emergency braking strategy named *Adaptive Emergency Braking* is proposed. The AEB strategy leverages the communication latency incurred by the platooning vehicles in dense traffic scenarios. It performs braking such that the last vehicle brakes first and the first vehicle brakes last. The proposed braking strategy has been compared with three state-of-the-art braking strategies. The simulation outcomes demonstrate that both these braking strategies show promising results in attaining the fail-safe state, but also that they have different benefits and drawbacks depending on the network load and number of packet losses. The idea of the AEB and SB strategies are presented in the *Section 5.3* of *Chapter 5*, and the corresponding evaluation results can be found in *Chapter 8*.

- *Research Contribution 4 (RC4) answers RQ4*: Rigorous simulations have been performed using the PlatoonSAFE tool to determine how packet losses affect the performance of a platoon. A braking scenario is simulated to determine the *good*, *fair*, and *poor* threshold using various combinations of packet losses experienced by the platooning vehicles. The presented results help to understand how to set the thresholds. *Section 7.3* of *Chapter 7* contains these results.

- *Research Contribution 5 (RC5) answers RQ5*: Six different network configurations are introduced to generate various levels of network loads. To this end, the number of neighbouring vehicles, vehicle density, beacon frequency, etc., are varied. A hundred simulation runs have been performed to observe which configuration gives rise to what level of channel busy ratio, DENM delay, and ACK delay. Under all six configurations, emergency braking is tested using the several different braking strategies implemented in the PlatoonSAFE tool. The results help us understand how different levels of network load affect the emergency braking in pur-

Table 2.1: Mapping of research questions, research contributions and the Chapters.

RQ1	⟶	RC1	⟶	Chapter 5 and Chapter 6
RQ2	⟶	RC2	⟶	Section 5.1 in Chapter 5 and Chapter 7
RQ3	⟶	RC3	⟶	Section 5.3 in Chapter 5 and Chapter 8
RQ4	⟶	RC4	⟶	Section 7.3 in Chapter 7
RQ5	⟶	RC5	⟶	Section 8.3.2 in Chapter 8

suance of attaining the fail-safe state. *Section 8.3.2* in *Chapter 8* presents these results.

2.4 Research Process

The research process that has been adopted in this study to hypothesize the research problems and evaluate the proposed solutions is depicted in Figure 2.2. The first step was to identify the research domain. This was followed by initial screening, feasibility study, and planning. A list of research problems was then identified in the application domain. An extensive literature review was conducted to scrutinize the existing solutions and identify the research gap. The outcome of the literature review was used to propose novel solutions to the identified problems. For experimenting with the proposed solution, simulation studies were chosen. The rationale behind selecting simulation studies is that it facilitates large scale simulation environment and field-operational tests are often too expensive. To this end, the PlatoonSAFE tool was developed on top of the Plexe simulation framework. After implementing the idea, a range of evaluation parameters were identified to analyze the proposed solution, and compare the results with the state-of-the-art approaches. Rigorous simulations have been carried out in the data collection phase to improve reliance on the results. The obtained results were then analyzed to see if they are as expected, and the results associated with the benchmark strategies can be reproduced. This helps us understand the correctness of the simulator. After comparing the results of the proposed approaches with the state-of-the-art results, the findings were able to satisfy the expectations thus far. Besides, the shortcomings of the proposed solution were identified, and the obtained results suggested some alternative solutions. If the results were not satisfactory, the results would further be analyzed to understand why and how they deviate from the expected outcomes. Then it would require to reformulate the problem and repeat the whole process all over again.

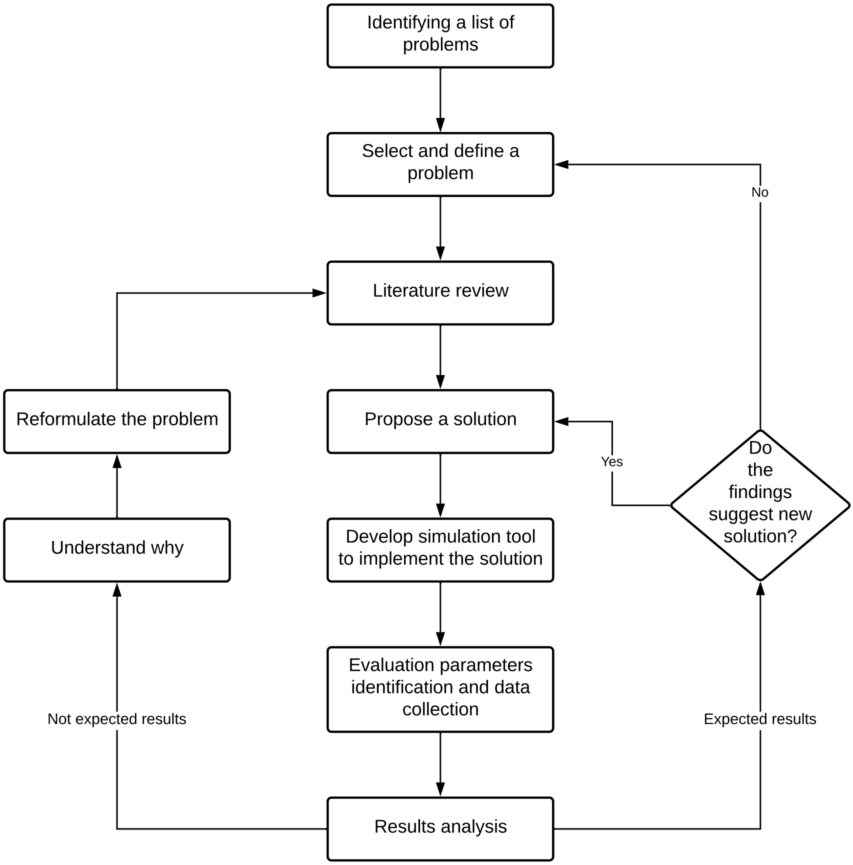


Figure 2.2: Research process.

Chapter 3

Background

In this chapter, a brief description of the levels of driving autonomy that is required to regulate a platoon is provided. This is followed by a discussion about CACC vehicle string and platooning. Then the V2X communication standards and the specifications for DENMs and CAMs are described. Finally, a background on the MAC protocols is given to unveil which protocol is more fitting for platooning applications.

3.1 SAE's Levels of Driving Automation

SAE International classifies the levels of driving automation into six categories in its J3016 standard [32]. According to this standard, Level 0 corresponds to no automation, and Level 5 corresponds to full vehicle autonomy. The main difference between Levels 0–2 and 4 in terms of driver's engagement is that, in Levels 0–2, the driver is required to be actively involved in driving by supervising the support features, e.g., lane centring, Adaptive Cruise Control, automated emergency braking etc., in order to maintain safety. On the other hand, in Level 4, the driver does not actually drive when the control features are on. However, in the case of Level 3 automation, the driver may need to take over the driving control upon request by the control features.

In terms of feature support, Level 2 differs from Level 1 in that the vehicle must have both the lane-keeping and ACC features. In contrast, a vehicle can be regarded as a Level 1 automated vehicle if any but not both of these two features are supported. Levels 3–4 vehicles can only be used in limited situations, e.g., local driverless taxi, a pre-planned hub to hub destination etc. with the aid of the support features. Level-5 automated vehicles should be able to drive in all conditions without any driver intervention whatsoever.

The coordination strategies between platooning vehicles can vary from fully decentralized to fully centralized meaning that, a group of vehicles can be said to be platooning even if there is no communication in play (decentralized) with SAE's automation level 2 [33]. This might be required in cases of communication failures where platoon safety would take precedence over attaining fuel-efficient benefits. In order to fulfil the level-3 automation requirements, the Highly Automated Vehicles (HAVs) would require some form of wireless communications to be able to drive in limited scenarios without any supervision from the driver. The communication strategy can be of different types, e.g., unidirectional, bidirectional, n-vehicle look ahead, fully centralized etc., depending on the situation arisen by a particular platooning scenario [33, 34, 35]. In this research work, it is considered that the following vehicles in the platoon operate at Level-4 automation, while the LV drives with Level-3 automation. The LV is equipped with Advanced Driver Assistance System (ADAS) that facilitates forward collision avoidance systems through automated emergency braking.

3.2 CACC and Platooning: Distinctions and relationships

The terms CACC and platooning are extensively used throughout this report. However, these two terms are often mistakenly used as synonymous even though there is an apparent distinction [23].

Both CACC and platooning fall under the broad category of automated vehicle speed control system [23]. However, platooning can be regarded as a tightly coupled system of vehicles that requires information from the LV to be reached to all the FVs. The principal distinctions and relationships can be described as follows:

- The goal of the platooning vehicles is common for a particular period, and there is a designated leader. However, the goals of the individual vehicles in a CACC vehicle string can be different; hence, the control of the CACC vehicle string is decentralized. Although the goals such as fuel efficiency improvement, close vehicle following, improvement in safety by avoiding rear-end collisions, driver offload are common for both the systems.

- CACC vehicle string offers longitudinal control only, i.e., the task of steering the wheel, lane maintenance, active monitoring of the driving environment is still up to the driver. In contrast, a platoon is required to accommodate both lateral and longitudinal control. Hence, CACC provides the SAE level-1 driving autonomy [32] only, whereas the automation of platooning starts from level-2. This is one of the significant differences between the CACC vehicle string and platooning [36].

- The second most essential distinction is that platoon operates based on the CDG strategy [17]. This means the gap between the platooning vehicles does not change with the variation of speed. On the other hand, the CACC vehicle string follows the CTG strategy in which the time gap between the vehicles change proportionally with speed. A *time gap* is the elapsed time between the ego vehicle's front bumper and preceding vehicle's rear bumper traversing a reference point on the road.

- The membership in a CACC vehicle string can be ad hoc. However, membership in a platoon is coordinated by the platoon leader.

- As the gap between the vehicles changes with the speed variation in a CACC vehicle string, it is less fuel-efficient. In platooning, the vehicles are tightly coupled; Browand *et al.* [37] reported 20% to 25% fuel saving by maintaining 3-6 meters gap between platooning vehicles tested under the California PATH project. Furthermore, another project named SARTRE [38] focused on multi-band platooning to minimize fuel consumption up to 20%, and fatalities due to road accidents up to 10%.

- Another differentiating parameter is the type of information required and the source from which they are collected. In CACC vehicle string, kinematic parameters (speed, acceleration, position) received from the preceding vehicle via V2V communication can be enough to perform longitudinal control. However, in Platooning, the FVs require lateral control information and platoon maintenance information in addition to the kinematic parameters both from the lead vehicle and/or the preceding vehicle.

The terms CACC and PLATOON will be distinctively used throughout this report. In CACC, the vehicles use the predecessor following strategy, i.e., a vehicle receives information through V2V communication from its immediate predecessor only. In addition, CACC uses the CTG approach. On the other hand, a PLATOON of vehicles uses the predecessor and leader following strategy and the CDG approach.

3.3 V2X Communication Standards

V2X communication is the key enabling technology for platooning applications that allows the platooning vehicles to communicate within themselves and other external systems such as, Road Side Units (RSUs), fog nodes, pedestrians, cyclists etc., Figure 1.1. In order to facilitate communication between the Intelligent Transport System-Stations (ITS-Ss), 75 MHz bandwidth was allocated in the 5.9 GHz band (5.850-5.925 GHz) in the US. This spectrum is subdivided into seven channels, 10 MHz each. In Europe, 30 MHz frequency band (two Service Channels (SCHs) and one Control Channel (CCH)) is allocated for safety and traffic efficiency applications [39] in the 5.9 GHz band. The CCH is intended for exchanging control messages such as periodic beacons and event-driven messages for safety applications. The SCH can accommodate specific applications, e.g., platooning. The IEEE 802.11p protocol [40] which has recently been integrated into the IEEE 802.11-2012 standard, provides the

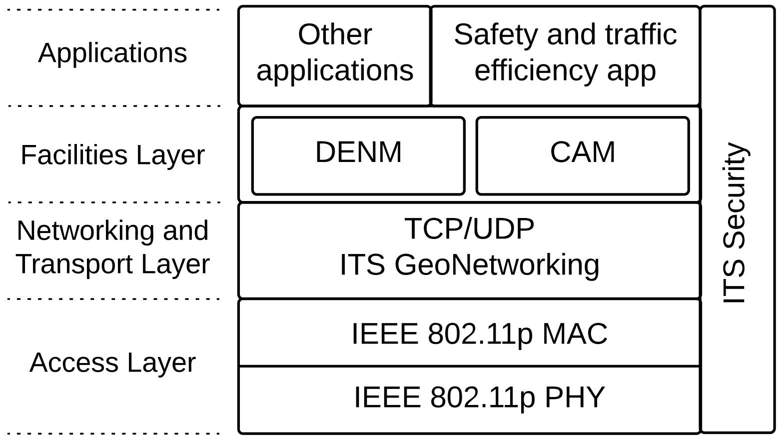


Figure 3.1: ETSI ITS-G5 Protocol Stack.

Physical (PHY) and MAC layer specifications for short-range vehicular communications. Europe and the US have separate protocol stacks built on top of the IEEE 802.11p protocol. The IEEE 1609 working group in the US defined the IEEE 1609.x protocol stack, also known as the Wireless Access in Vehicular Environment (WAVE). Its European counterpart, the ETSI specified the ITS-G5 architecture [41] for regulating operations in the 5 GHz frequency band, Figure 3.1.

In the WAVE protocol stack, the IEEE 1609.4 MAC sublayer provides specifications for multi-channel operations in the form of channel timing and switching. The IEEE 1609.3 sitting on top of the 1609.4 defines Internet Protocol version 6 (IPv6) addressing and Wave Short Message Protocol (WSMP). Application layer and Security specifications are administered by the IEEE 1609.11 and 1609.2 standards, respectively. The corresponding WSMP functionalities in the ETSI ITS-G5 stack are specified by CAM [24] and DENM [25].

According to the ETSI EN 302 637-2, CAMs are exchanged between the ITS-Ss in the ITS network intending to provide situational awareness and support cooperative performance. CAMs contain *status* and *attribute* information of the originating ITS-S. In case of platooning vehicles, the *status* information can include speed, position, acceleration, steering angle, etc. The *attribute* information is the physical properties of the vehicles, e.g., vehicle type, dimension, etc. CAM acts like an adjustable container that can be tuned according to the application specifications. Hence, the information contained in CAM is application dependent. Besides, the number of CAMs broadcasted per second is also flexible. According to the ETSI EN 302 637-2 specifications, CAMs are generated at a frequency between 1 to 10 Hz based on the change of vehicle kinematic parameters in order to keep the channel congestion under control.

The size of the CAMs can also be in the range of 200B to 800B. Examples of status and attribute data contained in CAM are as follows:

- A heading that univocally identifies the stationID of the sending ITS-station
- Timestamp containing the CAM generation time
- Vehicle kinematic parameters such as speed, position, acceleration, heading, curvature, etc.
- Vehicle category, vehicle roles, etc.
- Signature, certificate, etc.

CAMs can also be generated based on the change in position, speed, and direction of the ITS-station. The EN ETSI 302 672-2 standard has also defined the threshold for CAM generation. They are as follows:

- *Speed*: Change in position by more than 4 meters
- *Direction*: Change of direction greater than or equal to $+/-4\,\text{deg}$
- *Speed change*: More than or equal to 0.5 ms^{-1}

The ETSI TS 102 637-2 standard defines DENM specifications intending to support a notification service about road status. Although the specification is intended for road safety applications, it can be utilized by the ITS-Ss to receive information about the road traffic conditions. DENMs are event-driven messages which are triggered when an event of common interest occurs and are spread within an area of interest for the duration of the event. DENMs have soft deadlines but require high reliability, i.e., the messages can have a certain level of usefulness even after the deadline, but they must be delivered in order to react to the hazard that triggers the DENM. According to the standard specification, a DENM should have the following attributes:

- *Event type*: The type of the event that triggers DENM dissemination, e.g., traffic jam, the sudden appearance of road hazard, a broken car on the highway, road accident etc.
- *Event position*: Specific position of the event or the geographical area
- *Event detection time*: The expected time by which the event is anticipated to be terminated
- *Destination area*: The area over which DENMs are required to be disseminated
- *Transmission frequency*: The frequency at which the sending ITS-station transmits DENMs

Table 3.1: Parameters for EDCA Access Categories in IEEE 802.11p [42].

Traffic type	AC	CW_{min}	CW_{max}	AIFSN
Background	0	15	1023	9
Best effort	1	15	1023	6
Video	2	7	15	3
Voice	3	3	7	2

DENM is transmitted over a period according to the specified frequency. Once the status of the event changes, the ITS-station updates the DENM contents. Upon reception of the DENMs, the receiving ITS-stations take necessary actions, e.g., emergency brake. Besides, the stations can also relay the message to inform other neighbouring stations if the event is of common interest.

3.4 Medium Access Control Protocols

The ETSI ITS-G5 stack adopts the MAC protocol from the IEEE 802.11 standard for Wi-Fi. This protocol is non-deterministic and performs poorly in dense traffic scenarios. If a platooning scenario is considered in which the non-platooning vehicles also transmit CAMs periodically, then this can lead to channel congestion. To this end, ETSI published TS 102 687 that is a specification for Decentralized Congestion Control (DCC) mechanism in the access layer of ITS-G5 protocol stack. Each vehicle in the ITS network applies DCC in the access layer independently from the other ITS-Ss. This makes the DCC mechanism a fully distributed system. In order to ensure fairness and mitigate channel congestion, access to the channel is performed based on several transmission parameters, e.g., transmit power, radio sensitivity, data rate etc., according to the measured channel load.

The MAC protocols are intended for providing channel access, but also reliability and real-time requirements to the contending nodes by sharing the channel fairly. In vehicular networks, the MAC protocols can be divided into three broad categories: Contention-based protocols, contention-free protocols, and hybrid protocols.

The IEEE 802.11p protocol employs contention-based channel access protocol as MAC method; it is inherited from the Enhanced Distributed Channel Access (EDCA) mechanism of IEEE 802.11 protocol. According to EDCA, a wireless node has four different Access Categories (ACs) for four different traffic types such as voice, video, best effort and background traffic. Each AC has different values for Contention Window (CW) and Inter-Frame Space Number (AIFSN[AC]) that determines the backoff time. The default EDCA parameters for different ACs are depicted in Table 3.1.

The EDCA mechanism employs the Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) protocol to administer how the contending nodes access the shared medium. In CSMA/CA, a wireless node senses the medium first and checks if any other node is transmitting. If the medium is free for AIFS period, the node still does not transmit; rather it defers the transmission adopting a random backoff time. The backoff procedure in 802.11 protocol is performed in the following steps:

1. A backoff time is selected uniformly between [0, CW+1] where the initial value of the CW is CW_{min} of the corresponding AC.

2. If the subsequent transmission fails, the backoff interval is doubled until the CW_{max} value is reached.

3. The backoff value is reduced only when the channel is sensed free

4. Once the backoff value reaches zero, the station can transmit immediately

In the CSMA/CA protocol, there is no guarantee for timely channel access, and the mobile nodes can experience unbounded delay and packet losses in dense traffic scenarios. The authors in [43, 44] demonstrated through extensive simulations that CSMA/CA protocol tends to behave like ALOHA protocol because the benefits of the *listen before talk* mechanism diminish in high-density traffic scenarios. Consequently, the CSMA/CA protocol is unsuitable for safety-critical applications that have stringent performance requirements.

Due to this, contention-free MAC protocols were proposed in which the medium is allocated based on a predefined schedule to overcome unbounded delay and fairness issue of CSMA/CA protocol. Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA), Frequency Division Multiple Access (FDMA) are the examples of contention-free MAC protocols used for inter-vehicle communications [45]. These protocols resolve the collision problem of IEEE 802.11p MAC by assigning unique time slot, code sequence, and frequency band, respectively for the individual nodes. TDMA-based MAC protocols have emerged as a viable solution to ensure fairness and provide a bounded delay in the vehicular networks. In TDMA, time is partitioned into frames, and then frames are further subdivided into timeslots. Each vehicle can transmit in one or more dedicated timeslots in a frame. In TDMA protocol, the nodes use the same frequency but at different times; therefore, the nodes are required to be frequency synchronized. As the nodes transmit at different time slots, TDMA solves the issue of interference due to concurrent transmissions. However, the efficiency of TDMA-based MAC protocols is limited by the requirement of high-level of coordination and extra overhead in the events of retransmissions [46].

The hybrid MAC protocols leverage the benefits of contention-based and contention-free MAC protocols. The Self-organizing Time Division Multiple Access (STDMA) [47] is an example of such a hybrid protocol that was developed for real-time communications. STDMA is a decentralized and predictable scheme in which synchronization is performed with the aid of Global Positioning System (GPA). STDMA ensures that all the nodes can gain access to the shared medium. Each node selects a TDMA time slot after the four steps: *initialization*, *network entry*, *first frame*, and *continuous operation*. The authors compared the channel access delay of the CSMA/CA protocol and STDMA. Simulation results demonstrate that the IEEE 802.11p MAC protocol is not suitable for real-time communications because of the unacceptable channel access delay. STDMA, on the other hand, exhibits bounded and relatively small channel access delay. Furthermore, Bilstrup *et al.* showed that when concurrent transmission takes place in space, access to the channel is randomly distributed in case of CSMA [48]. However, in STDMA, side information from the position messages is used to schedule the channel access. Hence, STDMA performs exceptionally well in dense traffic scenarios. STDMA protocol ensures fairness, predictable channel access delay and scalability, which is why it is more suitable for real-time applications such as platooning. Another type of hybrid protocol that has received significant research attention is called overlay TDMA protocol. In overlay TDMA, a TDMA layer is placed on top of the IEEE 802.11p native CSMA/CA protocol. In platooning, the vehicles transmit packets periodically in a cyclic manner following the overlay TDMA scheme. As the CSMA protocol is still in place, it helps to tolerate the external traffic generated by the non-platooning vehicles. Thus, the overlay TDMA protocol virtually removes collision within the platoon by placing a TDMA layer on top of CSMA, and inherits the scalability feature of CSMA protocol, and fairness feature of TDMA protocol. The slotted-TDMA protocol presented in [49] is an example of an overlay TDMA strategy. There are several other hybrid MAC protocols available in the literature, e.g., CS-TDMA [50], SOFT-MAC [51], HER-MAC [52] etc.

In all the works presented in this report, the standard IEEE 802.11p MAC protocol is used despite its drawbacks. The intention was to examine how CSMA/CA protocol performs in dense traffic scenarios, and whether it is suitable for safety-critical application such as platooning. Some scenarios are introduced in which the vehicles experience high communication latency due to packet losses and high channel access delay. In this report, fail-operational and fail-safe approaches are proposed taking into consideration the high latency that can be incurred by the platooning vehicles. In future, the efficacy of the hybrid MAC protocols to attain fail-operational and fail-safe state will be studied.

Chapter 4

Related Works

When a group of connected and automated vehicles form a platoon, it requires a control algorithm that dictates the relative distance to be maintained with the preceding vehicle while moving at a consensual speed. The efficacy of a control algorithm is evaluated by its ability to sustain *string stability* and enable fuel-efficiency by maintaining short inter-vehicle distances. A string of vehicles is said to be string stable when the control algorithm can attenuate the motion and space disturbances at the head of the platoon towards the tail of the platoon [30]. Different control algorithms proposed in the literature use different control topologies to achieve string stability, e.g., predecessor following topology, leader-predecessor following topology, bidirectional topology etc. In this chapter, some state-of-the-art controllers that are used in the works presented in this report are analyzed. In addition, the pros and cons of some collision avoidance systems found in the literature are discussed to help to compare with the emergency braking strategies presented in this report. Moreover, a brief literature review on the safety analysis methods is conducted. This is related to keeping a platoon fail-operational in the events of failures due to transient communication errors.

4.1 Control Algorithms

4.1.1 Adaptive Cruise Control

ACC is the speed control system that receives the information required for following a preceding vehicle from radar, lidar, and/or camera sensors. An ACC enabled vehicle can adjust its speed to maintain a safe gap with the front vehicle based on the data provided by the radar sensor. When a fleet of vehicles uses ACC only, they result in string instability due to sudden change in speed. This is because, the second vehicle must sense, process, incorporate in control, and then respond to the speed change of the lead vehicle. Similarly, the other following vehicles must go through the same steps, and it causes detection and actuation lag. Shladover *et al.* reported that if a high-performance vehicle requires 1.5 s as the detection, processing, and actuation delay, then the fourth vehicle in a fleet would need 4.5 s to respond to the speed change of the lead vehicle while using ACC only [36].

In this report, the ACC algorithm presented by Rajamani in [19] is used, and the control law of i^{th} vehicle is given by

$$\ddot{x}_{i\_des} = -\frac{1}{T}(\dot{\varepsilon}_i + \lambda\delta_i), \tag{4.1}$$

$$\delta_i = x_i - x_{i-1} + l_{i-1} + T\dot{x}_i, \tag{4.2}$$

$$\dot{\varepsilon}_i = \dot{x}_i - \dot{x}_{i-1}, \tag{4.3}$$

where i is the ego vehicle, and $i-1$ is the front vehicle. T is the time gap, x_i is the position of the i_{th} vehicle, $\dot{x}_i$ is the speed of the i_{th} vehicle, and l_{i-1} is the length of the front vehicle. Therefore, δ_i is the distance error; the difference between the desired distance $T\dot{x}_i$ and the actual distance $x_i - x_{i-1} + l_{i-1}$ to the front vehicle. $\dot{\varepsilon}_i$ is the relative speed between the ego and the front vehicle.

A vehicle using this ACC control law follows the CTG policy. It maintains a T seconds time gap with respect to the front vehicle to preserve string stability. Recall that in CTG policy, the inter-vehicle gap grows proportionally with the speed of the front and ego vehicles.

4.1.2 Cooperative Adaptive Cruise Control

In CACC, the ego vehicle receives the intention of the preceding vehicle through V2X communication in addition to the sensor information. Thus, the ego vehicle can know, for example, the braking intention of the preceding vehicle even before it has started decelerating. CACC eradicates the actuation lag of the preceding vehicle and sensor detection lag of the ego vehicle. The only delay in dealing here is the communication delay. The most basic form of CACC is the predecessor-following approach in which the ego vehicle receives information from the preceding vehicle only through V2X communication in addition to the radar sensor [53]. In a four-vehicle fleet, the intention of the lead vehicle is propagated to the fourth vehicle within 400-800 ms, considering small processing delay and 100 ms of CAM interval [36]. This is much better than the 4.5 s required in the ACC controller.

Segata [54] defines the CACC control law as

$$\dot{u}_i = \frac{1}{T}(-u_i + k_p(x_{i-1} - x_i - l_{i-1} - T\dot{x}_i) + k_d(\dot{x}_{i-1} - \dot{x}_i - T\ddot{x}_i) + u_{i-1}), \tag{4.4}$$

where T is the time gap and u_{i-1} is the intended acceleration of the preceding vehicle that is communicated to the ego vehicle through V2V communication. k_p and k_d are the controller gains used to tune the controller behaviour. CACC

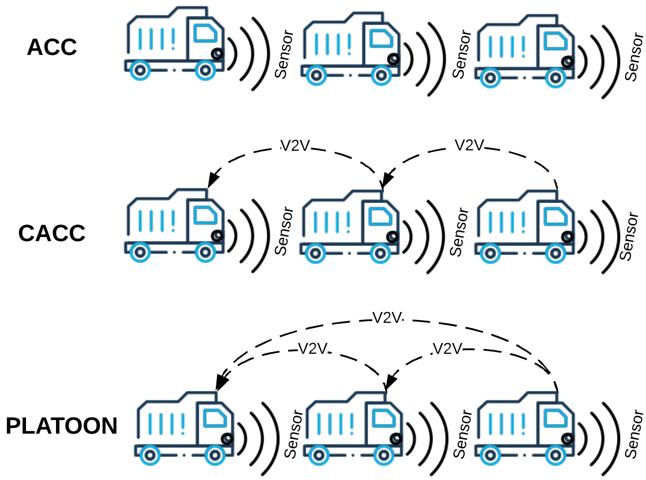


Figure 4.1: ACC, CACC, and PLATOON driving.

controller also uses the CTG policy, and Ploeg *et al.* demonstrated through practical experimentation that a fleet of six vehicles could exhibit string stable behaviour with 0.7 s time gap.

4.1.3 PLATOON

In the California PATH project [55], the authors proposed a controller based on the leader-predecessor communication strategy and the CDG policy. In this controller, an ego vehicle can receive information from the lead vehicle that may not be in the line of sight, in addition to the front vehicle. In the four vehicle fleet example, the fourth vehicle would require only 100 ms to learn the intention of the lead vehicle (in an ideal scenario) [36]. Because of the constant distance gap policy and the additional phase lead due to the information from the lead vehicle, the vehicles can maintain inter-vehicle gaps as short as 5 m [54]. The short inter-vehicle distance enables fuel-efficiency and enhances road throughput. To this end, the PATH CACC controller is regarded as the PLATOON controller throughout this report. Figure 4.1 graphically shows the difference between the ACC, CACC, and PLATOON controllers in terms of the source from which the ego vehicle collects information.

According to [19], the control law of the PLATOON controller can be given by

$$\ddot{x}_{i\_des} = \alpha_1 \ddot{x}_{i-1} + \alpha_2 \ddot{x}_0 + \alpha_3 \dot{\varepsilon}_i + \alpha_4 (\dot{x}_i - \dot{x}_0) + \alpha_5 \varepsilon_i, \tag{4.5}$$

$$\varepsilon_i = x_i - x_{i-1} + l_{i-1} + gap_{des}, \tag{4.6}$$

$$\dot{\varepsilon}_i = \dot{x}_i - \dot{x}_{i-1}. \tag{4.7}$$

$\ddot{x}_0$ and $\dot{x}_0$ are the acceleration and speed of the leader, respectively. gap_{des} is the desired gap between the two vehicles. ε_i and $\dot{\varepsilon}_i$ are the distance error and relative speed between two vehicles, respectively. The α_i parameters are given by

$$\alpha_1 = 1 - C_1; \alpha_2 = C_1; \alpha_3 = -\Big(2\xi - C_1\big(\xi + \sqrt{\xi^2 - 1}\big)\Big)\omega_n, \tag{4.8}$$

$$\alpha_4 = -C_1\big(\xi + \sqrt{\xi^2 - 1}\big)\omega_n; \alpha_5 = -\omega_n^2. \tag{4.9}$$

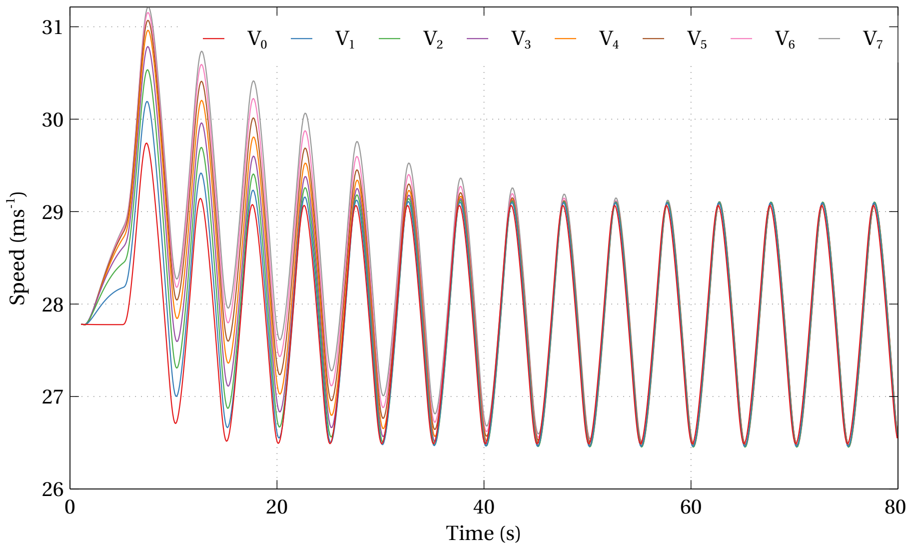
C_1 is the weighting factor, ξ is the damping ratio, and ω_n is the controller bandwidth.

Santini *et al.* in [56], proposed a controller model in which the platoon acts as network topology, with vehicles being the nodes, and edges being communication links between the nodes. Such a model performs well in the cases where the platoon structure changes frequently, and there is a varying communication latency between the leader and the following vehicles. In addition to the above, there are a considerable number of works that studied the effects of communication latency on string stability [57, 58, 59], and proposed CACC controllers to mitigate them. These works mostly provide analytical models for determining the communication latency. However, the authors in [60, 61, 62] used simulations to assess the communication latency in the presence of packet losses, network failures, and different beaconing frequencies. The performance and robustness of the proposed controllers were then studied in the presence of such disturbances.

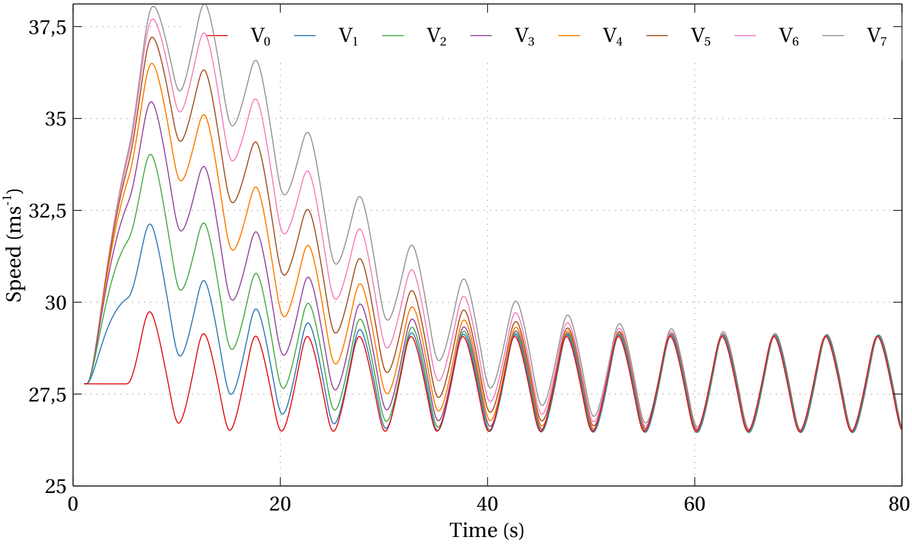
4.1.4 Controller Analysis

In this part, the ACC, CACC, and PLATOON controllers are analyzed by presenting some simulation results. The simulations are performed in the Plexe simulator [30], and the default simulation parameters are kept unchanged. A platoon of eight vehicles is considered. The leader drives at a speed of 100 kmph; the rate is varied in a sinusoidal fashion with an oscillation frequency of 0.2 Hz and an oscillation amplitude of 10 kmph. The ACC controller is tested with a time gap 0.3 s and 1.2 s. The CACC and PLATOON controllers are tested with 0.5 s time gap and 5 m constant distance gap, respectively. The beacon interval is 100 ms. Please refer to the Plexe website [63] for the details of the communication and controller parameters.

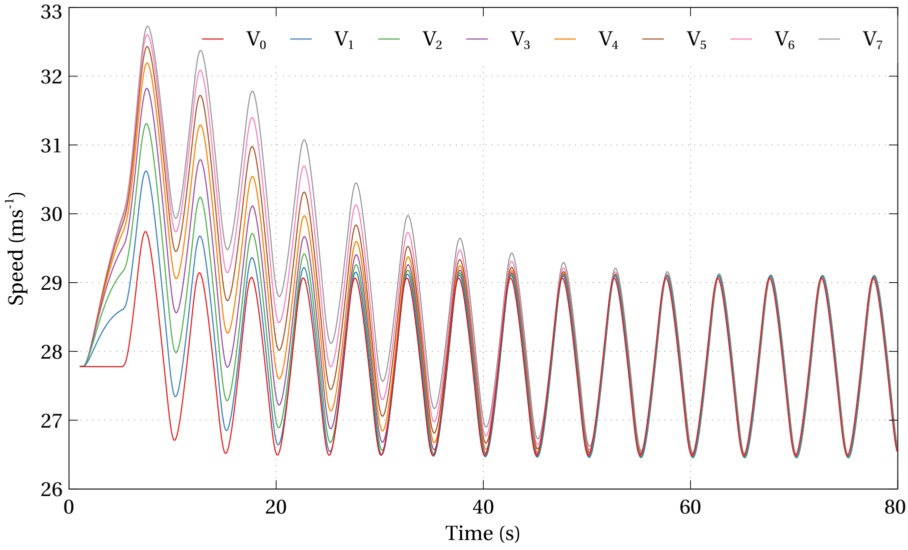
Figure 4.2 presents the speed profiles of the platooning vehicles with ACC, CACC, and PLATOON controllers. The aim is to analyze the string stability property. The following vehicles start with a time gap that is not the desired



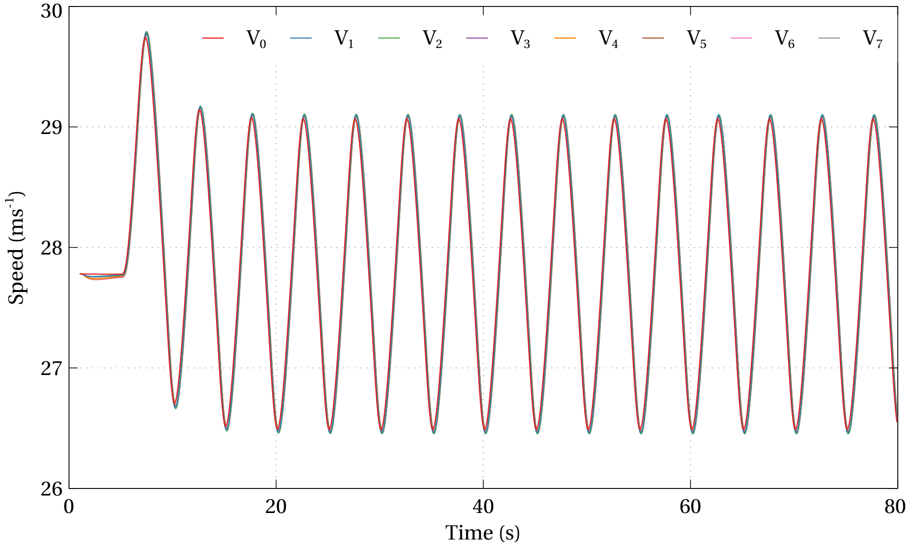
(a) ACC, time gap = 0.3 s



(b) ACC, time gap = 1.2 s



(c) CACC, time gap = 0.5 s



(d) PLATOON, constant distance gap = 5 m

Figure 4.2: Speed profiles of the platooning vehicles with ACC, CACC, and PLATOON controllers (sinusoidal scenario).

time gap. As a result, the following vehicles increase the speed to match with the front vehicle at the beginning of the simulation. With the CTG policy, the motion disturbance of the leader is amplified in the downstream direction due to the tracking lag, Figures 4.2a, 4.2b, 4.2c. When the time gap is as short as 0.3 s, it can potentially be dangerous. With PLATOON controller, the following vehicles can perfectly track the leader with no speed difference at all, Figure 4.2d. This is due to the leader-predecessor following strategy. However, this simulation scenario is ideal, i.e., no neighbouring vehicles are contending for channel access.

4.2 Collision Avoidance Systems

CASs for platooning applications have received significant research interest in the recent past. The most popular CAS aiming to avoid rear-end collisions is to use varying deceleration rates for different vehicles in the platoon. The authors in [11] proposed that the truck with higher brake torque becomes the FV in a two-vehicle platoon. They also tested an *override braking* scheme in a truck driving simulator in which the automatic braking system first brakes and then the backup driver makes full braking. This strategy was proven to perform better than manual braking. Zheng *et al.* in [9, 10] also proposed that the last vehicle in the platoon should brake at the highest deceleration rate, and the rate should gradually decrease in the upstream direction. In this report, this approach is regarded as the *Gradual Deceleration* strategy. The authors also investigated the possibility of braking and steering at the same time in the events of emergency. Simulations were performed in a truck driving simulator with twelve professional drivers, and the results demonstrate that the only braking action is a better option than steering and braking to ensure platoon safety. The authors in [64] proposed a braking strategy, namely *the law of the weakest*. The idea is that when a vehicle joins the platoon, if its deceleration capability is lower than that of the other platooning vehicles, then the whole platoon should adjust its maximum deceleration rate during the braking manoeuvre.

Some studies have considered the communication latency incurred by the platooning vehicles while designing CASs. For instance, the authors in [12] pointed out two problems that can occur while performing emergency braking: inter-vehicle communication failure and failure of the primary brake system. The proposed solutions are, adjusting braking force (last vehicle with highest braking force), and a redundant brake system. Taleb *et al.* [65] addresses the long message delivery latency issue from the time of detecting the emergency until when the platooning vehicles are notified about the event. The authors propose to adjust the contention window period of the CSMA/CA protocol based on the emergency level assigned to the platooning vehicles. The authors in [66] studied the effects of traffic densities on the communication channel while performing emergency braking for both LTE and IEEE 802.11p networks. The Key Performance Indicators (KPIs) used for performance evaluation are

Packet Error Rate (PER), communication delay, and inter-vehicle distance after emergency braking. However, no specialized braking strategy is used in the simulation of this work. Flores *et al.* [67] proposed a complete platooning model that uses a fusion of sensors, V2V communication, and Pedestrian-to-Vehicle (P2V) communication to predict the movements of pedestrians and perform emergency braking if necessary. The authors presented a car-following model, and a gap closing algorithm in case braking is performed by a vehicle in the middle of the platoon. Thunberg *et al.* proposed an analytical model that determines a feasible region of communication latency within which the platooning vehicles are guaranteed to perform safe braking [68]. Numerical results demonstrate that a safe braking probability of 1 can be reached in a 5-vehicle platoon when the vehicle with the best braking capacity is made the last vehicle in the platoon, and the vehicle with the worst braking capacity is made the lead vehicle. However, this approach would cause the lead vehicle to traverse longer distance.

In [69], the authors proposed a *Coordinated Emergency Brake Protocol (CEBP)* in which the lead vehicle broadcasts an emergency message to the FVs. Upon reception of this message, the FVs remain prepared for braking. Once the last vehicle receives the message, it passes on an ACK message in the upstream direction. The platooning vehicles perform their braking manoeuvre upon reception of the ACK message from the immediately succeeding vehicle. This way, the last vehicle brakes first, and the LV brakes last. One major drawback of this strategy is that the start of braking manoeuvre by a vehicle is conditioned by the successful reception of the ACK message. Moreover, that also has to be received from the succeeding vehicle only. This will incur additional delay for the ACK message to be propagated from the last vehicle to the first. In dense traffic scenarios, DENM delay, and ACK delay can be relatively high that can cause the lead vehicle to traverse long distance before it can brake. Another interesting study performed by the authors in [70] utilizes the Unscented Kalman Filter intending to predict the intents of the preceding vehicles 1.5 seconds earlier based on the vehicle kinematic parameters communicated through V2V communications and then informing the driver of the ego vehicle in the windshield display.

The emergency braking strategies proposed in this report emphasize on the fact that communication latency can be relatively high in dense traffic scenarios, and the latency significantly affects the platoon safety in the events of emergency. Thus, the proposed braking strategies devise solutions towards fail-safe emergency braking, keeping the latency constraint under consideration. The proposed AEB strategy improves the CEBP strategy to satisfy the requirements of the fail-safe state.

4.3 Safety Analysis Methods

In order to analyze the functional and operational safety of the platoon, it is essential to understand the safety methods adopted. There are two types of safety analysis methods available in the literature. The first kind considers the safety of the whole system by identifying the possible failures that can lead to a hazard. The second kind deals with safety on the component level.

The safety analysis and risk assessment methods such as Hazard and Operability Analysis (HAZOP), Fault Tree Analysis (FTA) consider safety for the whole system. These methods are used to identify potential hazards and failure modes. Based on the identified failure modes, appropriate system responses are planned to keep the system acceptably safe. An example of the system response to failures can be graceful degradation of system performance [71]. This is particularly relevant in the context of transient wireless communication errors that may or may not lead to failure. As a response to transient error, a system can gracefully degrade its performance so that the performance can be upgraded again when the error is resolved. In the context of platooning, the communication quality can be monitored during the runtime as it cannot be known during the design time, and a certain level of system safety can be assured by degrading the platoon performance gracefully. The SafeCOP runtime monitoring architecture [29] provides contract-based safety assurance for the cooperative systems. A contract $C =< A, G >$ of a system component can be defined as a pair of assertions in which the component behaviour is guaranteed according to the Guarantee G, given that the Assumptions A are fulfilled [72]. Sljivo *et al.* differentiate between strong and weak contracts [73]. The component behaviour guaranteed by the strong contract must always hold in every environment, whereas a weak contract is more flexible. A weak contract $C =< B, H >$ should only hold when the weak assumptions are fulfilled in addition to the strong assumptions.

Girs *et al.* [74] applied FTA to determine what might go unsafe in the communication between Cooperative-Cyber Physical Systems (CO-CPS) and proposed a contract-based safety assurance method based on that. The authors first perform a safety analysis by describing the reasons for communication failure and identifying two parameters (Packet Delivery Rate (PDR) and the number of consecutive failures) to detect it. Based on the identified parameters, the system switches its operation mode (normal, degraded, and full-stop) following some predefined contracts. In [75], the authors proposed contract-based design of degradation cascade as a way of keeping a platoon acceptably safe by switching to a lower platooning mode, e.g., from PLATOON mode to ACC mode. A set of strong contracts are proposed that represents the overall safety goals in a platoon. In addition, the weak contracts are proposed to settle when a platooning vehicle should switch between the controllers such as PLATOON, CACC, and ACC. The evaluation of the safety contracts in these works is left as future work.

The authors in [76], proposed a Safety Checker algorithm that calculates a safe headway distance and safe standstill distance for a CACC system in real-time. The design of this algorithm considers the transient errors that can occur due to temporary failure of radar and V2V communications and aim at continuing the platooning operation despite that. This graceful degradation algorithm takes inaccuracies in acceleration, velocity, and inter-vehicle gap as input, and dynamically adjusts the inter-vehicle distances in order to provide a fault-tolerant CACC system. The authors also simulated some critical CACC situations to analyze the effects of radar and communication failure in [77]. They showed that a dynamic spacing policy could significantly reduce the probability of rear-end collisions in a platoon. A significant finding of this work is that V2V failure can lead to a much more hazardous situation than radar failure while the LV in a platoon performs emergency braking. The works presented in [76, 77] are focused on the functional and operational safety of a platoon in the component level.

In this research work, the runtime manager proposed in [75] is implemented that gracefully degrades platoon performance based on certain predefined safety contracts in the events of transient communication errors. In our implementation, more weak contracts are implemented in addition to the ones specified in [75].

4.4 Existing Tools for VANET Simulation

Simulation is a widely used research method for performance evaluation of VANET applications [78]. This section aims at providing a basic familiarity with the available verification and validation tools for VANET simulation. The VANET simulators can be divided into two categories: Traffic mobility simulators and Network simulators. The traffic mobility simulators provide realistic mobility models of the vehicles, whereas, the network simulators are used for implementing and evaluating the performance of network protocols. From the VCPS point of view, traffic simulators simulate the physical plane, and the network simulators simulate the cyber plane.

4.4.1 Traffic Mobility Simulators

Simulation of Urban Mobility (SUMO) [79] is an open-source, microscopic traffic mobility simulator suitable for large scale vehicular mobility simulation. In SUMO, the physical properties of vehicles, realistic routes, obstacles, etc. can be specified. It also supports different car-following models, e.g., Intelligent Driver Model (IDM), Krauss model [80]. Moreover, SUMO allows bi-directional coupling with network simulators through an external application. The name of some other popular traffic simulators are VISSIM [81], VanetMobiSim [82] etc.

4.4.2 Network Simulators

NS-3: NS-3 [83] is a discrete event network simulator written in C++ that overcomes some of the complexities of its predecessor NS-2, e.g., removal of the C and TCL programming language interactions. NS-3 provides an implementation of IEEE 802.11p and IEEE 1609.x standards. Besides, NS-3 also

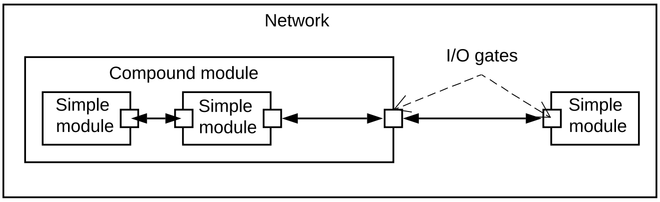


Figure 4.3: OMNeT++ simple and compound modules.

simulates some vehicular mobility models. This network simulator can be used in conjunction with traffic simulators such as SUMO, VISSIM etc.

OMNeT++: OMNeT++ [84] is an open-source, discrete event simulator with strong Graphical User Interface (GUI) that can be used to simulate any system that can be described as a queuing system [85]. Everything in OMNeT++ starts with a node which is a compound module consisting of some simple modules. The functionalities of a simple module are defined by C++ code. The simple modules can indefinitely be combined like Lego parts with I/O gates to form a compound module, Figure 4.3. This is why OMNeT++ is highly modular, and it is easy to modify the codes of OMNeT++-based simulators. There are several OMNeT++-based simulators developed for simulating different types of networks, e.g., SimuLTE [86] for LTE-user plane simulation, INET [87] for MANET simulation, Veins [61] for VANET simulation etc. A brief description of some essential built-in OMNeT++ functions and the file system is given below:

- *File System:* Let us consider a communication network that consists of some nodes, and a wireless network connects the nodes. OMNeT++ introduces NED language to describe a Network, and the file containing the network has the `.ned` extension. The corresponding network parameters are specified in an initialization file that has `.ini` extension. For instance, the data rate between two nodes, transmission power, channel model to be used, mobility parameters, etc. can be specified in the `.ini` file. The back-end programming language is C++. The variables declared in the `.ned` file are read in the C++ files, and the standard C++ library functions can be used to program the behaviour of the simulator. In addition, there are built-in OMNeT++ functions. Please refer to the OMNeT++ API reference for a detailed description [88].

- *scheduleAt():* This function is used to schedule a simulation event at a particular simulation time.

- *handleSelfMessage():* When an event is scheduled using the `scheduleAt`

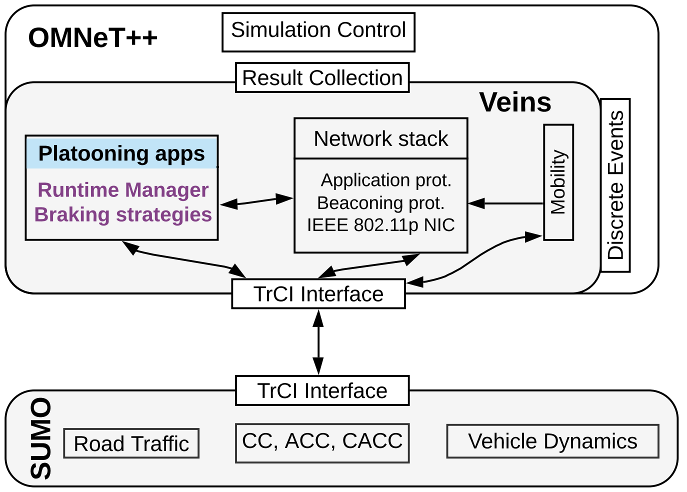


Figure 4.4: Schematic representation of the Plexe simulator [30].

function, a message is sent at the desired simulation time. The message is captured by the `handleSelfMessage` function. The required action to be taken upon reception of such a message can be programmed at the `handleSelfMessage` function.

- *simTime():* This function returns the current simulation time.

4.4.3 Platooning Simulator

In order to simulate the platooning application, a simulator requires to provide vehicular network simulation capability, realistic mobility models, vehicle dynamics, and implementation of platoon controllers. Fortunately, Plexe [30] simulator provides all these features. Hence, Plexe has been used to evaluate all the proposed solutions in this report.

Plexe is an OMNeT++ based simulator built on top of Veins, the VANET simulator. Plexe is tailored explicitly for simulating platooning applications, and it inherits the IEEE 802.11p protocol stack from Veins along with its physical layer channel models, Figure 4.4. In this simulator, a node in the OMNeT++ part is represented by a vehicle in the road traffic simulator SUMO. The communication between the OMNeT++ node and the SUMO vehicle is carried out by TraCI interface [89], a Transmission Control Protocol (TCP) based client/server interface. Platoon controllers are implemented in the SUMO part. Plexe offers necessary functionalities to simulate a wide range of scenarios to mimic real platooning environments and supports code level modifications both for the researchers of control theory and communications. The code level modification is convenient due to the modularity characteristic of the OMNeT++ simulation platform.

The Plexe simulator does not inherently support fail-operational and fail-safe features. For instance, no specialized braking strategy is followed in case of an emergency. To this end, separate modules alongside Plexe are developed to accommodate the runtime manager concept and cooperative emergency braking strategies.

Chapter 5

Platooning Applications for Safety

The platooning applications comprise two modules, i.e., *Runtime Manager* and *Cooperative Emergency Braking.* The idea of the runtime manager and cooperative emergency braking strategies are presented in this chapter. The RM facilitates fail-operational platooning by maintaining certain critical functionalities in the events of transient communication errors by degrading the performance temporarily. In case of emergency, one of the CEB strategies is followed to facilitate fail-safe platooning. Please recall from the Figure 2.1 that the platooning vehicles can encounter hazards at any of the *fuel-efficient*, or *sub-optimal* states in which the communication quality can be either *good*, *fair*, or *poor*. A braking strategy can be regarded as a fail-safe when it can prevent collisions within the platoon, and evade the hazard that causes the emergency braking under all wireless environments. The CEB module further comprises normal braking, gradual deceleration strategy, synchronized braking, coordinated emergency brake protocol, and adaptive emergency braking strategy. The implementation details of the RM and CEB applications in the PlatoonSAFE tool are presented in Chapter 6. The user can independently or jointly activate the RM and CEB modules. Section 5.3.1 in this chapter is published in the following paper[1]:

S. Hasan, A. Balador, S. Girs and E. Uhlemann, "Towards Emergency Braking as a Fail-Safe State in Platooning: A Simulative Approach," *IEEE 90th Vehicular Technology Conference (VTC2019-Fall),* Honolulu, HI, USA, 2019, pp. 1-5. [90].

5.1 Runtime Manager

The failure of V2V communication or hardware failure of the participating vehicles in a platoon cannot be anticipated during the design time. The system performance can be changed anytime during the runtime. Proactive measures must be taken so that the system behaves predictably in the events of failure during the runtime. Hence, the name *Runtime Manager*. The RM is applied to each of the platooning vehicles independently, and it periodically monitors the communication quality perceived by the ego vehicle with respect to its preceding vehicle and the lead vehicle. If the communication quality is changed, the RM upgrades/degrades the platoon performance by adjusting the inter-vehicle gap and/or adopting a suitable controller based on certain predefined safety contracts that are presented in Chapter 6. The assumption/guarantee contracts assure graceful degradation of the performance during runtime and facilitate the transition of the platoon to a known state. Moreover, the RM logs the ego vehicle data periodically, and checks for a safety violation, e.g., whether the gap between the vehicles is shorter than the minimum gap specified in a contract. Should any violations happen, the manager reports the violation.

The performance guarantee that is provided to a platooning vehicle following the safety contracts relies on the communication quality perceived by the vehicle. The communication quality is classified as *good*, *fair*, and *poor*, Figure 2.1. The runtime manager in a vehicle monitors the communication quality both concerning its front vehicle and the lead vehicle. The connection with the front vehicle and the lead vehicle is denoted by *C2F* and *C2L*, respectively. If the number of packet losses experienced by the ego vehicle reaches the *good*, *fair*, or *poor* threshold, a certain performance guarantee is provided according to the assumption/guarantee contracts. The RM offers three types of guarantees, e.g., controller switching, gap adjustment with the front vehicle, or both controller switching and gap adjustment. It is also one of the contributions of this study to determine what number of packet losses define the *good*, *fair*, and *poor* threshold.

The guarantees associated with the safety contracts can mean both upgradation and degradation in the platoon performance. For instance, if a vehicle is in the PLATOON mode, and it needs to increase the gap to the front vehicle or switch to a controller that requires longer gap (e.g., ACC), then the performance of the vehicle is said to be *degraded*. A vehicle *upgrades* its performance if it closes the gap to the front vehicle by either decreasing the gap to the front vehicle or switching to a controller that requires a shorter time gap or constant distance. Moreover, suppose a vehicle does not switch controller but adjusts the gap to the front vehicle. In that case, it is called *Gap Adjustment (GA)*. In addition, there can be situations in which the ego vehicle needs to switch its controller and adjust the gap to the front vehicle at the same time. When the communication quality to the front vehicle and the lead vehicle remains unchanged, the platooning vehicle retains the current state.

5.2 Runtime Manager State Machine

In Figure 2.1, a high-level state machine is presented that illustrates how the platooning vehicles switch between different fail-operational and fail-safe states. The state machine presented in Figure 5.1 is a more detailed version that explicitly defines the controller a platooning vehicle should adapt according to the perceived communication quality. The *fuel-efficient and safe platooning* state in Figure 2.1 is represented by the PLATOON state in Figure 5.1. The *fuel sub-optimal and safe platooning*, and *fuel sub-optimal and safe platooning without communication* states are represented by the CACC and ACC states, respectively in Figure 5.1. In addition, two intermediate states named *PLATOON & GA* and *CACC & GA* have been added. A platooning vehicle transitions to these states when the communication quality is not so poor to change the controller. If a hazard takes places from any of these states, the platooning vehicles would require to transition to the fail-safe state using some specialized emergency braking techniques. Please recall that the controllers proposed in [20], [35], and [19] are used as the PLATOON, CACC, and ACC controllers, respectively. According to Figure 5.1, if a vehicle adopts a state that is on the left of its current state, then the performance of the vehicle *degrades*. In contrast, the performance *upgrades* when a vehicle can transition to a state on the right of its current state.

In PLATOON mode, a vehicle requires *good* communication quality with both the lead vehicle and the front vehicle. Therefore, if the *C2F* is still *good*, but the *C2L* deteriorates to *fair*, a vehicle tries to retain the PLATOON controller. As a precaution, it degrades the performance by increasing the inter-vehicle distance. As long as the *C2F* and *C2L* are *good*, a vehicle maintains the PLATOON mode. In CACC mode, a platooning vehicle requires *good* communication quality with its front vehicle only. To this end, if the *C2F* is *good* disregard to the *C2L*, a vehicle adopts the CACC controller. No matter what the *C2L* is, if the *C2F* is *fair*, a platooning vehicle does not directly transition to the ACC state, rather it further degrades the performance by transitioning to the *CACC & GA* state. Transition to this state is possible from any of the PLATOON or CACC states. If the *C2F* is *poor*, this is considered as temporary communication failure. Therefore, a vehicle adopts the ACC controller that does not rely on V2V communication. However, even in this state, the vehicles keep trying to re-establish the V2V communication.

5.3 Coordinated Emergency Braking

Emergency braking is an integral part of the platooning applications, which is necessitated due to hazards such as the sudden appearance of animals, emergency braking by another vehicle or platoon in the front, etc. On the other hand, emergency braking itself can potentially lead to hazards. Front-and rear-end collision is the most prominent one among these [7]. As the harm posed by platoon-related hazards goes beyond one vehicle, it is of the utmost importance

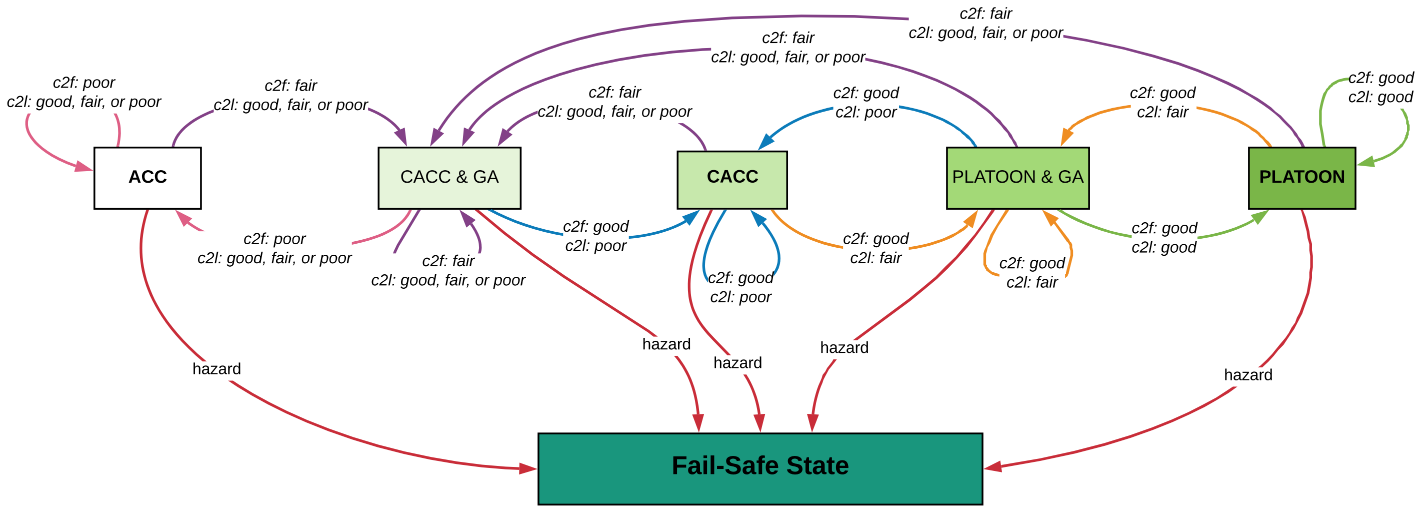


Figure 5.1: State machine representing the switching between different controllers. Update [2022-03-10]: an updated version of the state machine will be made available soon (submitted for reviewing in Feb 2022)

to design fail-safe features in platooning. In a fail-safe state, two conditions are required to be fulfilled. The first condition is that the inter-vehicle distance between the platooning vehicles should be greater than zero when the vehicles are at complete standstill after braking. The second condition dictates that the distance traversed by the lead vehicle since the detection of the hazard should be less than or equal to the distance to the hazard. However, in most recent works, only collision avoidance within the platoon has been regarded as emergency avoidance and claimed as fail-safe [9, 91]. Moreover, maintaining large gaps after full-stop has also been emphasized in some recent works [9, 49], although it is not useful to have large stopping distances if it causes the lead vehicle to traverse longer. While evaluating the CEB strategies, it is examined if both conditions of the fail-safe state are fulfilled. In CEB module, the Normal Braking strategy, Gradual Deceleration strategy, and Coordinated Emergency Brake Protocol are taken from the literature for comparison purposes, and they are briefly described below for the convenience of the reader. The Synchronized Braking and Adaptive Emergency Braking strategies are then elaborately described.

Normal Braking In normal braking, the platooning vehicles start braking as soon as they receive a DENM. The vehicles do not perform any kind of coordination that defines when to brake and how to brake. The braking is done at the full deceleration rate.

Gradual Deceleration Zheng *et al.* [9] proposed that the last vehicle in the platoon should brake at the highest deceleration rate, and the rate should gradually decrease in the upstream direction. The experiment results in [9] show that the platoon avoids collisions but, since the leading vehicle then uses a considerably lower deceleration rate, it traverses longer.

Coordinated Emergency Brake Protocol In [69], the authors presented a Coordinated Emergency Brake Protocol in which the last vehicle starts emergency braking upon reception of a DENM from the LV, and then it sends an ACK. A particular platooning vehicle does not begin braking until it receives ACK from its immediate succeeding vehicle; thus, the LV brakes last and the last vehicle brakes first. However, if packet losses occur, braking is delayed.

5.3.1 Synchronized Braking

The Synchronized Braking strategy can be used on top of the ETSI ITS-G5 protocol stack and does not require any expensive changes in the vehicle model or dynamics. The rationale behind the name is that when the leader detects a road hazard, it does not perform its braking manoeuvre immediately. Rather it disseminates DENM and waits for the following vehicles to be informed about the hazard so that the whole platoon can perform a synchronized braking as depicted in Figure 5.2.

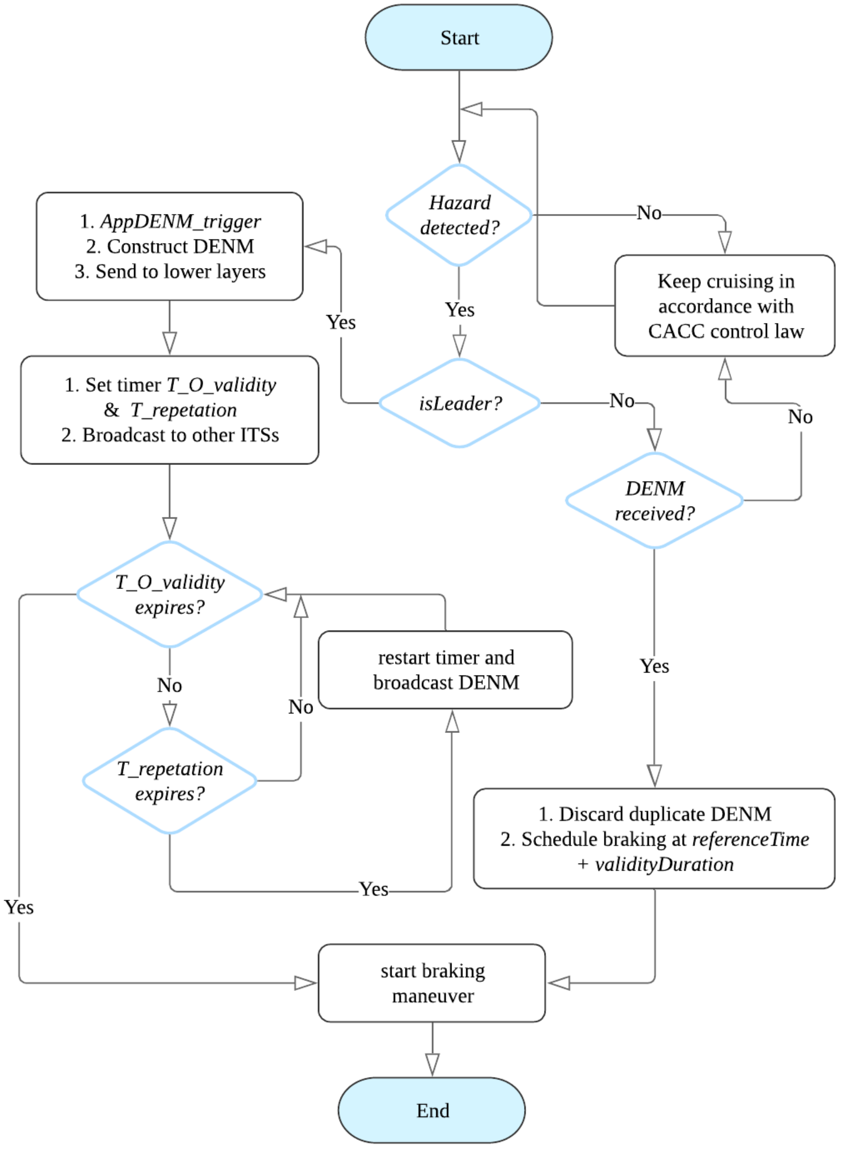


Figure 5.2: The Synchronized Braking strategy. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

Protocol Description

DENM is a facilities layer message which is initiated and terminated in the application layer of an ITS station. Upon detection of a hazard by the platoon leader, the transmission of DENMs is triggered in the application layer, which is regarded as *AppDENM_trigger* in [25]. The data, such as event detection time, position, DENM validity duration, repetition duration and interval, etc., are passed down to the DEN basic service which provides APIs for DENM processing. In this scenario, the *T_O_validity* timer is set to the waiting time (τ_{wait}) that all the platooning vehicles should pursue before they start their braking manoeuvre. In addition, DENM dissemination should be repeated during this entire waiting time which is specified by the parameters *repetition-Duration* and *repetitionInterval* in the application layer. After encoding and processing in the DEN basic service, the message is sent to the lower layers for broadcasting to the other platooning vehicles. Until the following vehicles receive a DENM, they keep cruising in accordance with the active controller. Upon receiving a message, they first check whether the same message has already been received by comparing *actionID* and *stationID*; the DEN basic service discards the message if duplication is found. The vehicles do not perform braking right away despite receiving a DENM. Rather they wait until the *referenceTime* + *T_O_validity* is expired. The extra distance traversed by the platooning vehicles while waiting for the DENMs to be received by all the vehicles is compensated by the high deceleration rate that the synchronized braking facilitates.

DENM Structure

The ETSI-defined DENM format is adopted to implement the SB strategy in the PlatoonSAFE simulator. The ITS PDU header contains information, such as *protocol version*, *messageType* and *vehicleID*. A detailed description of all the fields is out of the scope of this report but can be found in [25]. The mandatory fields and essential information for the considered use case are depicted in Figure 5.3. A new *actionID* is generated for every new event whereas, *repetitionDuration* and *repetitionInterval* specifies how long the message is valid and how often it should be transmitted. According to the standard, message validity can both be terminated by generating *AppDENM_termination* message and an auto timer. In this scenario, the DENM duration is equal to the waiting time before synchronized braking starts. In the situation container, *causeCode 99* corresponds to dangerous situation and *subCauseCode 5* corresponds to automatic emergency braking in the context of platoon emergency braking. Information like speed, position, etc., are appended in the location container field. In the *impactReduction* field, the vehicle data for mitigating collision possibility is included; in case of SB, the deceleration rate at which the whole platoon should brake is attached.

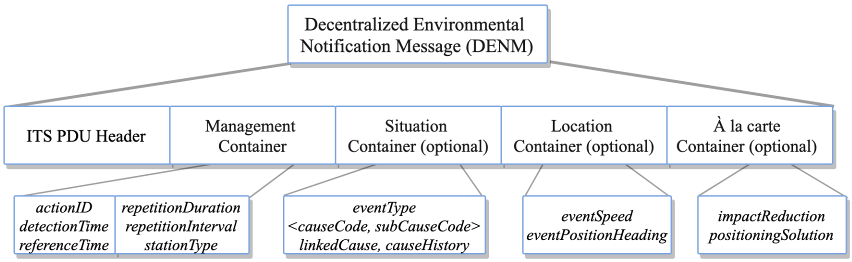


Figure 5.3: DENM structure.

5.3.2 Adaptive Emergency Braking

In Adaptive Emergency Braking, the last vehicle in the platoon brakes first, and the lead vehicle brakes last to avoid collisions. The idea of AEB strategy is depicted in Figure 5.4. The lead vehicle broadcasts DENM upon detection of a road hazard. The DENM is repeated at an interlude of *DENM interval*. Once the DENM is generated, the LV starts soft braking. This is to minimize the stopping distance of the LV. The other platooning vehicles also perform braking at a lower deceleration rate upon reception of a DENM from the LV. Only the last vehicle brakes at the full deceleration rate when it receives a DENM from the LV. The last vehicle also generates ACK messages and broadcasts them. The immediate preceding vehicle of the last vehicle starts full deceleration when it receives an ACK from the last vehicle. Besides, this vehicle generates ACKs and broadcasts them. This way, the ACK is propagated in the upstream direction of the platoon, and the vehicles brake sequentially one after another. A platooning vehicle except the last vehicle does not start braking at the full deceleration rate until it receives an ACK from its immediate following vehicle.

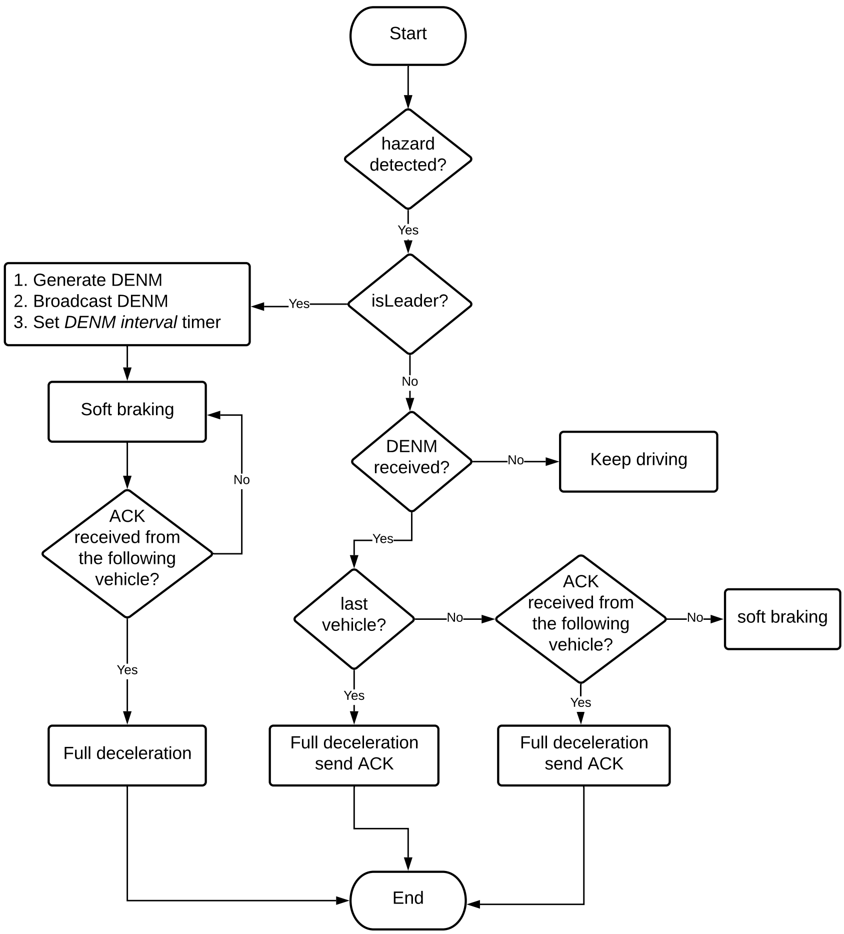


Figure 5.4: The adaptive Emergency Braking strategy

Chapter 6

Simulation Tool: PlatoonSAFE

The implementation details of the PlatoonSAFE tool that comprises the RM and CEB modules are provided in this chapter. The RM and CEB modules are developed on top of the VANET simulator Veins. The PlatoonSAFE tool inherits from the `BaseApp` class of Plexe simulation framework to facilitate the implementation of RM and CEB applications. A schematic diagram of the PlatoonSAFE tool is depicted in Figure 6.1. The logic of the RM and CEB strategies are implemented in the OMNeT++ part. Besides, the simulator provides strong GUI support using the SUMO simulator. For instance, a user can visually see which platooning vehicle is using which controller and the brake light is illuminated when a vehicle brakes, Figure 6.2. The PlatoonSAFE simulator also supports all the result collection features of OMNeT++.

6.1 Runtime Manager Module

This section describes how the RM module is implemented alongside the Plexe simulator. In addition, the proposed assumption/guarantee contracts based on which the RM upgrades or degrades the platoon performance are explained. Furthermore, the implementation details of the contracts and how user-defined contracts can be set in the PlatoonSAFE tool are detailed in this section.

6.1.1 Control Flow: RM

In the PlatoonSAFE simulator, the user is first required to enable the RM module by setting the `rmEnabled` parameter to true. The Plexe simulator can access the RM module through `onPlatoonBeacon` method. This method is invoked every time an ego vehicle receives a beacon from the preceding vehicle or the lead vehicle. The RM mainly performs three tasks: logging of vehicle data, monitoring, and taking actions based on the monitored state. The lead

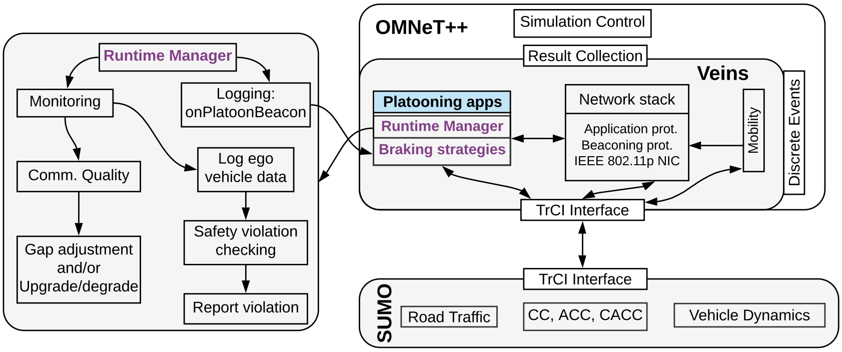


Figure 6.1: Schematic representation of the PlatoonSAFE tool.

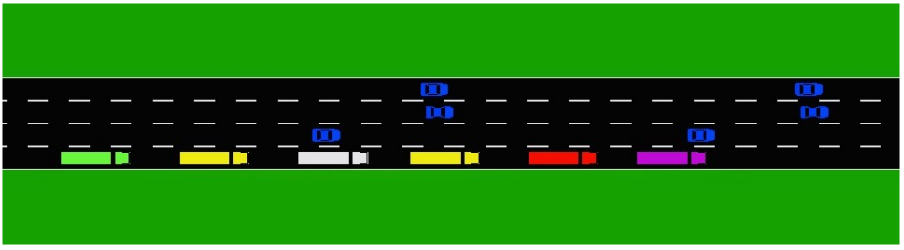


Figure 6.2: Screenshot of SUMO GUI. A platoon of six vehicles in the rightmost lane; different colours of the vehicles represent different active controllers, and the blue vehicles represent non-platooning vehicles.

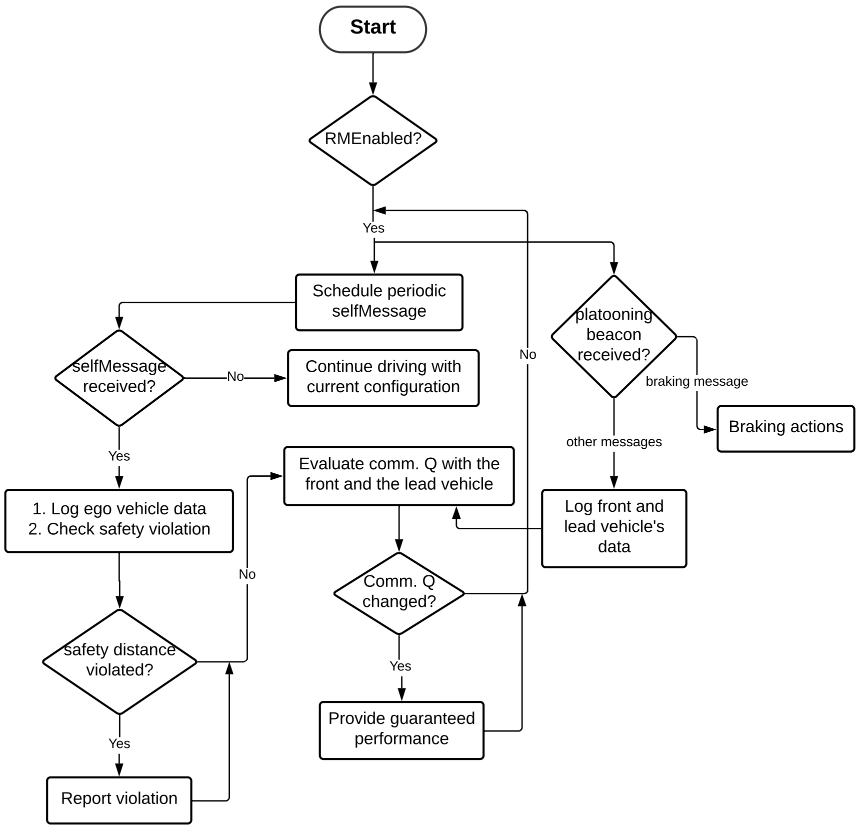


Figure 6.3: Control flow of the runtime manager.

vehicle and front vehicle data are found from the beacons received by the ego vehicle, and delivered through the `onPlatoonBeacon` method. If the received beacon is a braking message, then the simulator module concerning the braking strategies, i.e., CEB is invoked. Otherwise, the data encapsulated in the message is logged, Figure 6.3.

For the monitoring task, the RM uses the `handleSelfMessage` method of OMNeT++. This method is recursively called at an interlude of monitor interval until the simulation time limit is reached. The RM schedules the `selfMessage` every `rmMonitorInterval` seconds. When a `selfMessage` is received, the RM performs three tasks: logging the ego vehicle data, safety violation checking, and communication quality evaluation, Algorithm 1. The recorded ego vehicle data are, currently active controller, maximum deceleration rate, distance to the front vehicle measured by radar, and the current simulation time. These data are used for checking the safety violation. If a predefined safety distance is violated, then the RM reports the violation and the result is recorded in an output file. The RM then evaluates the communication quality concerning the front vehicle and the lead vehicle disregard to the result of the safety violation check, line 5 in Algorithm 1. If the communication quality is changed, a guaranteed platoon performance is ensured according to the assumption/guarantee contract. For instance, when the communication quality becomes poor, a weak contract C = <B,H> can be expressed as shown in Listing 6.1.

Listing 6.1: A weak assumption/guarantee contract

```
Bcacc = Active controller CACC AND no sensor failure AND V2X failure;
Hcacc = Transition to ACC controller;
```

where B_{CACC} is the assumption, and H_{CACC} is the guarantee. Thus, the RM guarantees transition to ACC controller due to *poor* communication quality.

Algorithm 1 `handleSelfMessage` function for logging, monitoring and evaluating

INPUT: cMessage *message

```
1: if message = monitoringMsg then
2:     callBackTime = simTime() + rmMonitorInterval;
3:     egoLog();
4:     safetyViolationCheck();
5:     evaluate();
6:     scheduleAt(callBackTime, monitoringMsg);
7: end if
```

6.1.2 Assumption/Guarantee

`Assumption` represents the current state of the ego vehicle, e.g., currently active controller, C2F vehicle, C2L vehicle etc. For a set of assumptions, the RM performs the task defined by the `Guarantee`. An assumption/Guarantee pair results in a new data type named a `Contract`. Thus, a contract represents a guaranteed performance ensured by the RM for the current state of the ego vehicle.

In the PlatoonSAFE simulator, two base classes named `Assumption` and `Guarantee` are defined as shown in the Listings 6.2 and 6.4. The base class assumption contains the data member named ACTIVE_CONTROLLER that logs the currently employed controller in the ego vehicle. Moreover, another class named `commQAssumption` is derived from the base class that implements the `evaluate` method, and contains the data members such as C2F and C2L to represent communication quality. Each of the C2F and C2L is further divided into three categories, as shown in Listing 6.3. The *good*, *fair*, and *poor* communication quality threshold can be specified by the user in the `.ned` file of OMNeT++. If the number of packet losses experienced by the ego vehicle reaches the `comm_Quality` threshold, a guarantee is provided by the RM according to the contracts.

Listing 6.2: `Assumption` and the derived classes.

```
class Assumption {
public:
    virtual void evaluate(parameter list);
protected:
    ACTIVE_CONTROLLER controller;
};

class commQAssumption : public Assumption {
public:
    void evaluate(parameter list) override;
private:
    C2F c2f;
    C2L c2l;
};
```

Listing 6.3: enum `comm_QUALITY`

```
enum comm_QUALITY {
    POOR,
    FAIR,
    GOOD,
};
```

Listing 6.4: `Guarantee` and the derived classes.

```
enum class GAP2FRONT {
    DEFAULT,
    INCREASE,
    DECREASE,
};
class Guarantee {
public:
    virtual void operator()(std::shared_ptr<Assumption> assumption) const;
};

class ChangeController : virtual public Guarantee {
public:
    void operator()(std::shared_ptr<Assumption> assumption) const override;
protected:
    ACTIVE_CONTROLLER to;
};

class AdjustGap2Front : virtual public Guarantee {
public:
    void operator()(std::shared_ptr<Assumption> assumption) const override;
protected:
    GAP2FRONT gap2front;
};

class ChangeControllerAndAdjustGap2Front : public AdjustGap2Front,
public ChangeController {
public:
    void operator()(std::shared_ptr<Assumption> assumption) const override;
};
```

Three types of guarantees are implemented in the PlatoonSAFE tool, e.g., change the controller, adjust gap with the front vehicle, or both change the controller and adjust the gap. In the base class `Guarantee`, a virtual function named `operator` is defined that can be overloaded by a derived class to facilitate necessary functionalities. This structure is adopted to provide a future extension of the PlatoonSAFE simulator. For instance, if a user is interested in implementing a new guarantee, they need to derive a new class from the assumption base class and override the `operator` function. In the current implementation, there are three derived classes, e.g., `ChangeController`, `AdjustGap2Front`, and `ChangeControllerAndAdjustGap2Front`.

The first two classes are derived from the base class Guarantee, and the `ChangeControllerAndAdjustGap2Front` class uses the `ChangeController` and `AdjustGap2Front` classes as its direct base class. The names of the derived classes reveal their functionalities. As an illustration, the `ChangeController` class changes the currently active controller of the ego vehicle to a more suitable controller based on the communication quality through the TraCI interface. The `AdjustGap2Front` class increases or decreases the distance to the front vehicle, or keeps it as it is if the communication quality remains unchanged. Once the RM performs the Guarantee for an associated Assumption, the `selfMessage` is rescheduled after an interval of `callBackTime`, and the whole process of logging, safety violation checking, and evaluating the state of the ego vehicle is repeated, see Algorithm 1 and Figure 6.3.

6.1.3 Configuration Parameters

A user of the PlatoonSAFE simulator needs to configure a few parameters in the `.ned` file to be able to use the RM module. The parameters are illustrated in the Listing 6.5. First, the RM module has to be enabled using the `rmEnabled` parameter. Otherwise, the simulator will act as the default Plexe simulator. Moreover, the user needs to define the initial gaps to be maintained while using the PLATOON or CACC controller by setting the `platoonSpacing` and `ploegHeadwayTimeGap` parameters, respectively. What number of packet losses are to be regarded as *poor* or *fair* communication quality, also needs to be specified by the user using the `nPacketLossPoor` and `nPacketLossFair` parameters. Furthermore, the factor by which the gap to the front vehicle is to be adjusted while using the PLATOON and CACC controllers is defined by the user by setting `caccHeadwayTimeGapFactor` and `platoonConstantSpacingFactor` parameters, respectively.

Listing 6.5: Configuration settings for Runtime Manager

```
rmEnabled                          = default (false)
platoonSpacing                     = default(5 m)
caccHeadwayTimeGap                 = ploegH
rmMonitorInterval                  = default(.05 s)
nPacketLossPoor                    = default(4)
nPacketLossFair                    = default(2)
minSafetyDistance                  = default(2 m)
caccHeadwayTimeGapFactor           = default(0.25)
```

```
platoonConstantSpacingFactor        = default(0.25)
```

6.1.4 Contracts

The runtime manager comes with a hard-coded Assumption/Guarantee contract list as depicted in Table 6.1. Rigorous simulations were performed to define these contracts. In PlatoonSAFE simulator, this is implemented using the `map` container of C++ Standard Template Library (STL). In map container, the elements are stored in such a way that a `value` is associated with a `key`, and no two elements have the same key. In RM, a unique assumption is used as a key, and the corresponding Guarantee is used as the value. When the RM detects a change in the `comm_QUALITY` of the ego vehicle, it looks for the corresponding Guarantee by iterating through the Assumptions of the contract list. Based on the actions taken by the Guarantee value, it is decided if the RM is to initiate upgraded or degraded performance of a vehicle.

Listing 6.6: Input format for user defined `Contract`.

```
::contract[Type : comma/space separated Assumption variables
: comma/space separated Guarantee variables]
```

Listing 6.7: Order of pairs for `Assumption` component.

```
c2f=value ; c2l=value ; mode=value
```

Listing 6.8: Order of pairs for `Guarantee` component.

```
transition2mode=value ; dist2pred=value
```

Listing 6.9: Two possible combinations of the user-defined contract format.

```
::contract[ctype=value : c2f=value ; c2l=value;
mode=value : transition2mode=value]
::contract[ctype=value : c2f=value ; c2l=value;
mode=value : transition2mode=value ; dist2pred=value]
```

The PlatoonSAFE simulator also supports user-defined contracts. To facilitate user-defined contracts, the RM defines a specific input format, as shown in Listing 6.6. A parser converts the input into an object of the C++ `Contract` class type. If the users define their own contract list, the default contract list is ignored by the RM.

The components of the Assumption and Guarantee are defined as a collection of `key` and `value` pair. For instance, the key and value pairs for the current list of contracts are listed in Table 6.2. The user has to provide all three pairs of assumption components according to the ordering depicted in Listing 6.7. For guarantee component, the user can choose between either of the pairs. Listing 6.8 depicts the order of Guarantee pairs that should be followed in case the user chooses both the available pairs.

Action Type	Assumption			Guarantee	
	c2f	**c2l**	**mode**	**Transition2mode**	**dist2pred**
Degradation	GOOD	POOR	PLATOON & GA	CACC	-
	POOR	GOOD	CACC & GA	ACC	-
	POOR	FAIR	CACC & GA	ACC	-
	POOR	POOR	CACC & GA	ACC	-
	FAIR	GOOD	CACC	CACC & GA	INCREASE
	FAIR	FAIR	CACC	CACC & GA	INCREASE
	FAIR	POOR	CACC	CACC & GA	INCREASE
	GOOD	FAIR	PLATOON	PLATOON & GA	INCREASE
	FAIR	GOOD	PLATOON	CACC & GA	INCREASE
	FAIR	FAIR	PLATOON	CACC & GA	INCREASE
	FAIR	POOR	PLATOON	CACC & GA	INCREASE
	FAIR	GOOD	PLATOON & GA	CACC & GA	INCREASE
	FAIR	FAIR	PLATOON & GA	CACC & GA	INCREASE
	FAIR	POOR	PLATOON & GA	CACC & GA	INCREASE
	c2f	**c2l**	**mode**	**Transition2mode**	**dist2pred**
Upgradation	GOOD	POOR	CACC & GA	CACC	DECREASE
	GOOD	GOOD	PLATOON & GA	PLATOON	DECREASE
	FAIR	GOOD	ACC	CACC & GA	-
	FAIR	FAIR	ACC	CACC & GA	-
	FAIR	POOR	ACC	CACC & GA	-
	GOOD	FAIR	CACC	PLATOON & GA	-
	c2f	**c2l**	**mode**	**Transition2mode**	**dist2pred**
Self states	GOOD	GOOD	PLATOON	PLATOON	DEFAULT
	GOOD	FAIR	PLATOON & GA	PLATOON & GA	DEFAULT
	GOOD	POOR	CACC	CACC	DEFAULT
	FAIR	GOOD	CACC & GA	CACC & GA	DEFAULT
	FAIR	FAIR	CACC & GA	CACC & GA	DEFAULT
	FAIR	POOR	CACC & GA	CACC & GA	DEFAULT
	POOR	GOOD	ACC	ACC	DEFAULT
	POOR	FAIR	ACC	ACC	DEFAULT
	POOR	POOR	ACC	ACC	DEFAULT

Table 6.1: Default Assumption/Guarantee contract list for the Runtime Manager; GA stands for Gap Adjustment.

Component	**key**	**value**
`Type`	ctype	wifi
`Assumption`	c2f c2l mode	GOOD/FAIR/POOR GOOD/FAIR/POOR ACC/CACC/PLATOON
`Guarantee`	transition2mode dist2pred	ACC/CACC/PLATOON DEFAULT/INCREASE/DECREASE

Table 6.2: Available `keys` and associated `value` for user defined `Contract`.

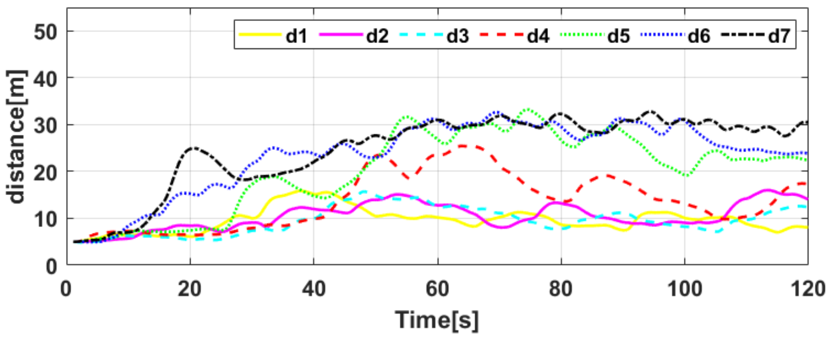


Figure 6.4: The distance profiles of the platooning vehicles; *fair* = 2, *poor* = 4; ACC CTG = 2 s, CACC CTG = 1 s, PLATOON CDG = 5 m.

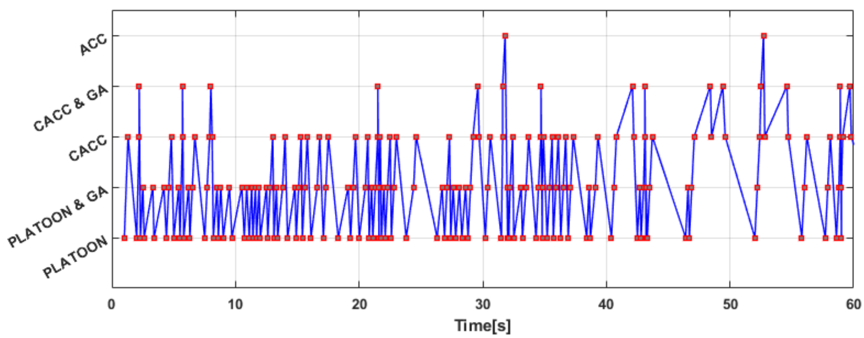


Figure 6.5: How vehicle 5 switches between different states using the runtime manager; *fair* = 2, *poor* = 4; ACC CTG = 2 s, CACC CTG = 1 s, PLATOON CDG = 5 m. GA stands for Gap Adjustment.

Based on the rules defined above, there are two possible formats that a user can use to define their contracts, and it is depicted in Listing 6.9. The first format is to be used when one of the guarantees is provided, and the second one is to be used when both of the guarantees are provided.

6.1.5 Sample Use Case: RM

In this part, a sample use case is considered to demonstrate how the RM module functions. To this end, a platoon of eight vehicles and fifty non-platooning vehicles are simulated that generate high-level of interference throughout the simulation time. Figure 6.4 depicts the inter-vehicle distance profiles of the platooning vehicles; the inter-vehicle distance between the vehicles v_{i-1} and v_i is denoted by d_i. Figure 6.5 shows how vehicle 5 switches between different states. Here, the results of only 60 s time window and vehicle 5 are illustrated to make the state switching better visible in the figure. In both figures, the thresholds for *fair* and *poor* communication quality are chosen to be 2 and 4 packet losses, respectively. The details of the communication and controller parameters, and the choice of *good*, *fair*, and *poor* communication quality will be discussed in the subsequent chapters.

The first thing to notice in Figure 6.4 is that the vehicles avoid rear-end collisions despite transient communication errors. When the perceived communication quality is different for different vehicles with respect to the lead vehicle, the vehicles can frequently switch between the controllers as a response to the communication error. Thus, the platoon acts like a decentralized system when the situation demands. For instance, during the 60–80 s time window, the front vehicles in the platoon maintain very short inter-vehicle distance, whereas the tail vehicles maintain a gap of approx. 30 m due to transient communication errors.

Figure 6.5 allows us to closely look at how vehicle 5 uses the RM to tackle the temporary communication error problem. During the 60 s time window, the vehicle has switched between all five states. When the communication quality is *good* with respect to the leader, the vehicle stays in the PLATOON mode. When the communication quality deteriorates, the RM can respond to that very fast, and degrade the performance. One important point to notice here that a vehicle does not directly switch from the PLATOON mode to ACC mode for example due to 4 packet losses. It must have experienced 2 packet losses before that and thereby, transitioned to some intermediate state, e.g., CACC, CACC & GA. This is why no direct transitions between PLATOON and ACC states, PLATOON and CACC states are shown in Figure 5.1. This contradicts with the Figure 3 presented in Paper [75] in which the authors suggest direct transition between the PLATOON and ACC states using degradation cascades.

6.2 Coordinated Emergency Braking Module

The CEB module is implemented as a separate module in the PlatoonSAFE simulator like the RM module. The implementation details of the emergency braking strategies described in Section 5.3 is presented in this section. Please note that if the RM module is disabled, and the CEB module is enabled, then the vehicles would not do the state switching according to the safety contracts of RM before emergency braking. In that case, the vehicles would stick to the predefined controller.

6.2.1 Control Flow: CEB

The control flow of the CEB module is rather simple and depicted in Figure 6.6. A user is required to define if the CEB module is to be enabled explicitly. If the CEB module is enabled, the user must further define which of the four braking strategies to be used by the platooning vehicles. The user can choose

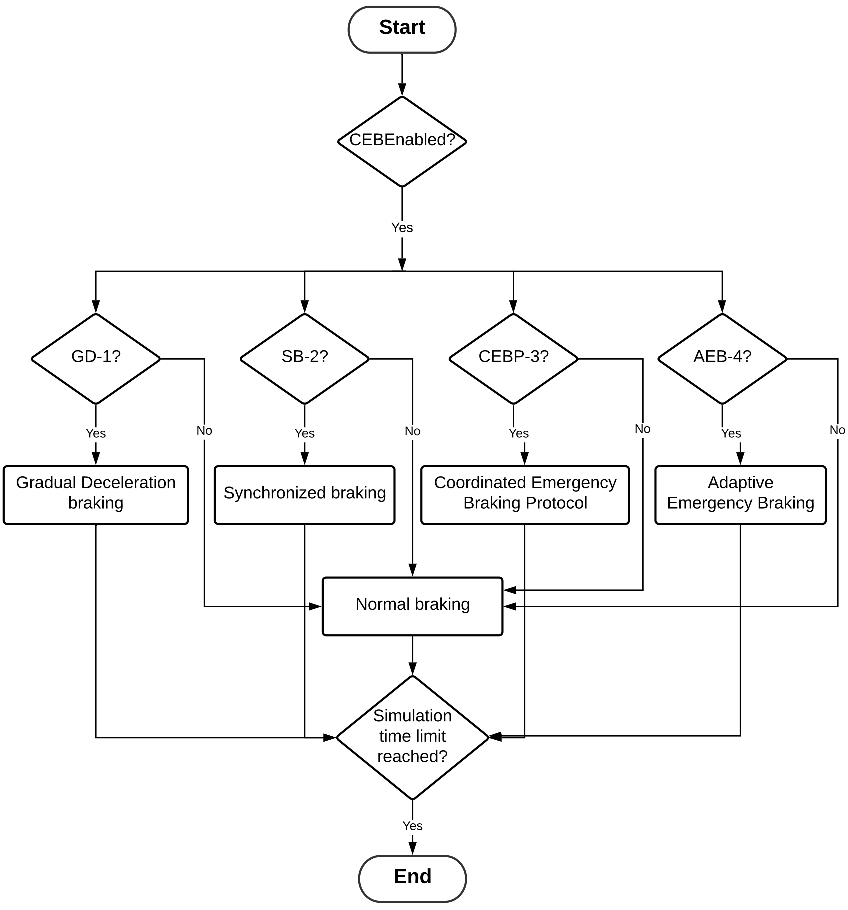


Figure 6.6: Control flow of the Coordinated Emergency Braking module.

between the four braking strategies, e.g., GD-1, SB-2, CEBP-3, and AEB-4. If the user chooses none of these braking strategies, the PlatoonSAFE simulator uses the normal braking strategy by default. The simulation runs until the simulation time limit is reached unless otherwise interrupted by the user. The choice of the braking strategies can be set during runtime in the PlatoonSAFE simulator, thanks to the OMNeT++ GUI.

6.2.2 Normal Braking Realization

The Plexe simulator comes with the normal braking scenario by default, and the implementation details are depicted in Algorithm 2. The user specifies when the platoon should start braking, and at what deceleration rate using the parameters `brakeAtTime` and `decelerationRate`, respectively. If the current simulation time is greater than the scheduled braking time, then a braking message is scheduled at the current simulation time using the `scheduleAt` function. Otherwise, the braking message is scheduled at the time when the braking manoeuvre should be performed. In either case, the braking message is received by the `handleSelfMsg` function, and the vehicle is set with the desired deceleration rate using the `setFixedAcceleration` function. In this algorithm, the developers of the Plexe simulator do not use DENMs to coordinate the braking, i.e., the vehicles do the braking using the periodic CAMs that are used for cruising the platoon.

Algorithm 2 Normal Braking Strategy

INPUT: brakeAtTime, decelerationRate, leaderSpeed

```
procedure INITIALIZE
    if vehicleId = leaderId then
        if simTime() > brakeAtTime then
            startBraking = simTime();
            scheduleAt(simTime(), brakingMsg);
        else
            scheduleAt(startBraking, brakingMsg);
        end if
    end if
end procedure
procedure HANDLESELFMSG (MSG)
    if msg = brakingMsg then
        traciVehicle->setFixedAcceleration(1, -decelerationRate);
    end if
end procedure
```

6.2.3 Realization of Gradual Deceleration Strategy

Please recall that in Gradual Deceleration strategy, the last vehicle in the platoon brakes with the highest deceleration rate, and the rate gradually decreases in the upstream direction [9]. As a result, the lead vehicle brakes with the lowest deceleration rate among all the platooning vehicles. The implementation of this strategy does not require any coding in addition to the normal braking strategy. The user only needs to set the vehicle-specific deceleration rates in the `.ini` file. An example is demonstrated in Listing 6.10.

Listing 6.10: Configuration settings for Gradual Deceleration strategy.

```
*.node[*].scenario.brakeAtTime        = 20 s
*.node[4].scenario.decelerationRate   = 8 mpsps
*.node[3].scenario.decelerationRate   = 7 mpsps
*.node[2].scenario.decelerationRate   = 6 mpsps
*.node[1].scenario.decelerationRate   = 5 mpsps
*.node[0].scenario.decelerationRate   = 4 mpsps
```

6.2.4 Realization of Synchronized Braking

Algorithm 3 Synchronized Braking Strategy

```
Require: brakeAtTime, DENMInterval, isSBEnabled = true, waitingTime;
repeat for every vehicle
procedure INITIALIZE
    if myVehicleId = leaderId then
        if simTime() > brakeAtTime then
            scheduleAt(simTime(), sendDENM);
        else
            scheduleAt(brakeAtTime, sendDENM);
        end if
    end if
end procedure
procedure SENDDENM
    broadcastDENM(eventAttributes);
    scheduleAt(simTime() + DENMInterval, sendDENM);
end procedure
procedure ONDENM(DENM)
    if DENMSet.count(myVehicleId) then
        delete DENM;
    else
        DENMSet.insert(myVehicleId);
        scheduleAt(brakeAtTime + waitingTime, synBrkMsg);
    end if
end procedure
procedure HANDLESELFMSG
    setFixedAcceleration(1, −decelerationRate);
end procedure
```

The implementation of the SB strategy is depicted in Algorithm 3. In the initialization, a DENM self message is scheduled at the time of the imaginary road hazard. The DENM dissemination is repeated at an interval of `DENMInterval`. When a vehicle receives a DENM, it first checks if a DENM has already been received. If not, a braking message is scheduled after the `waitingTime` from the `brakeAtTime`, line 20 in Algorithm 3. The vehicles perform the braking at the desired deceleration rate when the `synBrkMsg` is received in the `handleSelfMsg` method.

6.2.5 Realization of Adaptive Emergency Braking

The pseudocode of the AEB strategy is presented in Algorithm 4. The algorithm first checks whether the current vehicle is the LV. If it is, a DENM broadcast is scheduled at a specific time step in which the LV detects an imaginary hazard using the *scheduleAt* function of OMNeT++, line 7. In the $sendDENM$ procedure, the LV broadcasts the DENM that contains event attributes. This procedure is recursively called at an interlude of $DENMInterval$. In the meantime, the LV also schedules the initiation of soft braking after a *brakeLag* period. The $onDENM$ procedure (line 19) takes necessary actions upon reception of a DENM by a vehicle. It first checks if a DENM has already been received by this vehicle. If the vehicle is the last in the platoon and it has begun decelerating by tracking its preceding vehicle based on the radar measurements, then it performs full deceleration directly, line 28. Otherwise, the vehicle first prepares the brake for *brakeLag* duration, then executes full deceleration. In any case, the Last vehicle schedules an ACK broadcast. If the DENM receiving vehicle is not the last, then soft deceleration is scheduled after the *brakeLag* period unless it has already started slow down tracking the preceding vehicle, line 35. The soft deceleration continues until the vehicle receives an ACK from its immediate FV. Just like *sendDENM* procedure, the $sendACK$ procedure is also recursively called every $ACKInterval$ seconds as indexed by line 40. When a vehicle receives an ACK packet, it checks if the packet is sent by its immediately following vehicle and if this is the first packet received from that vehicle. In that case, the vehicle intervenes soft deceleration and actuates braking at full force, see lines 44-54. The vehicle also calls the $sendACK$ procedure to inform the preceding vehicle about its full brake manoeuvre.

In the AEB algorithm, full deceleration is only initiated upon reception of an ACK packet except for the last vehicle, whereas DENM triggers soft deceleration, also except for the last vehicle. The last vehicle directly performs full deceleration on DENM. These two functions are performed by $onFullDeceleration$ and $onSoftDeceleration$ procedures, lines 55 and 58. The deceleration is achieved by a built-in function of Plexe simulator named $setFixedAcceleration$. In the $onSoftDeceleration$ procedure, it must also be ensured that a full deceleration manoeuvre does not get interrupted by soft deceleration if a vehicle already receives an ACK while preparing the brake.

Algorithm 4 Adaptive Emergency Braking Strategy

```
Require: brakeAtTime, DENMInterval, ACKInterval, brakeLag, softDecelerationRate, fullDecelerationRate, DENMSet, ACKSet, isAdaptiveBrakeEnabled = true, flag = true;
repeat for every vehicle
procedure INITIALIZE
    if myVehicleId = leaderId then
        if simTime() > brakeAtTime then
            scheduleAt(simTime(), sendDENM);
        else
            scheduleAt(brakeAtTime, sendDENM);
        end if
    end if
end procedure
procedure SENDDENM
    broadcastDENM(eventAttributes);
    scheduleAt(simTime() + DENMInterval, sendDENM);
    if flag then
        scheduleAt(simTime() + brakeLag, softDeceleration);
        flag = false;
    end if
end procedure
procedure ONDENM(DENM)
    if DENMSet.count(myVehicleId) then
        delete DENM;
    else
        DENMSet.insert(myVehicleId);
        if myVehicleId = getPlatoonSize() − 1 then
            if vehicleData.acceleration < 0 then
                scheduleAt(simTime() + brakeLag, fullDeceleration);
            else
                scheduleAt(simTime(), fullDeceleration);
            end if
            scheduleAt(simTime(), sendACK);
        else
            if vehicleData.acceleration < 0 then
                scheduleAt(simTime(), softDeceleration);
            else
                scheduleAt(simTime() + brakeLag, softDeceleration);
            end if
        end if
    end if
end procedure
```

```
procedure SENDACK
    broadcastACK(getVehicleId());
    scheduleAt(simTime() + ACKInterval, sendACK);
end procedure
procedure ONACK(ACK)
    if ACKPacket.getVehicleId() = myVehicleId + 1 then
        if ACKSet.count(myVehicleId) then
            delete ACK;
        else
            ACKSet.insert(myVehicleId);
            scheduleAt(simTime(), fullDeceleration);
            scheduleAt(simTime(), sendACK);
        end if
    end if
end procedure
procedure ONFULLDECELERATION(ACK)
    setFixedAcceleration(1, −fullDecelerationRate);
end procedure
procedure ONSOFTDECELERATION(DENM)
    if ACKSet.count(myVehicleId) = 0 then
        setFixedAcceleration(1, −softDecelerationRate);
    end if
end procedure
until simulation time-limit reached
```

To this end, the algorithm first checks if a vehicle has already received an ACK, line 59.

6.2.6 Coordinated Emergency Brake Protocol Realization

The implementation of the CEBP strategy proposed in [69] is similar to the AEB strategy. In CEBP, the vehicles do not perform soft deceleration upon reception of the DENMs; they perform full deceleration when an ACK is received from the immediately following vehicle. To enable the CEBP strategy, the user needs to set the parameters in Listing 6.11. Besides, the other braking strategies are required to be disabled.

Listing 6.11: Configuration settings for CEBP strategy.

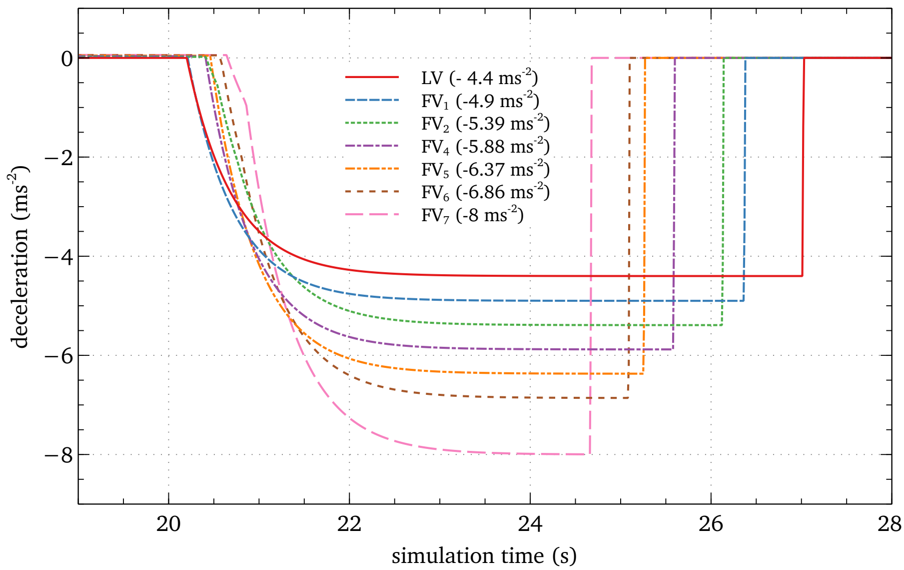
```
isCEBPEnabled      = default (false)
brakeAtTime        = default(20 s)
decelerationRate   = default(8 mpsps)
DENMInterval       = default (50 ms)
ACKInterval        = default (50 ms)
brakeLag           = default (200 ms)
```

6.2.7 Sample Use Case: CEB

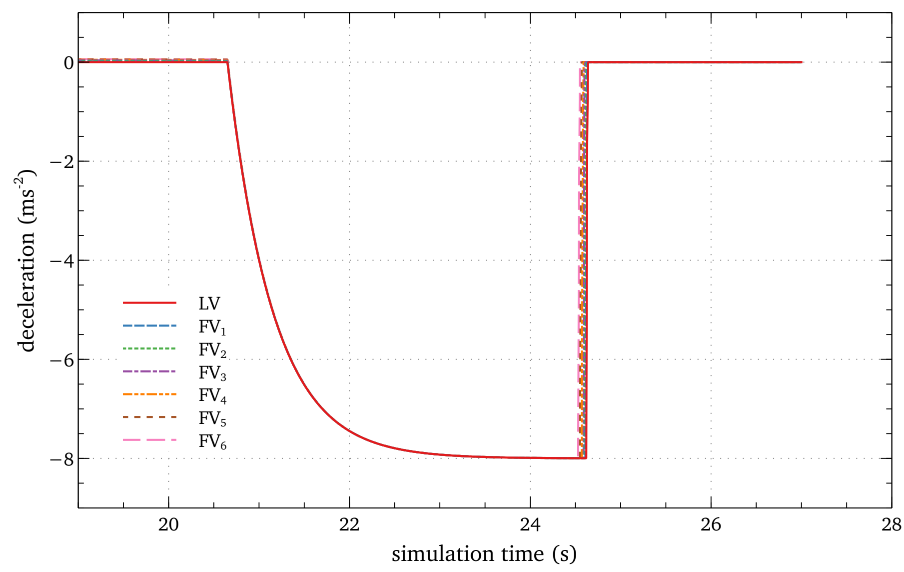
In this part, a sample use case is considered to demonstrate how the platooning vehicles perform emergency braking using the Cooperative Emergency Braking strategies described above. To this end, simulations are carried out with a platoon of seven vehicles that use the PLATOON controller. At 20 s of the simulation time, an imaginary road hazard is detected by the LV, and it starts broadcasting DENMs. The DENMs are not relayed by any other platooning vehicles. However, the ACKs are sent/relayed by every vehicle starting from the last vehicle for the AEB and CEBP strategies. Six hundred non-platooning vehicles are introduced to model the neighbouring traffic that causes a high level of interference. This is the same simulation scenario as the config-1 of Section 8.3.1 in Chapter 8.

Figure 6.7 shows the acceleration profiles of the platooning vehicles for the four different braking strategies. LV stands for Lead Vehicle, and FV_i stands for the Following Vehicle i. For GD-1 in Fig. 6.7a, where the individually selected deceleration rates depend on the vehicle's position in the platoon, the deceleration rates have been set following [9]. Despite a high delay before reception of its first DENM, the last vehicle manages to come to a complete standstill before the other vehicles by pursuing the highest deceleration rate (-8 ms^{-2}), and the LV stops last by performing braking at a rate of -4.4 ms^{-2}.

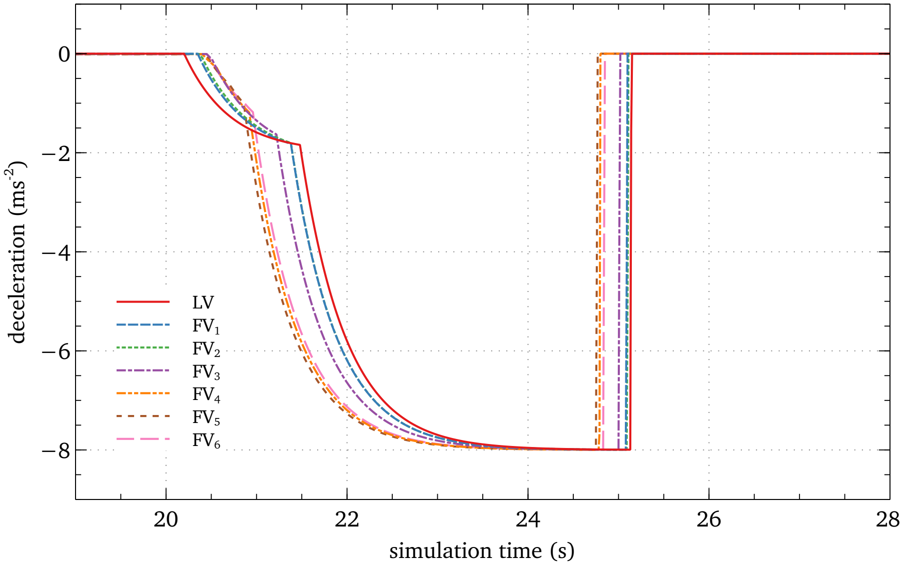
In SB-2, all the vehicles wait for $\tau_{wait} = 625$ ms before braking, where 625 ms is the average DENM delay of FV_6. All vehicles receive the DENMs successfully by this time and perform synchronized braking with full deceleration rate (-8 ms^{-2}), Fig. 6.7b.



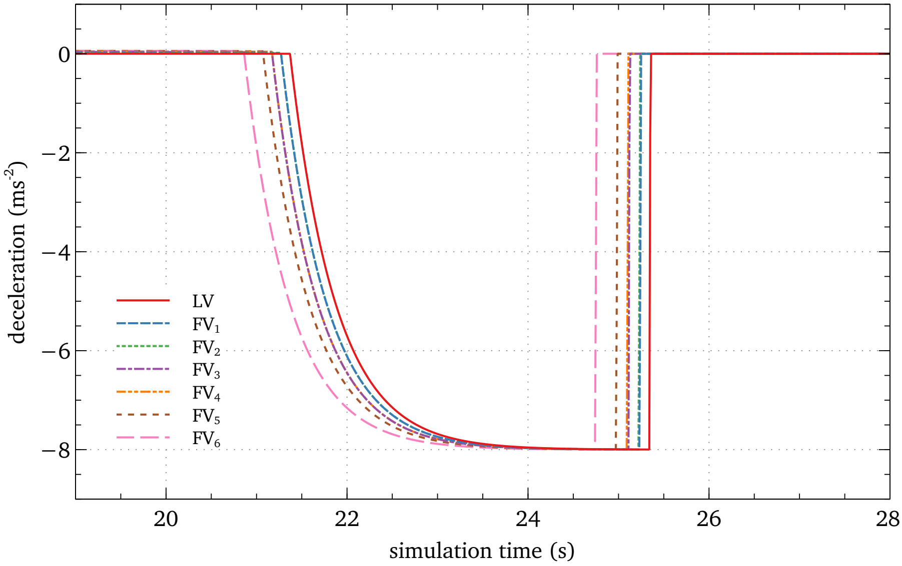
(a) Gradual Deceleration



(b) Synchronized Braking, $\tau_{wait} = 625\ ms$, decelerationRate $= -8\ ms^{-2}$.



(c) Adaptive Emergency Braking, softDecelerationRate = $-2\ ms^{-2}$, fullDecelerationRate = $-8\ ms^{-2}$.



(d) Coordinated Emergency Brake protocol, decelerationRate = $-8\ ms^{-2}$.

Figure 6.7: Acceleration profiles of the platooning vehicles for different braking strategies.

In CEBP-3, the last vehicle starts braking first upon reception of a DENM from the LV, Fig. 6.7d. The LV brakes last once it receives ACK from the FV_1. In this case, all the vehicles brake with a deceleration of rate $-8\ ms^{-2}$.

The acceleration profiles of the platooning vehicles for the AEB-4 strategy are presented in Fig. 6.7c. In AEB-4, the LV broadcasts DENM and starts soft deceleration at a rate of $-2\ ms^{-2}$. All the FVs except FV_6 also perform soft deceleration upon reception of the first DENM from the LV. Once the last vehicle receives a DENM, it performs full deceleration at a rate of $-8\ ms^{-2}$ and sends ACK in the upstream direction. The ACK propagates sequentially, and the vehicles which receive it from its FV also perform full deceleration in order. In Fig. 6.7c, it seems like the last vehicle, FV_6, also performs soft deceleration, which it is not supposed to do. What happens in this particular scenario is that the FV_6 does not receive a DENM until 960 ms, but FV_5 receives it long before, and it starts soft deceleration. Although the FV_6 loses V2V communication temporarily, it still has its radar sensor to measure the distance to the preceding vehicle (FV_5). The PLATOON controller uses this information to maintain the desired gap (gap_{des}) with FV_5. It takes 1480 ms in total for the LV to receive its first ACK in this scenario, and during this time, the LV performs soft deceleration intending to minimize its stopping distance.

Chapter 7

Evaluation of Fail-Operational Platooning

In this chapter, the efficacy of the Runtime Manager in maintaining Fail-operational platooning is analyzed. To this end, simulation results are first presented to understand the effects of packet losses on *good*, *fair*, and *poor* communication quality based on which a platooning vehicle switches between the controllers, and/or adjusts the inter-vehicle distance as illustrated in the state machines in Figures 2.1, 5.1. In addition, rigorous simulations have been carried out to demonstrate how runtime manager can maintain certain critical platooning functionalities in the events of transient communication errors by gracefully degrading the performance instead of abolishing the V2V communications between the platooning vehicles. The simulation results presented in this chapter do not consider any specialized braking strategy, i.e., the vehicles perform normal braking only. A part of the work presented in this chapter is published in the following papers[1][2]:

S. Hasan, A. Balador, S. Girs and E. Uhlemann, "Towards Emergency Braking as a Fail-Safe State in Platooning: A Simulative Approach," *IEEE 90th Vehicular Technology Conference (VTC2019-Fall),* Honolulu, HI, USA, 2019, pp. 1-5. [90].

S. Hasan, M. A. Al Ahad, I. Sljivo, A. Balador, S. Girs and E. Lisova, "A Fault-Tolerant Controller Manager for Platooning Simulation," *IEEE International Conference on Connected Vehicles and Expo (ICCVE),* Graz, Austria, 2019, pp. 1-6. [92].

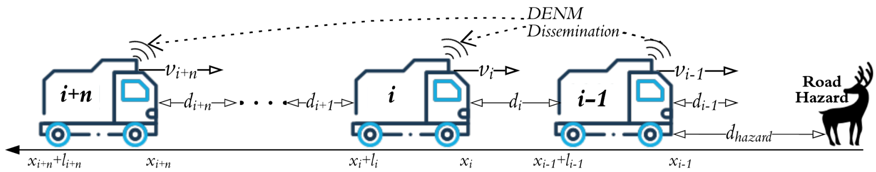


Figure 7.1: Platoon model. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

7.1 Platoon Model

Let us consider a homogeneous platoon where the i^{th} vehicle is moving at a constant speed of v_i as illustrated in Figure 7.1. Here, x_i and l_i are the position of the front bumper and the length of the i^{th} vehicle, respectively. The desired distance D_i^{des} between the i^{th} vehicle in the platoon and its predecessor $i-1$ can, according to [93], [53] be given by:

$$D_i^{des}(t) = D_i^{st} + h_t v_i(t), \tag{7.1}$$

where D_i^{st} is the distance at standstill and h_t is the constant *time headway* defined as the time required by the front of the i^{th} vehicle to reach the point on the road where the front of its predecessor $i-1$ currently is. *Space headway* h_s, on the other hand, is the difference in position between the fronts of the i and $(i-1)^{th}$ vehicles which can be expressed as $x_{i-1}(t) - x_i(t)$. So, the actual distance d_i between the i and $(i-1)^{th}$ vehicle is:

$$d_i(t) = h_s(t) - l_{i-1}. \tag{7.2}$$

A platoon is said to have a rear-end collision if $d_i(t) < 0$. The space error ε_i of vehicle i can therefore be defined as the difference between the actual and desired distances:

$$\varepsilon_i(t) = d_i(t) - D_i^{des}(t). \tag{7.3}$$

Finally the velocity error $\dot{\varepsilon}_i$ of the i^{th} vehicle with respect to the leader can be formulated as:

$$\dot{\varepsilon}_i(t) = v_i(t) - v_L(t), \tag{7.4}$$

where v_L is the velocity of the leader. While platooning, it is essential to have $\varepsilon_i(t)$ and $\dot{\varepsilon}_i(t)$ as low as possible to minimize the tracking error, and maintain string stability. While emergency braking, however, the necessary conditions for avoiding rear-end collision and the road hazard ahead are $d_i(t) > 0$ and $d_L \leq d_{hazard}$ respectively, where d_L and d_{hazard} are the distance traversed by the leader since the braking manoeuvre started and the distance to the upcoming hazard, respectively. These are the necessary conditions for fail-safe platooning.

7.2 Simulation Scenario and Settings

A platoon of length eight is considered, and the lead vehicle drives at a speed of 100 kmph. Besides, the lead vehicle oscillates its speed at a frequency of 0.2 Hz with oscillation amplitude 10 kmph. The following vehicles try to obey the speed of the lead vehicle. In addition, there are 50 non-platooning vehicles on the highway that surrounds the platoon. A highway of 4 lanes is considered, and the platoon cruises in the first lane. The platooning vehicles can choose between the PLATOON, CACC, and ACC controllers. Various scenarios with different time gaps (ACC, CACC), and constant distance gaps (PLATOON) have been simulated. The standard IEEE 802.11p communication parameters are followed in our simulations, Table 7.1. However, the beacon interval of the non-platooning vehicles does not follow the standard (100 ms). This is to introduce a scenario in which there would be packet losses; recall that the runtime manager concept tackles the transient communication error problem in platooning application. A similar packet loss scenario could also be introduced by using a vast number of neighbouring vehicles, and a reasonable beaconing interval.

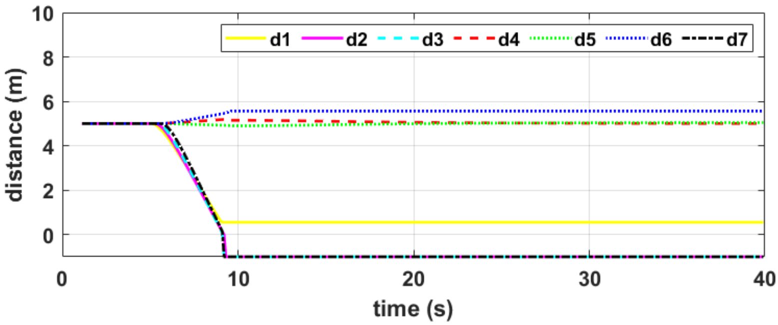
In PlatoonSAFE simulator, the runtime manager is activated by setting the `rmEnabled` parameter to true. In addition, the coordinated emergency braking strategies have been disabled by setting the `cebEnabled` parameter to false. Therefore, the results presented in this chapter consider normal braking strategy only, when required. The safety violation checking and the monitoring of the packet losses is performed at an interval of 0.1 s. If a vehicle does not require to switch controller, but it requires to adjust the gap to the front vehicle, then the factor by which the gap is increased or decreased is 0.25. Rigorous simulations have been performed with different thresholds for `fair` and `poor` communication qualities by tuning the `nPacketLossFair` and `nPacketLossPoor` parameters.

7.3 Impact of Packet Losses on *good*, *fair*, and *poor*-threshold

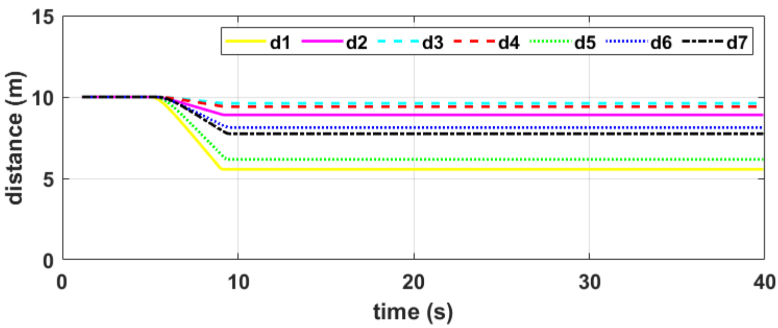
Let us start with identifying the scenarios that exhibit rear-end collisions while the platooning vehicles use the PLATOON controller only, e.g., Figure 7.2a, 7.2c. To this end, a braking scenario is simulated in which the lead vehicle starts braking at the 10^{th} s of simulation time. Figure 7.2 presents the bumper to bumper distance profiles of the platooning vehicles with initial constant distance gap of 5, 10, and 15 m. With 5 m initial gap, some of the platooning vehicles undergo rear-end collisions, Figure 7.2a. This is due to short inter-vehicle distances, and a high number of packet losses. For the same simulation scenario with 10 m initial gaps, the platooning vehicles can avoid collisions. Different vehicles start braking at different times, as they receive the braking message at different times. As a result, the inter-vehicle distances at complete

	Parameter	Value
communication	PHY/MAC model	IEEE 802.11p/1609.4
	Path loss model	Free space ($\alpha = 2$)
	TxPower	100 mW
	Packet size	200 B
	Bit rate	6 $Mbps$
	Sensitivity	-94 dBm
	Thermal noise	-95 dBm
	Frequency	5.89 GHz
	Bit rate (non-platooning vehicles)	3 $Mbps$
	Beacon interval (platooning vehicles)	0.1 s
	Beacon interval (non-platooning vehicles)	0.005 s
mobility	Leader speed	100 $kmph$
	Platoon size	8
	Non-platooning vehicles	50
	No. of platoons	1
	Leader oscillation frequency	0.2 Hz
	Oscillation amplitude	10 $kmph$
	Total no. of lanes	4
controller	Controllers	PLATOON, CACC, ACC
	Engine lag τ	0.5 s
	Weighting factor C_1	0.5 s
	Controller bandwidth ω	0.2 Hz
	Damping factor	1
	Headway distance	$3 \sim 20$ m
RM	`rmEnabled`	*true*
	`rmMonitorInterval`	0.1 s
	`platoonConstantSpacingFactor`	0.25
	`caccheadwayTimeGapFactor`	0.25
	`nPacketLossPoor`	$2, 3, 4$
	`nPacketLossFair`	$1, 2, 3$

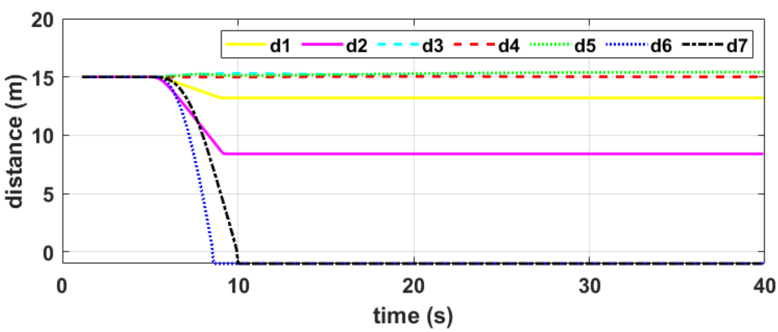
Table 7.1: Configuration parameters for simulation and analysis. Copyright © 2019, IEEE, reprinted from Hasan *et al.* ICCVE'2019.



(a) PLATOON constant distance gap = 5 m



(b) PLATOON constant distance gap = 10 m



(c) PLATOON constant distance gap = 15 m

Figure 7.2: Inter-vehicle distance profiles of the platooning vehicles; no runtime manager, and different constant distance gaps for the PLATOON controller.

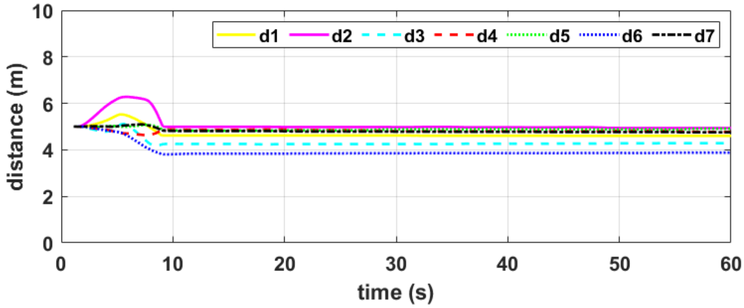
standstill is not the same for all the vehicles. These two scenarios suggest that the vehicles with even the PLATOON controller can undergo collisions in dense traffic scenarios. To avoid this collision, larger initial inter-vehicle distance is required. However, a too large gap is not feasible either. For instance, the last two platooning vehicles in Figure 7.2c undergo collisions despite 15 m initial gap. This happens because the last two vehicles cannot successfully receive the DENMs transmitted by the lead vehicle as they are too far away (the path loss effect). Besides, the messages from the predecessors take time to propagate, and the radar sensor has detection, processing, and actuation lags.

The RM has been tested for the above three scenarios with different combinations of `nPacketLossPoor` and `nPacketLossFair` values. In order to keep the discussion concise, all the results are not presented. However, enough data is presented to have a good understanding of the *poor*, *fair*, and *good* threshold.

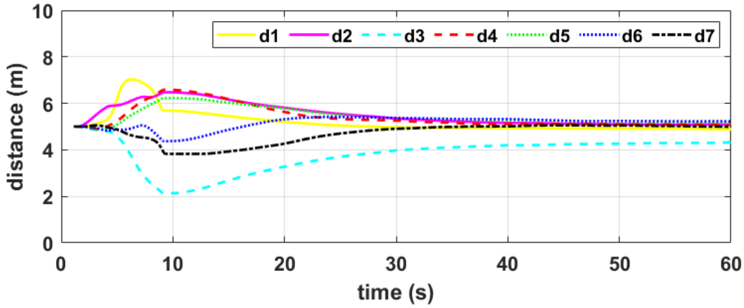
Figure 7.3 shows the results corresponding to the Figure 7.2a when runtime manager is employed with $nPacketLossFair = \{1,2\}$ and $nPacketLossPoor = \{3,4\}$. The time gaps for ACC and CACC controllers are 2 s and 1 s, respectively. The constant distance gap for the PLATOON controller is 5 m. Figures 7.3a and 7.3b demonstrate that the platooning vehicles can avoid collisions while using the runtime manager despite 5 m initial gap used in the PLATOON controller. The threshold for *fair* communication quality is 1 packet loss in these two cases. For the same scenario, the third vehicle runs into the second vehicle even with the runtime manager, when the threshold for *fair* communication quality is set to 2, Figures 7.3c, 7.3d. When the threshold for *fair* communication quality is 1, the runtime manager can respond to the packet loss by increasing the distance to the front vehicle. For instance, the vehicle 3 (d_2) increases the distance to the front vehicle due to one packet loss in Figures 7.3a and 7.3b. However, the vehicle 3 does not increase the distance to the front vehicle until 2 packet losses in the scenarios represented by the Figures 7.3c and 7.3d, which causes collisions.

The scenarios presented in Figure 7.4 exhibit completely different results in contrast to the Figure 7.3. In this case, the initial constant distance gap for PLATOON controller is set to 10 m. When the threshold for *fair* communication quality is 1, there are rear-end collisions in Figure 7.4a. The vehicles in Figure 7.4b barely avoid collisions. However, when the threshold for *fair* communication quality is set to 2, the platooning vehicles can successfully avoid collisions as shown in Figures 7.4c and 7.4d.

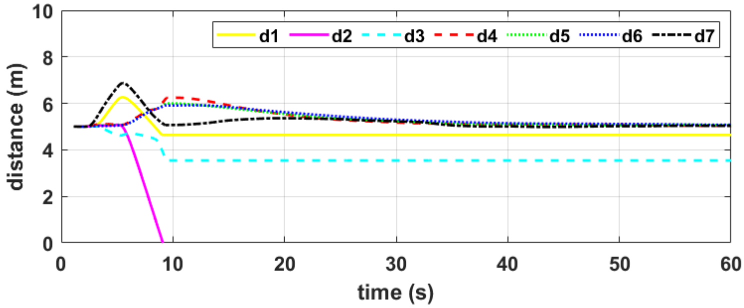
The results presented above suggest that when an initial inter-vehicle distance as short as 5 m is used, the runtime manager requires to react to packet losses fast by increasing the distance to the front vehicle. To this end, a small value for *fair* threshold should be chosen, Figures 7.3a and 7.3b. However, when the initial constant distance gap is larger, e.g., 10 m, performance degradation by increasing the inter-vehicle distance too early will further increase the inter-vehicle gap; this causes more packet losses due to path loss effects, and eventually causes rear-end collisions, Figure 7.4a. In Figure 7.4b the vehicles can avoid collisions because the threshold for *poor* communication quality is



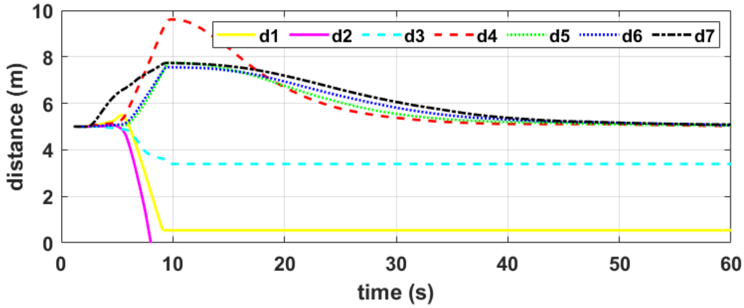
(a) *nPacketLossFair = 1, nPacketLossPoor = 3*



(b) *nPacketLossFair = 1, nPacketLossPoor = 4*

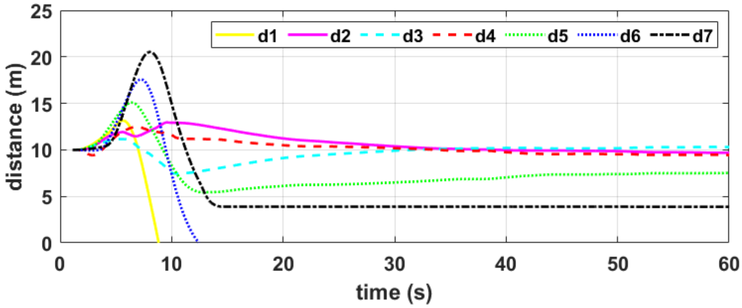


(c) *nPacketLossFair = 2, nPacketLossPoor = 3*

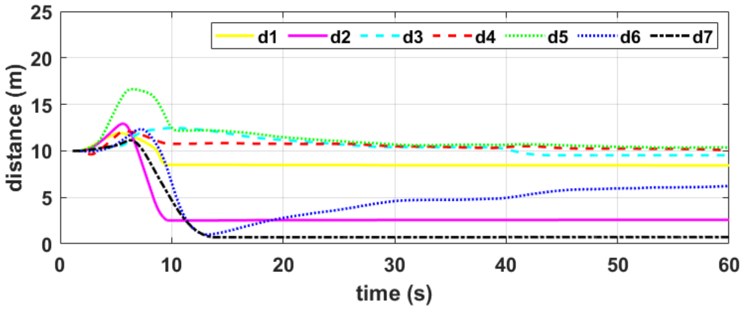


(d) *nPacketLossFair = 2, nPacketLossPoor = 4*

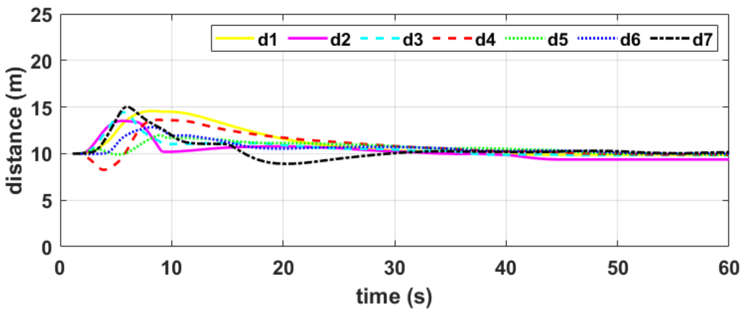
Figure 7.3: Inter-vehicle distance profiles of the platooning vehicles; ACC time gap = 2 s, CACC time gap = 1 s, PLATOON constant distance gap = 5 m.



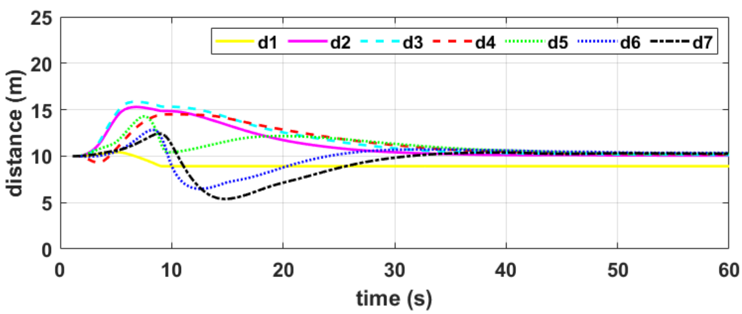
(a) *nPacketLossFair = 1, nPacketLossPoor = 3*



(b) *nPacketLossFair = 1, nPacketLossPoor = 4*

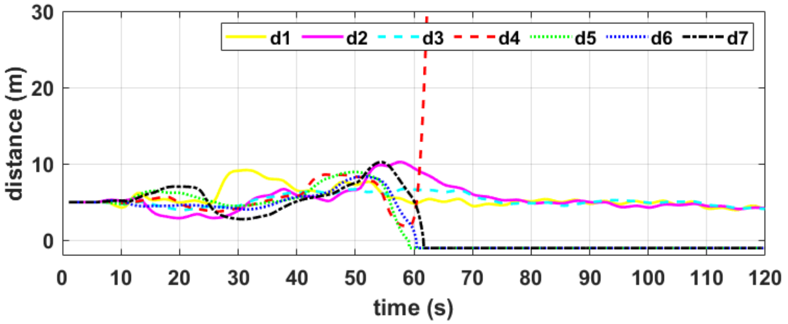


(c) *nPacketLossFair = 2, nPacketLossPoor = 3*

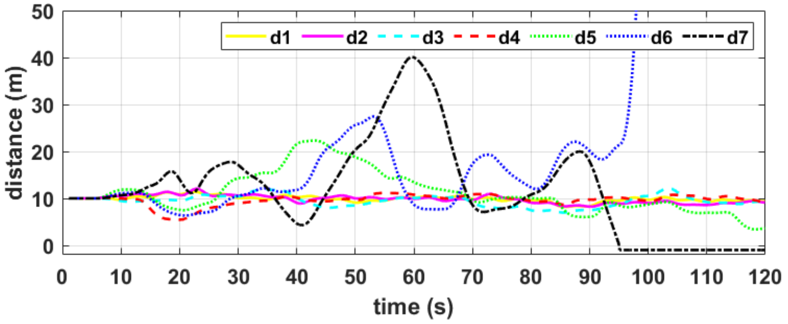


(d) *nPacketLossFair = 2, nPacketLossPoor = 4*

Figure 7.4: Inter-vehicle distance profiles of the platooning vehicles; ACC time gap = 2 s, CACC time gap = 1 s, PLATOON constant distance gap = 10 m.



(a) PLATOON constant distance gap = 5 m



(b) PLATOON constant distance gap = 10 m

Figure 7.5: Inter-vehicle distance profiles $d_i(t)$ of the platooning vehicles; sinusoidal scenario, no runtime manager, and different constant distance gaps for the PLATOON controller. Copyright © 2019, IEEE, reprinted from Hasan *et al.* ICCVE'2019.

set to 4 which prevents too frequent controller switching. To summarize, when the initial gap is short, the runtime manager must act fast to respond to the packet losses, but in case of a larger initial gap, the runtime manager should refrain from too frequent gap adjustment and controller switching.

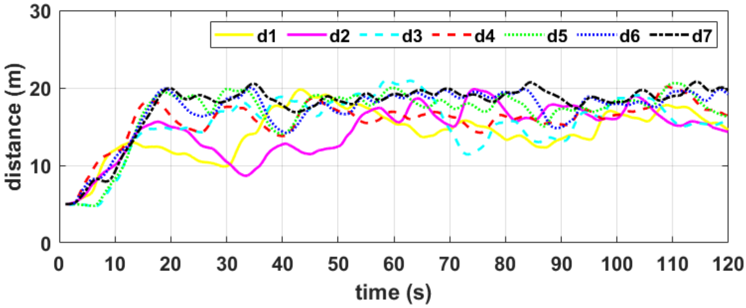
7.4 Runtime Manager Analysis

The results presented in this section correspond to the sinusoidal scenario in which the lead vehicle oscillates at a frequency of 0.2 Hz, and the following vehicles try to obey it. The simulation settings presented in Table 7.1 are also used here. This section aims to reveal how the runtime manager can help to maintain fail-operational platooning.

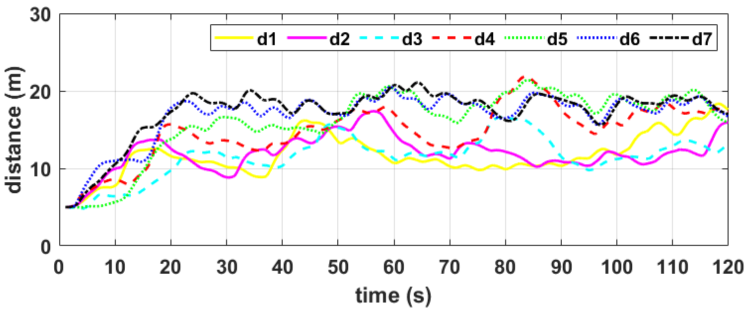
In this section also the platoon of length eight is simulated for 120 s that uses the PLATOON controller with initial constant distance gaps 5 and 10 m. The resulting distance profiles are depicted in Figure 7.5. For CDG of 5 m, at the 58^{th} second of simulation time, vehicle 5 runs into the vehicle 4 from the behind, and brings it into a complete standstill, Figure 7.5a. As a result, vehicles 6 and 7 also undergo rear-end collisions. Vehicles 0–3, on the other hand, keep cruising, and this increases the distance between vehicles 3 and 4 sharply (the red curve, d_4). As the non-colliding vehicles (0–3) move away from the rest, this platoon of three vehicles becomes more string stable, and can maintain the default 5 meters gap. This is because all the vehicles are close to the leader and receive the beacons successfully. Similar behaviour can be observed for initial CDG of 10 meters with PLATOON controller, as shown in Figure 7.5b. As the vehicles 6 and 7 are far away from the leader due to longer inter-vehicle distances, they experience more packet losses due to path loss effect. At around 95^{th} second of the simulation time, vehicle 7 runs into vehicle 6 from the behind. The rest of the platooning vehicles keep moving and maintain the default 10 m gap, and exhibit string stable behaviour. The runtime manager can detect the transient communication errors that cause these collisions and temporarily degrade the performance to avoid them.

The same scenario as in Fig. 7.5a has been simulated with runtime manager which always starts with the PLATOON controller; the initial CDG for PLATOON controller is 5 m. The distance profiles $d_i(t)$ of the platooning vehicles are depicted in Figures 7.6a – 7.6d with *fair* and *poor* thresholds {1,2} and {3,4}, respectively. It is apparent from the figures that the runtime manager avoids the collision at the 58^{th} second for all the combinations of *fair* and *poor* values that occurred in Fig. 7.5a. In order to explain the proactive measures taken by the runtime manager to avoid the collision that is caused by vehicle 5, the state profiles of the vehicle 5 are presented in Figures 7.6e – 7.6h. In Figure 7.6g, just before the 58^{th} second, the vehicle is using the PLATOON controller, and due to 2 packet losses it increases the gap to the front vehicle, and transitions to the PLATOON & GA state. In the meantime, the runtime manager notices 3 packet losses with respect to the lead vehicle which is considered as *poor* connection, and the vehicle switches to the CACC controller at the 58^{th} second. In Figure 7.6h, as the *poor* threshold is set to 2, due to 2 packet losses vehicle 5 just increases the gap, but still can avoid collision. When the vehicle uses the CACC controller, and the connection to the lead vehicle gets better, it switches back to the PLATOON controller again. This way, vehicle 5 hovers between the PLATOON and CACC controllers to both minimize the gap and avoid collisions. If we look at the difference between the Figures 7.6e – 7.6f and 7.6g – 7.6h, it can be observed that there are too many state switching when the threshold for *fair* communication quality is set to 1. In contrast, the vehicle exhibits less state switching when *fair* threshold is set to 2. When the *poor* threshold is set to 3, the vehicle sometimes need to adopt the ACC controller. However, when the *poor* threshold is set to 4, there are less switching between the controllers.

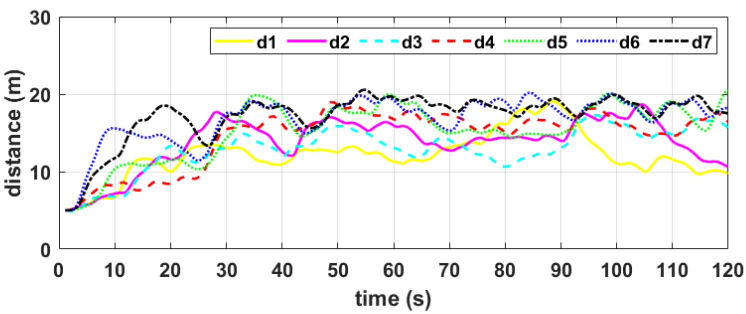
The runtime manager results corresponding to Figure 7.5b with initial CDG of 10 m for the PLATOON controller are depicted in Figure 7.7. It is apparent from the figure that the vehicle 7 which caused collision at the 95^{th} second in Figure 7.5b, avoids collision in this case. During that time, the vehicle switches



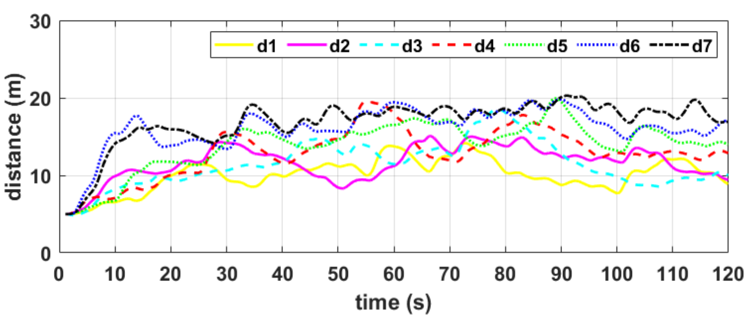
(a) *nPacketLossFair = 1, nPacketLossPoor = 3*



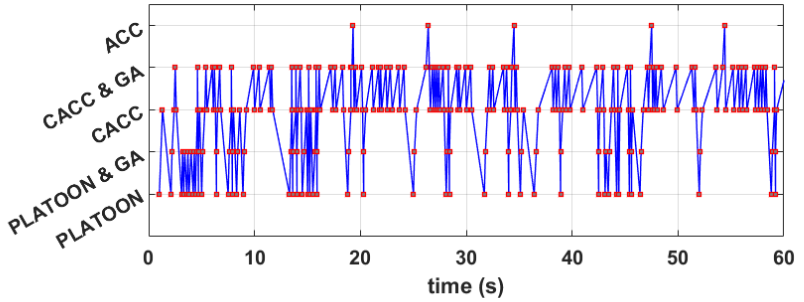
(b) *nPacketLossFair = 1, nPacketLossPoor = 4*



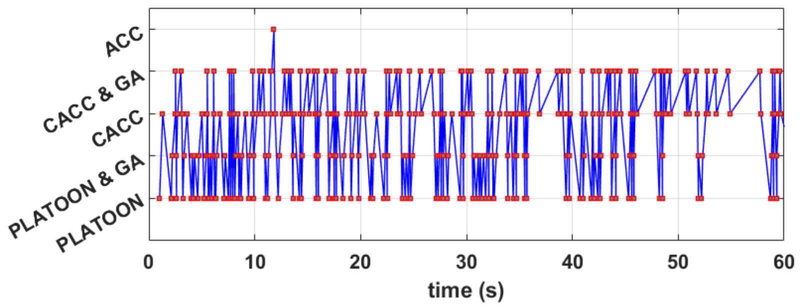
(c) *nPacketLossFair = 2, nPacketLossPoor = 3*



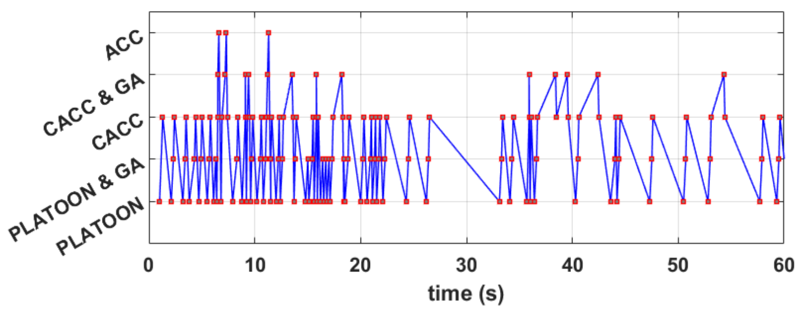
(d) *nPacketLossFair = 2, nPacketLossPoor = 4*



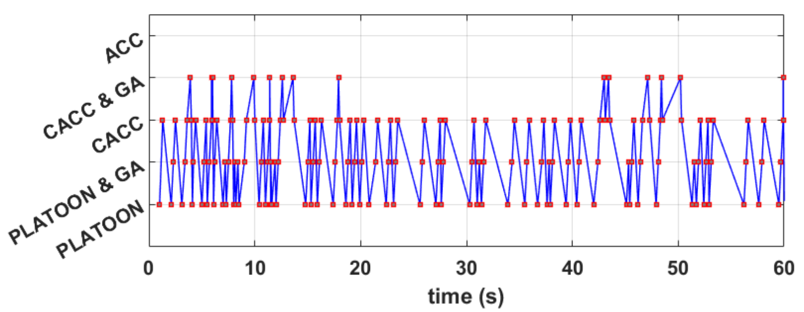
(e) *nPacketLossFair = 1, nPacketLossPoor = 3*



(f) *nPacketLossFair = 1, nPacketLossPoor = 4*

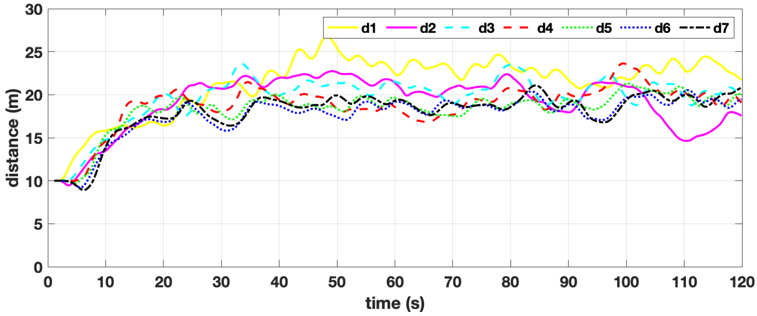


(g) *nPacketLossFair = 2, nPacketLossPoor = 3*

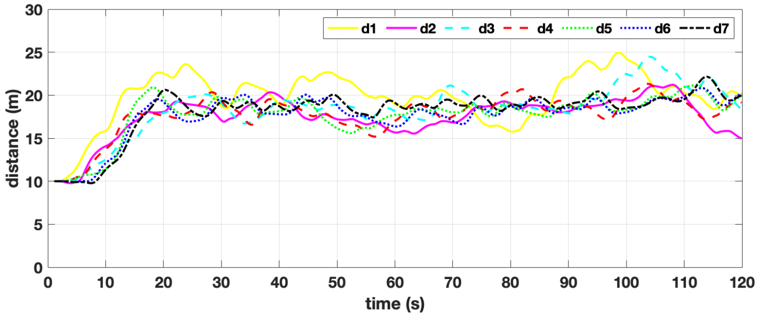


(h) *nPacketLossFair = 2, nPacketLossPoor = 4*

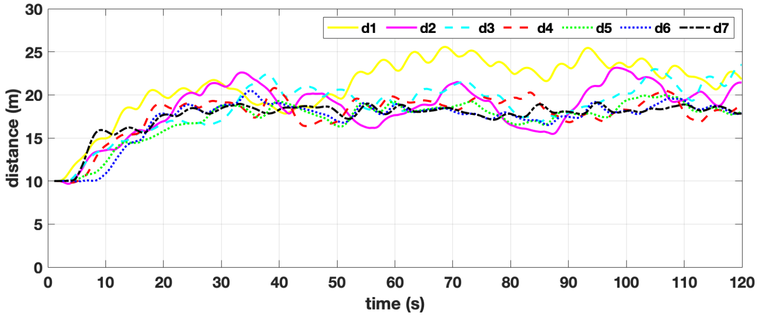
Figure 7.6: Figures 7.6a–7.6d present the inter-vehicle distance profiles, and Figures 7.6e–7.6h present the states of vehicle 5 while using the RM; ACC CTG = 1 s, CACC CTG = 0.6 s, PLATOON CDG = 5 m.



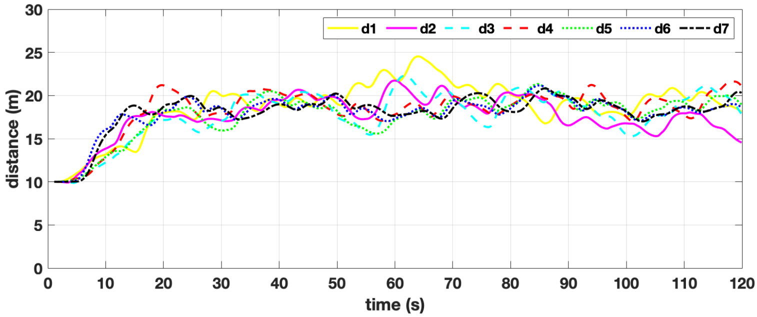
(a) *nPacketLossFair = 1, nPacketLossPoor = 3*



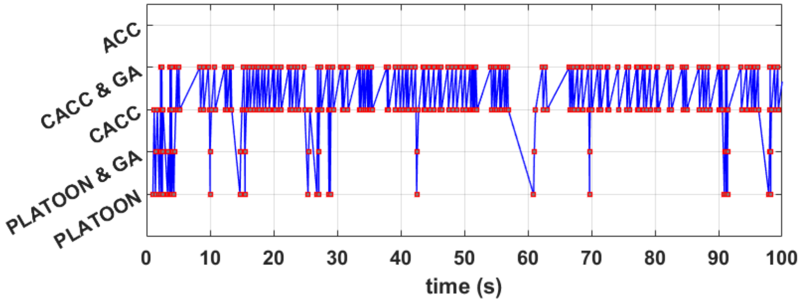
(b) *nPacketLossFair = 1, nPacketLossPoor = 4*



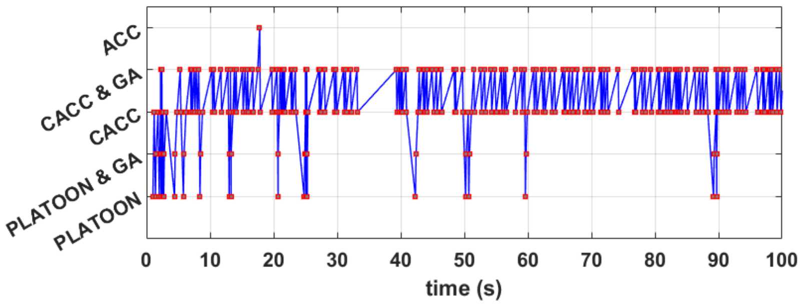
(c) *nPacketLossFair = 2, nPacketLossPoor = 3*



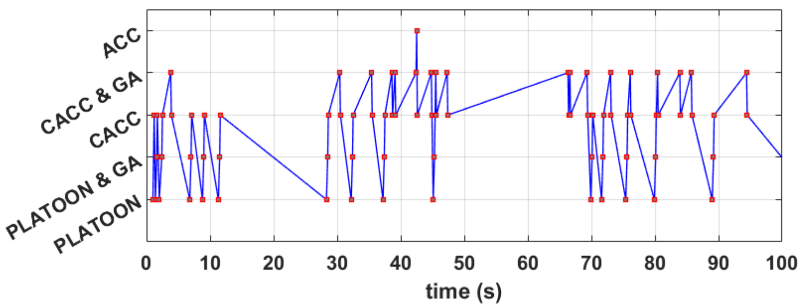
(d) *nPacketLossFair = 2, nPacketLossPoor = 4*



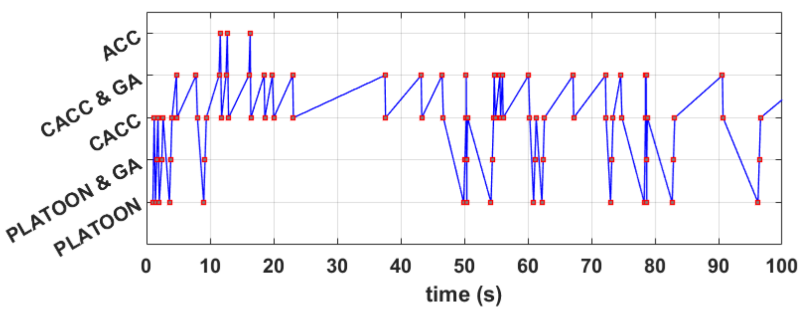
(e) *nPacketLossFair* = *1*, *nPacketLossPoor* = *3*



(f) *nPacketLossFair* = *1*, *nPacketLossPoor* = *4*



(g) *nPacketLossFair* = *2*, *nPacketLossPoor* = *3*



(h) *nPacketLossFair* = *2*, *nPacketLossPoor* = *4*

Figure 7.7: Figures 7.7e – 7.7h present the inter-vehicle distance profiles, and Figures 7.6e–7.6h present the states of vehicle 7 while using the RM; ACC CTG = 1 s, CACC CTG = 0.6 s, PLATOON CDG = 10 m. Copyright © 2019, IEEE, reprinted from Hasan *et al.* ICCVE'2019.

to CACC controller due to *poor* connection to the leader, as shown in Figure 7.7h. For *poor* threshold 4, vehicle 7 switches to the PLATOON controller due to *good* connection with both the front and the lead vehicles. Although the sequence of state changing is different for different *fair* and *poor* values, in both cases vehicle 7 can avoid the collision by switching to CACC controller in time. Figure 7.7 also demonstrates that there are too frequent state switching when the threshold for *fair* communication quality is 1, comparatively less state switching can be observed when the *fair* threshold is set to 2, Figures 7.7g, 7.7h.

Figure 7.7 uses initial CDG of 10 m for the PLATOON controller whereas, it is 5 m in the case of Fig. 7.6. A careful look at these figures shows that the scenarios represented by Fig. 7.7 has fewer state switching than the scenarios in Fig. 7.6. This is because of the initial constant distance gap. Whenever vehicle 7 in Figures 7.7e – 7.7h switches to PLATOON controller, the connection to the leader becomes *poor* as it is far away from the leader, and it switches to CACC controller in the immediate next monitor interval. This is why, most of the time, the vehicles in Fig. 7.7 use CACC controller as it does not require connection to the leader, and exhibits a consistent inter-vehicle spacing in comparison to the vehicles in Fig. 7.6. Due to shorter CDG (5 meters), the vehicles in the front of the platoon in Fig. 7.6 are closer to the leader while using the PLATOON controller. If the connection to the leader worsens, they switch to CACC or PLATOON & GA state. Thus, these vehicles alternate between PLATOON and CACC controllers quite frequently and maintain a shorter inter-vehicle distance on an average throughout the simulation time. This is good from a fuel efficiency point of view. However, from a safety and string stability point of view, less state switching and thus consistent inter-vehicle distance like in Fig. 7.7 is better. Moreover, Figures 7.6e – 7.6h and, 7.7e – 7.7h exhibit an aggressive increase in inter-vehicle distance from the very beginning of the simulation. This is because the non-platooning vehicles generate interference throughout the simulation time that causes packet losses. The runtime manager successfully detects these and takes necessary safety measures the whole time.

Rigorous simulations have been performed with different combinations of *poor* and *fair* thresholds (1–6), different CDGs for PLATOON controller (5, 10, and 15 m), different CTGs for ACC controller (1 s, 2 s) and CACC controller (0.6 s, 1 s). For all these combinations, the runtime manager can avoid any sort of collisions between the vehicles despite the harsh wireless environment that causes packet losses all the times. For the sake of brevity, only a few results are presented in this chapter.

7.5 Discussion

The number of packet losses that determine the *good*, *fair*, and *poor* communication quality is difficult to predict during the design time, and this should also be determined during the runtime. The thresholds that are used in this chapter are suitable for this particular simulation scenario only. For a different channel condition, different thresholds could be more suitable. In this chapter, it is shown that the runtime manager can very efficiently keep a platoon fail-operational even in a very harsh wireless environment, apparently for all choices of the *fair* and *poor* threshold. This proves the robustness of the runtime manager in maintaining fail-operational platooning.

Looking at the distance profiles presented in this chapter one might wonder why the vehicles are using such a large inter-vehicle distance while using the runtime manager, and obviously, it is not fuel-efficient. The reason is that in the chosen scenario the platooning vehicles experience packet losses all the time, mainly due to interference from the non-platooning vehicles. However, in a realistic scenario, this does not always happen, and in that case, the platooning vehicles would be able to drive in close formation. The runtime manager ensures sub-optimal fuel-efficiency in the presence of transient communication errors. Moreover, when it comes to safety, increasing the inter-vehicle distances as a proactive measure due to transient errors is more crucial than fuel-efficiency and string stability. This is why the runtime manager acts independently in all the platooning vehicles in a decentralized fashion.

Chapter 8

Evaluation of Fail-Safe Platooning

The performance of the SB and AEB strategies are evaluated by comparing them with the state-of-the-art braking approaches, e.g., Gradual Deceleration, Coordinated Emergency Brake Protocol, and also the normal braking. The aim is to understand which braking approaches are more suitable for transitioning the platoon to the fail-safe state. Rigorous simulations have been performed to evaluate the performance of the braking strategies under different network loads. The results presented in Section 8.2 are based on the following publication[1]:

- S. Hasan, A. Balador, S. Girs and E. Uhlemann, "Towards Emergency Braking as a Fail-Safe State in Platooning: A Simulative Approach," *IEEE 90th Vehicular Technology Conference (VTC2019-Fall),* Honolulu, HI, USA, 2019, pp. 1-5. [90].

8.1 Performance and Communication Metrics

In this section, the performance and communication metrics that are used to analyze the braking strategies are defined.

8.1.1 Performance Metrics

Section 7.1 in Chapter 7 defines the necessary conditions for fail-safe state in platooning which are, $d_i(t) > 0$ and $d_L \leq d_{hazard}$. The first condition implies that the gap between the vehicles at complete standstill after braking is greater than zero, and the second condition dictates that the stopping distance of the lead vehicle is less than or equal to the distance to the hazard. The CEB braking strategies are evaluated in terms of these fail safe conditions. The performance metrics that are considered at the application level are as follows:

1. *Waiting time (τ_{wait})*: This is the waiting time that all the platooning vehicles should pursue before they start braking in accordance with the synchronized braking strategy.

2. *Total time to stop (T_{total}):* This is the total time required by the whole platoon to come into a complete standstill. This metric defines how long it requires for the platoon to transition to the fail-safe state since the generation of DENM by the lead vehicle.

3. *Stopping distance of LV (d_L)*: The distance traversed by the lead vehicle from the time it generates a DENM until it comes into a complete standstill.

4. *CDG with PLATOON controller*: This metric is used to determine a suitable constant distance gap to be used with the PLATOON controller.

5. *Minimum inter-vehicle distance ($d_i(t)_{min}$)*: The minimum distance between any pair of vehicles after the platoon has come into a complete standstill. This metric is used to determine if there are collisions in the platoon, and $d_i(t)_{min}$ is zero if there is a collision.

8.1.2 Communication Metrics

The following metrics are used to evaluate the state of the communication network.

1. *Channel Busy Ratio (CBR)*: CBR estimates the channel load, and it is defined as the elapsed time during which the physical layer perceives the channel status busy over a given time window [49]. CBR is calculated every 1 s throughout the whole simulation time, and it is a different value for each node.

2. *DENM delay*: DENM delay is calculated for each FV, and this is the elapsed time between the LV broadcasting its first DENM and an FV successfully receiving a DENM for the first time. An FV may receive multiple copies of the same DENM, which are discarded after the first copy has been received.

3. *ACK delay*: ACK delay is calculated as the time from when the LV disseminates the first DENM until a vehicle receives the first ACK from its immediately following vehicle. This is calculated for each vehicle except for the last FV.

Table 8.1: Configuration parameters for simulation analysis.

	Parameter	Value
communication	Path loss model	Free space ($\alpha = 2$)
	TxPower	100 mW
	Packet size	200 B
	Bit rate	6 $Mbps$
	Sensitivity	-94 dBm
	Thermal noise	-95 dBm
	Frequency	5.89 GHz
	Bit rate (non-platooning vehicles)	3 $Mbps$
mobility	Leader speed	27.778 m/s
	Platoon size	8
	No. of non-platooning vehicles	100
	Leader oscillation frequency	0.2 Hz
	Total no. of lanes	4
braking	brakeAtTime	20 s
	repetitionInterval	105 ms
	waitingTime	100 ms
	decelerationRate	-12 ms^{-2}

8.2 Evaluation of Synchronized Braking Strategy

In this section, the performance of the synchronized braking strategy is analyzed by comparing it with the normal braking. The performance metrics 1–4 defined above are used to perform the analysis.

8.2.1 Simulation Scenario and Settings

A platoon of length eight is simulated in the presence of 100 non-platooning vehicles which are located within the vicinity of the considered platoon. The non-platooning vehicles transmit CAMs periodically for generating interference and increase contention. The PHY and MAC layer parameters have been kept the same as in the Plexe simulator, which follows the IEEE 802.11p standard, and some new parameters are introduced as outlined in Table 8.1.

The platoon cruises at a speed of 100 kmph in accordance with the PLATOON, CACC, or ACC control law. At the 20^{th} s of the simulation time, the platoon encounters an imaginary road hazard and performs emergency braking. The DENMs are transmitted at an interval of 10 ms. The non-platooning vehicles transmit CAM at an interval of 50 ms intending to increase channel contention and induce packet losses. The platooning vehicles perform braking at a very high rate (–12 ms^{-2}) while using the synchronized braking strategy.

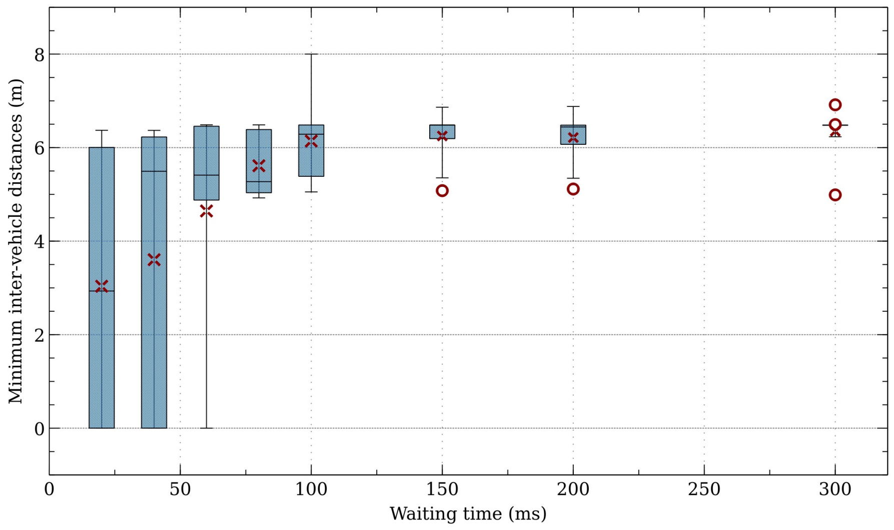
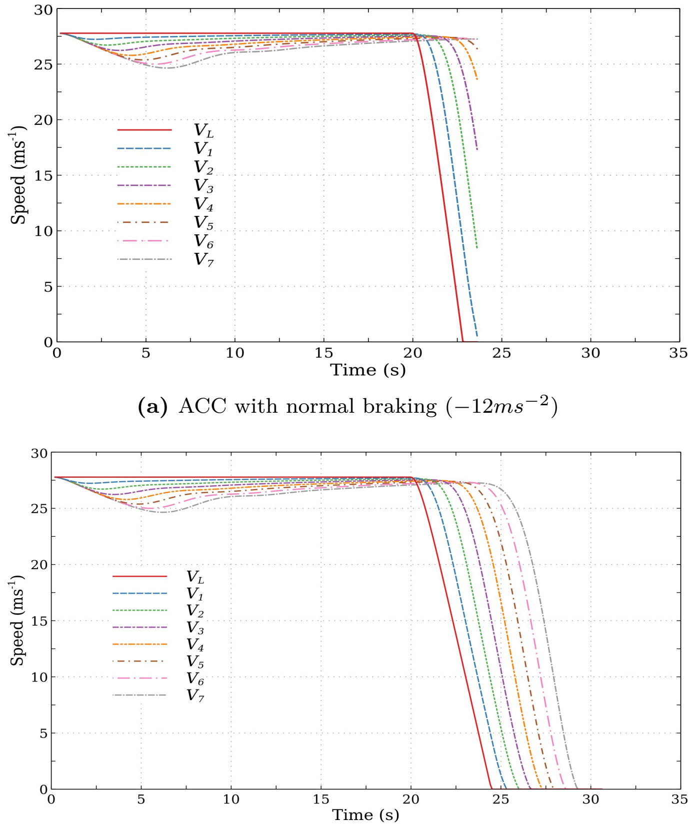


Figure 8.1: Minimum inter-vehicle distances for different waiting times for a platoon of length 8 cruising at a speed of 100 kmh^{-1} with an initial inter-vehicle distance of 8 meters. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

8.2.2 Results and Analysis

Waiting time (τ_{wait}) The analysis begins with finding a suitable waiting time that a platoon should pursue to minimize the stopping time, but still being able to avoid rear-end collisions. To this end, simulations are carried out for a number of waiting times 20, 40, 60, 80, 100, 150, 200 and 300 ms, ten runs for each. For each run, the minimum inter-vehicle distance between any two vehicles in the platoon after it has come into complete standstill has been recorded and represented by the box plot in Figure 8.1, in which x marks the mean and o marks outliers. In case there is a rear-end collision, the inter-vehicle distance is considered to be zero. The size of the Inter-Quartile range (IQR) represents how spread the data points are, which also reflects the platoon stability. In this case, the platoon started cruising at 100 kmh^{-1} with an initial inter-vehicle distance of 8 meters. For waiting times 20, 40 and 60 ms, there are collisions in the platoon for some runs. For a waiting time of 80 ms, all ten runs can avoid rear-end collision. For 100 ms or higher, we can observe an even better average and shorter IQR. In the following analysis, all vehicles travel at 100 kmh^{-1} and wait 100 ms, and this corresponds to travelling 2.78 meters before starting synchronized braking.

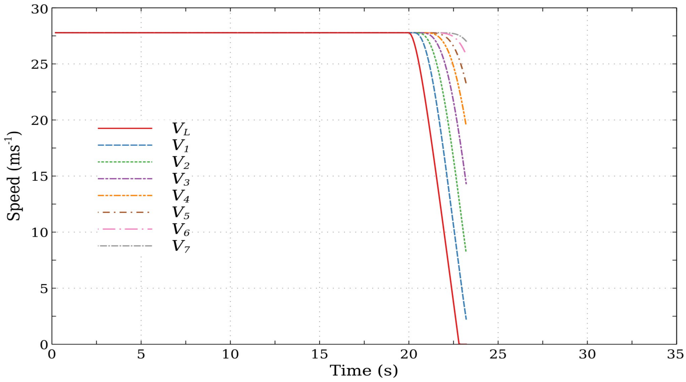
Speed profiles For the sake of analyzing the performance of synchronized braking strategy together with ACC, CACC and PLATOON controller, let us first look at their speed profiles as illustrated in Figures 8.2, 8.3, and 8.4.



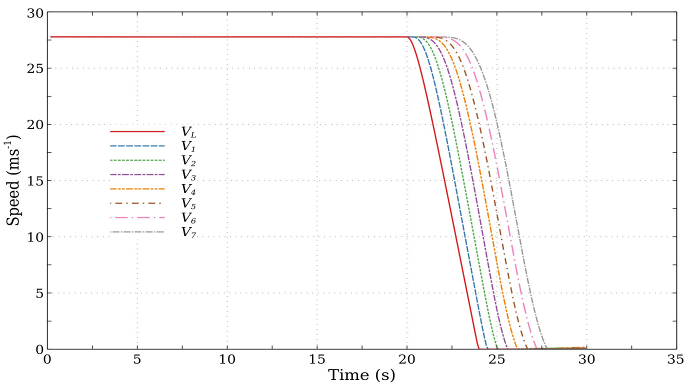
(a) ACC with normal braking ($-12ms^{-2}$)

(b) ACC with normal braking ($-7ms^{-2}$)

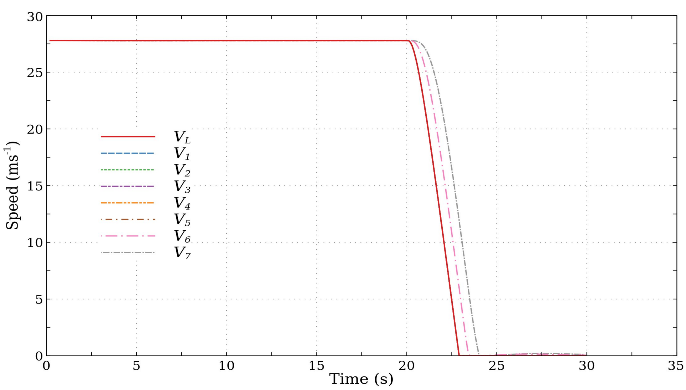
Figure 8.2: Speed profiles of platooning vehicles for ACC controller with an initial constant time gap of 1.2 s meters and speed 100 kmh^{-1}. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.



(a) Normal braking with CACC controller ($-12ms^{-2}$)

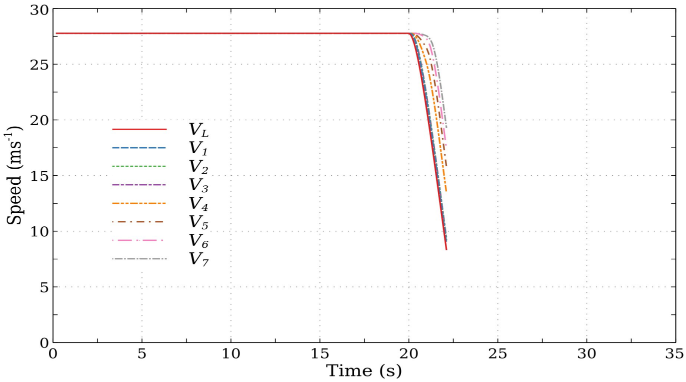


(b) Normal braking with CACC controller ($-8ms^{-2}$)

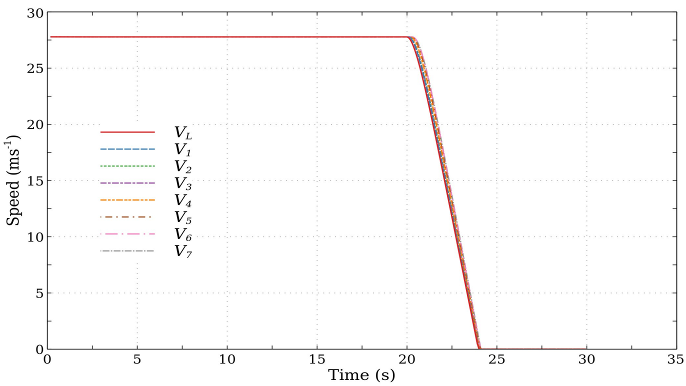


(c) Synchronized braking with CACC controller ($-12ms^{-2}$)

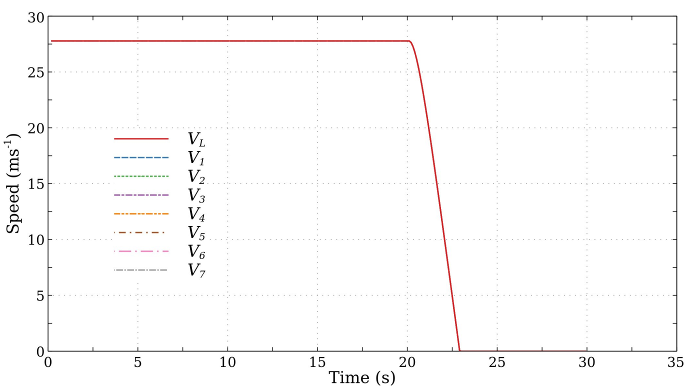
Figure 8.3: Speed profiles of platooning vehicles for CACC controller [53] with constant time gap of 0.6 s and speed 100 kmh^{-1}. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.



(a) Normal braking with PLATOON controller ($-12ms^{-2}$)



(b) Normal braking with PLATOON controller ($-8ms^{-2}$)



(c) Synchronized braking with PLATOON controller ($-12ms^{-2}$)

Figure 8.4: Speed profiles of platooning vehicles with PLATOON controller [20] with an initial CDG of 5 meters and speed 100 kmh^{-1}. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

Note that the purpose of Figures 8.2 – 8.6 is not to analyze which controller performs better, as a qualitative analysis of this can be found in Section 4.1. Also, the analysis provided in this section is not a comparison between the use of controllers and the use of synchronized braking. Rather, it is demonstrated that the synchronized braking can be implemented together with these controllers and compare it to normal braking.

When using the simulation settings from Table 8.1 and CTG of 1.2 s, 0.6 s, and CDG of 5 m for ACC, CACC, and PLATOON controllers, respectively as suggested by the authors themselves [53, 19], all three controllers undergo rear-end collisions with normal braking at a deceleration rate of –12 ms^{-2} as illustrated in Figures 8.2a, 8.3a, and 8.4a, respectively. However, at a lower deceleration rate (–7ms^{-2} for ACC and –8ms^{-2} for CACC and PLATOON at their respective CTGs and CDGs) the vehicles can avoid rear-end collisions using normal braking, Figures 8.2b, 8.3b, 8.4b. The main reasons behind the collisions with normal braking using the CACC and PLATOON controllers are communication problems due to high interference from non-platooning vehicles which increases the time required to deliver DENMs to all vehicles, and high deceleration rate. Segata *et al.* also reported that the tolerable communication delay decreases with the increase of deceleration rate in [94]. In case of synchronized braking as shown for CACC and PLATOON controllers in Figures 8.3c and 8.4c, all vehicles wait 100 ms but then complete their braking manoeuvre successfully even with a deceleration rate of –12 ms^{-2}. Recall that during the 100 ms delay, the leading vehicle will have travelled 2.78 meters, which is not visible in the time scale used in the figures. Due to the long inter-vehicle distances with CACC controller and progressive communication delay, the last two vehicles did not receive any DENM within the 100 ms waiting period and thus, completed the braking manoeuvre based only on the CAMs and radar sensor in accordance with the CACC control law.

Stopping distance of LV (d_L) and the distance traversed during τ_{wait}
Although the platoon can avoid rear-end collisions for a braking scenario with lower deceleration rate, in case of using ACC, CACC or PLATOON controllers, it is evident that the leader will traverse longer and thus, endanger the purpose of emergency braking. Results from our quantitative analysis of how much the leader traverses in the normal braking and synchronized braking scenarios for different deceleration rates are presented in Table 8.2. The ACC field in synchronized braking has been left blank since there is no wireless communication in ACC, and thus, it cannot use synchronized braking. With synchronized braking, the gain of avoiding rear-end collision comes at the cost that the leader traverses 2.77 meters longer before braking due to waiting for 100 ms. However, the leader has to traverse 16.5 meters more by applying a lower deceleration rate in order to avoid rear-end collisions for normal braking using the CACC and PLATOON controllers. To further clarify the trade-offs between synchronized braking and the distance traversed by the leader, the distances traversed by the leader have been recorded for different waiting times in the

Table 8.2: Stopping distance of LV d_L. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

Controller	Normal braking scenario				Synchronized braking scenario	
	Deceleration rate (ms^{-2})	Distance traversed d_L (m)	Deceleration required to avoid collision (ms^{-2})	Distance traversed d_L (m)	Deceleration rate (ms^{-2})	Distance traversed d_L (m)
ACC	-12	44.243	-7	67.827	-	-
CACC	-12	44.243	-8	60.817	-12	47.02m
PLOEG	-12	44.243	-8	60.817	-12	47.02m

Table 8.3: Distance traversed by LV for different waiting times τ_{wait}. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

Synchronized braking	τ_{wait} (ms)	20	40	60	80	100	150	200	300	500
	Distance traversed d_L (m)	44.70	45.35	45.90	46.46	47.02	48.40	49.79	52.57	58.13
Normal braking	τ_{wait} (ms)	0	0	0	0	0	0	0	0	0
	Distance traversed d_L (m)	44.24	44.24	44.24	44.24	44.24	44.24	44.24	44.24	44.24

synchronized braking scenario and they are shown in Table 8.3. For the normal braking scenario with no waiting time, the leader traverses 44.243 metres, and the platoon experiences rear-end collisions. For a waiting time of even 500 ms in synchronized braking, the leader traverses a shorter distance in total (58.13 m) compared to the normal braking scenario (60.817 m), which needs to decelerate slower to avoid rear-end collisions.

CDG with PLATOON controller An effort has been made to suggest a suitable initial inter-vehicle distance for synchronized braking such that all the platooning vehicles can receive DENMs within the τ_{wait} period (100 ms). The boxplots in Figure 8.5 show the inter-vehicle distances in a platoon at a complete standstill for different initial gaps using the PLATOON controller. It can be concluded from Figure 8.5 that the initial gaps between 3–12 metres can retain almost the same gap even after braking due to the successful reception of DENMs. The problem with very small inter-vehicle gaps such as 3 or 5 metres is that if the vehicles do not receive DENMs in time and are required to decelerate at a high rate, there is a high chance of rear-end collision in the normal braking scenario. In case of using synchronized braking, however, additional time is allowed to wait for the DENM to be received and performance increases.

Table 8.4: Total time to stop T_{total} for different waiting times τ_{wait}. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

Synchronized Braking	τ_{wait} (ms)	20	40	60	80	100	150	200	300	500
	T_{total} (s)	4.53	5.06	4.88	4.54	3.72	3.45	3.53	3.73	3.85
	No. of collisions	3	2	2	0	0	0	0	0	0
Normal braking	τ_{wait} (ms)	0	0	0	0	0	0	0	0	0
	T_{total} (s)	5.43	5.43	5.43	5.43	5.43	5.43	5.43	5.43	5.43
	No. of collisions	3	3	3	3	3	3	3	3	3

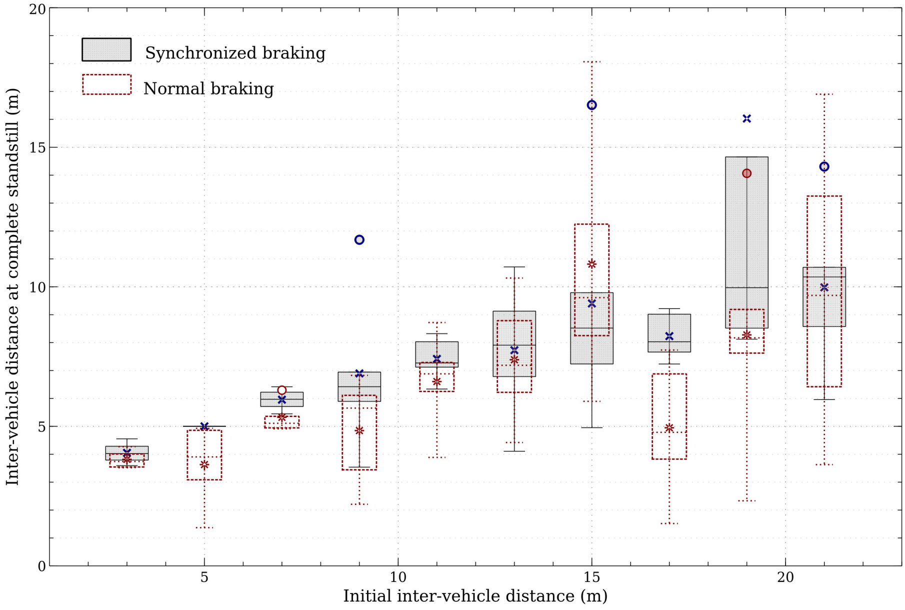


Figure 8.5: Inter-vehicle distances at complete standstill for different initial CDGs with a deceleration rate of – 12 ms^{-2}.

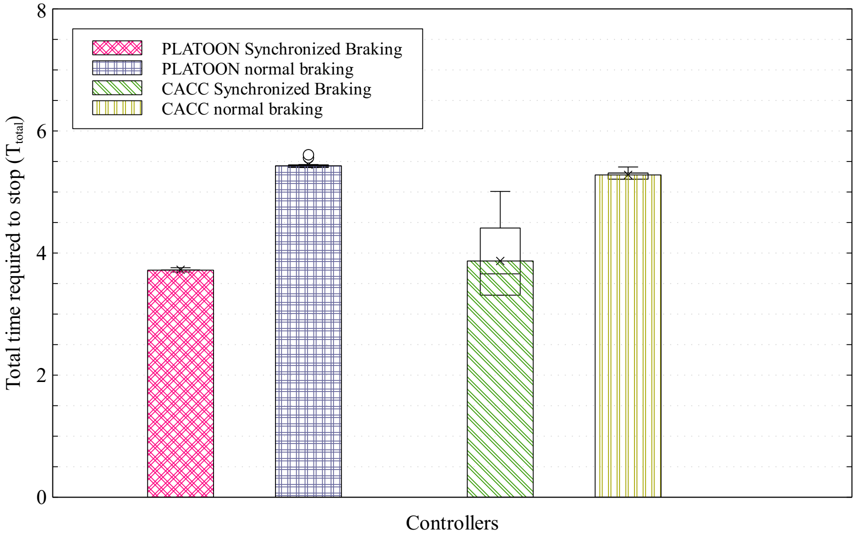


Figure 8.6: Total time required for the platoon to come into complete standstill, T_{total} with different controllers. Copyright © 2019, IEEE, reprinted from Hasan *et al.* VTC2019-Fall.

Total time to stop and T_{total} for different waiting times In this part, the total time required for the whole platoon to come into complete standstill from the moment of detecting the road hazard is analyzed, and it is shown that waiting for a certain period when using the synchronized braking strategy does not necessarily prolong the total time to stop. To this end, ten simulation runs have been carried out with the CACC and PLATOON controllers, and the average total time required to stop the platoon is recorded as illustrated in Figure 8.6 with the help of bars and boxplots. The speed of the platooning vehicles is 100 kmh^{-1}, and the CDG and CTG are 5 m and 0.6 s for PLATOON and CACC controllers respectively, while the other parameters are set according to Table 8.1. Synchronized braking outperforms normal braking in this respect as well. Moreover, the average of the total time to stop for different waiting times is presented in Table 8.4. The table also shows the number of runs for which the platoon has experienced rear-end collisions. For τ_{wait} of 20, 40 and 60 ms, there has been collisions for some runs and the total stopping time is quite long. This is due to inadequate waiting time, i.e., one or two vehicles in the tail of the platoon fail to receive DENMs, and either end up colliding or stop with the aid of regular periodic beacons and thus, take longer time. The primary purpose of Table 8.4 is to disclose that inadequate waiting time does not necessarily avoid rear-end collisions. So, if the hazard imposed by a road hazard is so imminent that the leader cannot afford to traverse a few more meters even though synchronized braking can eradicate rear-end collisions and ensure fail-safety, then it might be appropriate to brake immediately.

8.3 Evaluation of Adaptive Emergency Braking

Section 8.2 focuses on how the synchronized braking strategy can be used together with the state-of-the-art controllers. Moreover, various performance metrics are benchmarked. In this section, we conduct a quantitative comparison analysis among all four CEB strategies presented in Chapter 5, e.g., GD-1, SB-2, CEBP-3, and AEB-4, to understand to what extent these braking strategies can fulfil the conditions of the fail-safe state. A different simulation setting is used for this purpose. The communication metrics, e.g., CBR, DENM delay, and ACK delay are first evaluated under various network loads. Then the results associated with the braking strategies are presented.

8.3.1 Simulation settings and traffic model

A platoon of seven vehicles is considered that uses the PLATOON controller. Recall that with the PLATOON controller, a platooning vehicle relies on CAMs from its preceding vehicle and the LV, i.e., leader-predecessor following strategy. It is assumed that at 20 s of simulation time, the LV detects an imaginary road hazard and starts broadcasting DENMs. The ACK packets are sent/relayed by every vehicle starting from the last vehicle in case of CEBP-3 and AEB-4 strategy. The channel models used for accounting the path loss and fading ef-

Table 8.5: Network and mobility parameters for simulation analysis.

	Parameter	Value
communication	Path loss model	Free space ($\alpha = 2$)
	Fading model	Nakagami-m (m = 3)
	PHY/MAC model	802.11p/1609.4 single channel (CCH)
	Frequency	5.89 GHz
	TxPower	100 mW
	Sensitivity	-94 dBm
	Thermal noise	-95 dBm
	Packet size	200 B
	Bit rate	6 $Mbps$
	Bit rate (non-platooning vehicles)	3 $Mbps$
mobility	Platoon speed	100 km/h
	Platoon size	7
	No.of platoons	1
	gap_{des}	5 m
	Controller	PLATOON
	Simulation duration	30 s
braking	BrakeAtTime	20 s
	brakeLag	200 ms
	softDecelerationRate	-2 ms^{-2}
	fullDecelerationRate	-8 ms^{-2}

fects are the free space path loss ($\alpha = 2$) model and the Nakagami-m ($m = 3$) fading model. The IEEE 802.11p and IEEE 1609.4 models that Plexe inherits from Veins are used to simulate the PHY and MAC layers. Table 8.5 summarizes all communication and mobility parameters. Please refer to the Plexe documentation[2] for details of vehicle dynamics and controller parameters.

To model the neighbouring traffic of the platoon, the number of non-platooning vehicles are varied to generate different levels of channel loads. All the simulations have been performed using all six different configurations presented in Table 8.6. Config 1 is intended to generate a high level of interference by overloading the channel, whereas config 6 is meant to provide a tolerable level of load. To this end, in addition to the neighbouring traffic, the number of lanes are also varied to introduce different vehicle densities. Configs 2-5 are introduced to understand the effects of beacon frequency, vehicle density, no. of neighbouring vehicles, etc. on DENM and ACK delays. A hundred simulation runs have been carried out to determine the communication metrics such as CBR, DENM delay, and ACK delay.

8.3.2 Performance evaluation of communication metrics

CBR The CBR of the LV for all six configurations is presented in Table 8.7. The CBRs of the other platooning vehicles are similar to the LV, so for brevity, all the results are not presented. As seen, config 1 and config 2 saturate the

[2] http://plexe.car2x.org/

Table 8.6: Different configurations for varying the channel load.

Configuration number	No. of non-platooning vehicles	No. of lanes	Vehicle density (vehicles/km)	Beacon frequency (Hz)			
				DENM	CAM	ACK	non-platooning vehicles
config 1	600	4	95	20	20	20	40
config 2	300	3	65	15	15	15	30
config 3	300	3	65	15	15	15	20
config 4	150	3	65	15	15	15	10
config 5	150	3	65	10	10	10	10
config 6	50	2	36	10	10	10	10

Table 8.7: Channel Busy Ratio (CBR).

Config no.	config 1	config 2	config 3	config 4	config 5	config 6
CBR	0.914	0.894	0.818	0.671	0.652	0.358

channel. At first glance, config 1 might seem unreasonable, but this scenario is very much possible in highways leading to big cities during the rush hours. Beacon frequency is another important factor. For instance, configs 2 and 3 only differ in the beacon frequency of the non-platooning vehicles, and this gives us two quite different CBRs.

DENM delay Figure 8.7 presents the DENM delay for all six configurations. In general, the rear vehicles in the platoon experience higher DENM delay in comparison to others due to the increasing effects of the path loss and fading. This causes the rear vehicles to experience more packet losses. The average number of repetitions required for delivering the first DENM to the FVs is presented in Table 8.8. The packet drops are the main cause of delay in Fig. 8.7, and therefore the total DENM delay is highly dependent on the DENM frequency (how long it takes before a repetition is made). However, it should be noted that the waiting time to gain access to the channel, which can be long when the CBR is high, also affects this delay. Config 1 demonstrates a very high DENM delay, especially for the last three vehicles in the platoon. This delay reduces drastically when the number of non-platooning vehicles and their CAM frequency is reduced, config 2. The difference in DENM delay caused only by the change of the CAM frequency of the non-platooning vehicles is demonstrated by the curves representing configs 2 and 3. The significantly higher delay with config 2 in comparison to config 3 is mainly because of a larger number of packet losses in dense traffic scenarios. Config 4 has a higher DENM frequency than config 5, but the CAM frequency of the non-platooning vehicles is kept the same, 10 Hz. In this case, we can see that a higher DENM frequency reduces the DENM delay by a large margin since repetitions come faster in case of lost packets. Therefore, a high DENM frequency can mitigate the DENM delay unless the channel load is very high. Finally, config 6 demonstrates an acceptable level of delay due to low vehicle density and CBR. One important thing to notice here is that contention-based MAC protocols such as CSMA perform poorly in high-density traffic scenarios, and this is not a new finding.

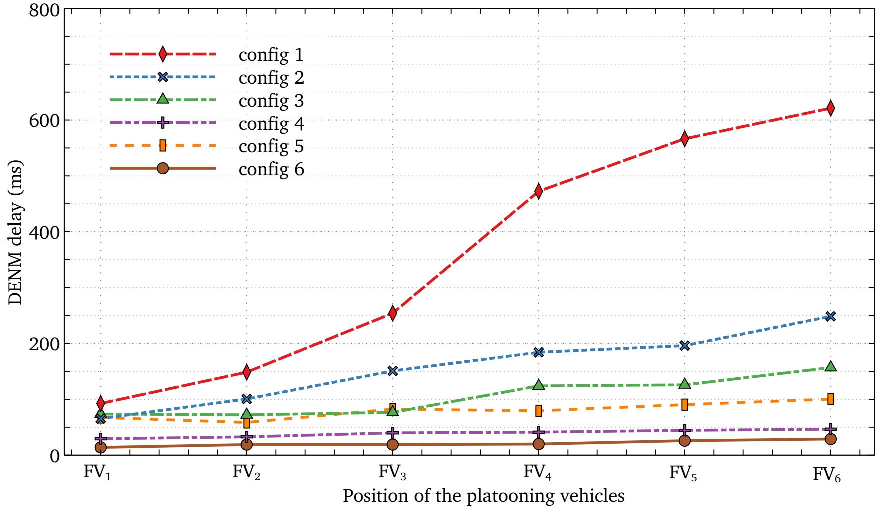


Figure 8.7: DENM delay for different configurations.

Table 8.8: Average number of repetitions before reception.

	config 1	config 2	config 3	config 4	config 5	config 6
FV_1	1.77	0.87	1.01	0.42	0.66	0.08
FV_2	2.89	1.37	0.99	0.47	0.57	0.18
FV_3	5.00	2.09	1.05	0.58	0.78	0.18
FV_4	9.36	2.57	1.73	0.60	0.81	0.19
FV_5	11.25	2.74	1.76	0.65	0.89	0.25
FV_6	12.33	3.49	2.20	0.68	0.99	0.25

Bilstrup *et al.* [47] also showed that the IEEE 802.11p MAC protocol exhibits very high channel access delay and packet drops at the MAC layer. Such a level of delay demands specialized emergency braking techniques and DCC algorithms [95] to ensure platoon safety.

ACK delay The propagation time of ACK packets in the upstream direction in a platoon can also be very high, especially when a vehicle accepts an ACK from its immediately following vehicle only, Fig. 8.8. For config 1, it requires 365 ms on average for the ACK packet to reach the LV from the last one, and this number is 314, 219, 90, 106, and 47 ms for configs 2, 3, 4, 5 and 6 respectively. Therefore, the ACK delay is not negligible in any of these cases. Considering both Figures 8.7 and 8.8, the average time between the first DENM transmitted by the LV and the first ACK received by the LV is 1066, 615, 432, 143, 225 and 94 ms for configurations 1, 2, 3, 4, 5 and 6 respectively. The AEB strategy proposed in this paper leverages this long delay in the event of

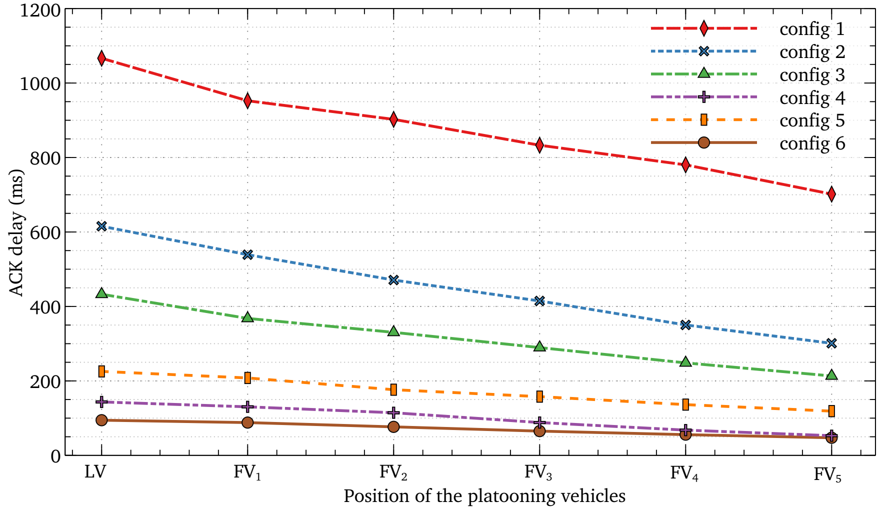


Figure 8.8: ACK delay for different configurations.

an emergency by performing soft deceleration instead of remaining idle while waiting for the ACK.

8.3.3 Simulation of the braking strategies

A study is performed by inspecting the acceleration profiles, minimum inter-vehicle distances, and the distance traversed by the LV. Ten simulation runs have been carried out for all six configurations and all four braking strategies. However, for the sake of conciseness, only the results of the most challenging configuration, config 1 are presented. For some simulations, e.g., SB-2, the results of the communication metrics determined in Section 8.3.2 are used as input. The results concerning the acceleration profiles are discussed in Section 6.2.7, which demonstrates how the platoon performs braking using each of the four braking strategies.

Minimum inter-vehicle distance ($d_i(t)_{min}$) The boxplot in Fig. 8.9 presents the minimum inter-vehicle distances after the platoon entirely stops for config 1; x marks the mean of 10 simulation runs. This figure shows if there are any collisions with the presented braking strategies. With GD-1, the platoon undergoes rear-end collisions for two simulation runs. This is because the FV_6 receives DENM long after the FV_5 starts braking with high deceleration rate. The vehicles make a significant gap with their preceding vehicles by decelerating at a higher rate, and thus the gap after complete stop can be very high with GD-1. With SB-2, there are also rear-end collisions for two simulation runs with config 1. This is due to the inadequate τ_{wait} time.

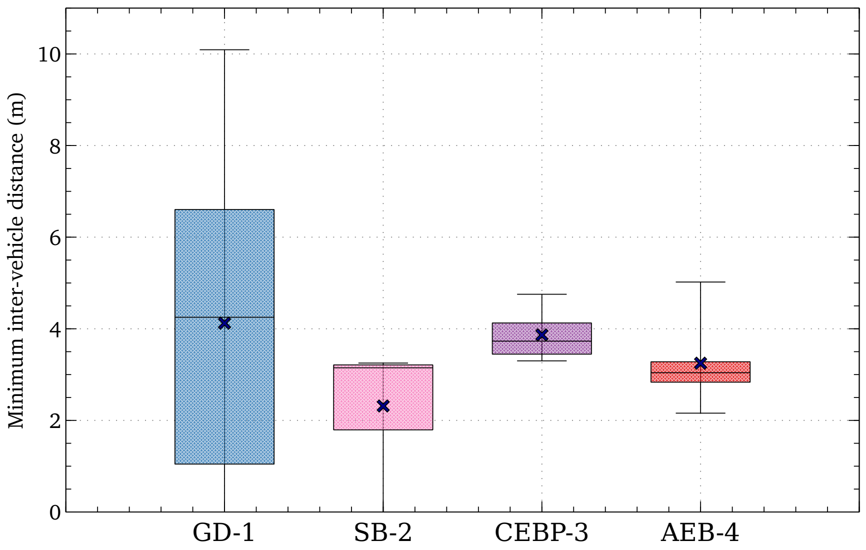


Figure 8.9: Minimum inter-vehicle distances $d_i(t)_{min}$ after the platoon has come into complete standstill, config 1, gap$_{des}$ = 5 m.

For instance, if all the vehicles except FV_6 receive DENM by $\tau_{wait} = 625$ ms and perform braking with full deceleration, it causes collisions in the platoon. GD-1 and SB-2 avoid a collision for all the runs under any of the other configurations. For brevity, all the results are not presented. CEBP-3 and the proposed AEB-4 successfully avoid collisions for all ten runs with each of the six configurations. However, it cannot yet be concluded that CEBP-3 and AEB-4 are fail-safe braking strategies if collision avoidance causes the LV to traverse too long distance.

Stopping distance of LV (d_L) Fig. 8.10 shows the average stopping distance of the LV for all four braking strategies for all six configurations. In GD-1, as the LV brakes with the same deceleration rate in all cases, the distance traversed is also always the same. The stopping distance is the highest among the presented braking strategies, 106.56 m. This is because the LV decelerates at a lower rate ($-4.4\ ms^{-2}$) to avoid collisions. In SB-2, the same τ_{wait} is not used for all the configurations. This is because τ_{wait} should be set according to the average DENM delay of the particular configuration [90]. To this end, τ_{wait} equal to 625 ms for config 1, 250 ms for config 2, 150 ms for config 3, and 100 ms for configs 4, 5, 6 are set according to Fig. 8.7. For τ_{wait} equal to 625, 250, 150 and 100 ms, the LV traverses 79.15, 68.03, 65.26 and 63.87 m respectively with SB-2. The stopping distance of the LV is the second-highest with CEBP-3 in configs 1, 2, and 3. This is because the LV re-

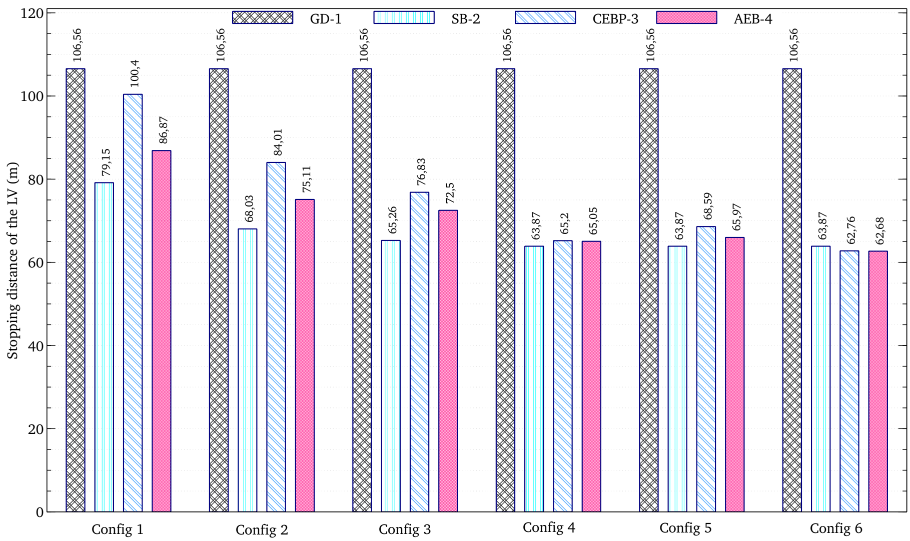


Figure 8.10: Stoping distance of the LV d_L for different configurations. For SB-2, τ_{wait} = 625 ms in config 1, 250 ms in config 2, 150 ms in config 3 and 100 ms in configs 4,5,6.

tains its speed until it receives ACK from the FV_1. The difference in stopping distance between the CEBP-3 and the proposed AEB-4 is not trivial. The LV traverses 13.53, 8.9, 4.33 m more with CEBP-3 compared to AEB-4 for configs 1, 2, and 3, respectively. However, for configurations 4, 5, and 6, there is no notable difference in distance traversed by the LV for CEBP-3 and AEB-4. This is because the LV does not get the chance for soft braking due to low DENM delay, which makes the two schemes basically the same. Therefore, in cases of low DENM latency, the proposed adaptive emergency braking strategy performs as good as the CEBP-3, whereas when the communication quality is bad, AEB-4 performs better. Among these four braking strategies, the SB-2 shows better performance in terms of the distance traversed by the LV. However, the LV needs to make a good prediction of the communication latency to avoid collisions.

8.4 Discussion

Communication assisted cooperative emergency braking is essential for fail-safe platooning while the vehicles are driving in close formation. The synchronized braking strategy shows good promise for avoiding collisions by waiting for a short time and facilitating braking at a very high deceleration rate. However, the waiting time τ_{wait} varies under different network loads which is difficult to predict during the design time. To address this problem, machine learning algorithms are planned to be employed in the future to forecast the waiting time during runtime. The other braking approaches such as GD, CEBP are good at avoiding collisions. Still, it comes at a cost that the lead vehicle needs to traverse a longer distance. The AEB-4 strategy, on the other hand, improves the CEBP-3 strategy to minimize the stopping distance of the lead vehicle. The simulation results demonstrate that the inter-vehicle distances after full-stop are shorter with AEB-4 than the CEBP-3 approach. The AEB-4 method leverages the inter-vehicle distance after full-stop to minimize the stopping distance of the LV. The assumption is that the platoon is still in the fail-safe state as long as there are no collisions, i.e., the inter-vehicle after full-stop takes lower precedence.

Chapter 9

Fuel-Efficient and Safe Platooning

In this chapter, the practical aspects of the platooning applications in the light of Chapters 5, 7, and 8 are discussed. First of all, the impacts of transient communication errors in determining the inter-vehicle distances and controllers are analyzed. Some conclusions are then inferred that suggest how to choose the packet loss thresholds for *fair* and *poor* communication quality. Later, the benefits and drawbacks of the SB and AEB strategies are highlighted. Finally, the appropriateness of the CEB methods for attaining the fail-safe state manifested in Figure 2.1 is discussed. Moreover, this chapter is aimed to be a technical conclusion of the platooning applications presented in this study.

The PlatoonSAFE simulator is the practical realization of the state machine presented in Figure 2.1. When it comes to fuel-efficient and safe platooning, the PLATOON controller shows promising performance by maintaining CDG of 5 m and exhibiting string stable behaviour in low packet loss scenarios. However, when the packet loss increases and the communication quality becomes so-called *fair* or *poor*, using the CDG approach becomes unsuitable. Because the tail vehicles in the platoon are far from the LV, and they experience more packet losses due to path loss and fading effects. Increasing the CDG does not necessarily ameliorate the situation. For instance, the tail vehicles in the platoon undergo collisions even with 15 m CDG while using the PLATOON controller and normal braking strategy, Figure 7.2c. Furthermore, increasing and decreasing the gap to the front vehicle due to poor communication quality endangers platoon safety. From a fuel-efficient point of view, also, frequent acceleration and deceleration contribute to more fuel-consumption. To this end, the RM proposed in this report suggests to switch to CACC controller when the communication quality of the ego vehicle with respect to the LV becomes *poor*. As the CACC controller relies on the beacons from the preceding vehicle and uses the CTG policy, the increased gap keeps the platoon acceptably safe. If the communication quality with the front vehicle also becomes *poor*, then ACC controller is adopted that maintains a longer CTG and does not require V2V communication. The simulation results presented in Section 7.4 demonstrate that the RM needs to use the ACC controller a very few times only. Gap adjustment and switching between the PLATOON and CACC controllers can maintain the fail-operational state. Two main factors contribute to the frequent state switching in the RM, e.g., the initial CDG and CTG values, and the packet loss thresholds for *fair* and *poor* communication quality. In harsh wireless environment, using higher CDG and CTG values prevent too frequent state switching, see Figures 7.6 and 7.7.

The number of packet losses that determine the *good*, *fair*, and *poor* thresholds regulate the switching between the fail-operational and fail-safe states. The results presented in Section 7.3 suggest that when the initial CDG is short, the threshold for *fair* communication quality should be small. This is because the ego vehicle requires to react to the packet losses fast due to short-inter vehicle distance. However, when the initial CDG is long (e.g., 10 m), the threshold for *fair* communication quality should be bigger. Because the tail vehicles experience more packet losses due to path loss and fading effects if the gap is increased too early. On the other hand, The threshold for *poor* communication quality controls the frequency of the state switching. The sinusoidal scenario described in Section 7.4 has been simulated for various combinations of *fair* and *poor* thresholds, initial CDG and CTG values. The RM can prevent collisions for all these combinations. This proves the robustness of the RM in maintaining the fail-operational state. However, in the braking scenario, there are some collision cases when the RM is used with normal braking strategy only, see Figures 7.3 and 7.4. To this end, coordinated emergency braking strategies are required to enable fail-safe states in platooning.

Two emergency braking strategies named Synchronized Braking and Adaptive Emergency Braking are proposed in this report. The synchronized braking strategy facilitates braking at a very high deceleration rate. The motivation behind this braking strategy is that the tolerable communication latency decreases with the increase of deceleration rate. In dense traffic scenarios, the tail vehicles in a platoon undergo rear-end collisions when the deceleration rate is high, Figures 8.2a, 8.3a, and 8.4a. To this end, all the vehicles in the platoon wait for a short time in synchronized braking strategy so that they can receive the DENMs broadcasted by the LV. The simulation results suggest that the stopping distance of the LV is 47.02 m when it waits for 100 ms to perform the synchronized braking, and all the vehicles can avoid collisions. In the same simulation scenario, the LV traverses 60.817 m with normal braking when it needs to decelerate at a slower rate to avoid collisions. Moreover, a suitable CDG is suggested in Section 8.2, when the vehicles wait for 100 ms before braking according to the SB strategy. Simulation results show that the platooning vehicles can retain the initial 7 m gap even after braking at a complete standstill. With SB strategy, the total time required for the entire platoon to completely stop also minimizes due to the high deceleration rate it facilitates.

The SB-2 and AEB-4 strategies have also been compared with GD-1 and

CEBP-3 approaches. The comparison has been performed under various levels of network loads. From Section 8.3.2, we can observe that the DENM delay experienced by the platooning vehicles reduce significantly when the CAM frequency of the non-platooning vehicles is decreased and/or the number of neighbouring vehicles reduce that generate interference. In dense traffic scenario, increasing the DENM frequency does not necessarily decrease the DENM delay if the channel busy ratio is high. The AEB-4 strategy aims towards avoiding collisions and minimizing the stopping distance of the LV in such dense traffic scenarios by performing soft deceleration until the vehicles receive an ACK from the immediate succeeding vehicle.

Given the benefits and drawbacks of the emergency braking strategies in CEB module, we can now argue which strategy is more appropriate for attaining the fail-safe state depicted in the state machine in Figure 2.1. When the communication quality is *poor*, the platooning vehicles should already adopt the ACC controller that does not require communication support in addition to the radar sensor. When the communication quality is *good*, all the SB-2, CEBP-3, and AEB-4 strategies can ensure the transition to the fail-safe state, Figure 8.10. However, from the *fuel sub-optimal* state in which the communication quality is *fair*, the SB-2 and AEB-4 strategies show better promise in transitioning to the fail-safe state. More specifically, the SB-2 method performs the best in terms of stopping distance and collisions avoidance despite relatively high waiting times. The GD-1 strategy shows the highest stopping distance; hence, it is not suitable when it comes to fail-safe platooning. Similarly, the CEBP-3 approach only focuses on collision avoidance, and it can do so by making the LV brake last, and the last vehicle brake last. However, the LV might have to wait for a long period (up to 2.5 s with config-1 in Table 8.6) which increases the stopping distance. It violates the criteria of the fail-safe state defined in this report.

Chapter 10

Conclusions and Future Work

10.1 Conclusions

A platoon is a safety-critical system as its failure can lead to death or severe injury of human lives, or damage to the equipment or environment. Any safety-critical system must have fail-operational features to mitigate the effects of the failures, and fail-safe features to safeguard lives in the events of emergency. To this end, the objective of this research work has been to devise mechanisms for ensuring fail-operational and fail-safe states in platooning in the presence of transient communication errors.

Platooning is a complex system that involves wireless vehicular networks, control theory, vehicle dynamics, and active safety systems. The research in this domain lacks a complete simulation framework that fits all these features. Moreover, a large scale Field Operational Test is often too expensive. The PlatoonSAFE simulator, which is a result of this research work, is developed to address this issue. In this simulator, contract-based safety assurance methods are implemented to maintain fail-operational platooning. In addition, four network-assisted cooperative emergency braking strategies are implemented in the simulator. The construction of the simulator is done in such a way that the users can derive their desired classes from the base classes, and override the virtual functions to accommodate their safety contracts and emergency braking strategies. The PlatoonSAFE tool is planned to be released to the research community.

As a result of extensive simulations, using the developed simulation tool, a state machine is proposed that demonstrates how a platooning vehicle should react when there are transient communication errors, permanent failures, or road hazards. Wireless communication that is the foundation of platoon-based driving is by nature unreliable, and the presence of transient errors is its intrinsic characteristic. It would be presumptuous to dissolve a platoon due to a temporary communication outage only. To this end, the state machine defines how a platooning vehicle should degrade its performance in terms of fuel-efficiency by increasing the gap to the vehicle in front, should the communication quality worsen. In the meantime, the vehicle keeps monitoring the communication quality and looks for an opportunity to transition back to the fuel-efficient state when the communication quality improves. In this research work, rigorous simulations have been performed with the PlatoonSAFE simulator to understand in which platooning state a vehicle should be considering the current number of packet losses. To this end, the number of packet losses that should be regarded as *good*, *fair*, and *poor* communication quality is suggested based on the simulated scenarios. A vehicle switches between different reliability regimes based on the perceived communication quality with respect to the lead vehicle and the front vehicle.

A control function termed the runtime manager is also designed and implemented in the PlatoonSAFE tool that monitors the communication quality during the runtime and assures a sufficient level of safety based on some predefined safety contracts in the events of transient errors. The efficacy of the runtime manager has been analyzed in terms of maintaining fail-operational platooning with the IEEE 802.11p-based communication system. The simulation results demonstrate that an eight-vehicle platoon can avoid collisions despite a very high number of packet losses. In fact, the developed runtime manager is able to ensure that a collision is avoided for all combinations of packet loss values that constitute the *good*, *fair*, and *poor* communication threshold. This proves the robustness of the runtime manager in maintaining fail-operational platooning. The runtime manager instructs the vehicles to adjust the gap to the preceding vehicle, and/or switch between different controllers, PLATOON, CACC, and ACC, based on the perceived communication quality. During the temporary communication outage, the platoon acts as a decentralized system, i.e., each vehicle employs the runtime manager independently in their systems. Thus, the vehicles are not string stable while there are errors because different vehicles use different inter-vehicle distances and different controllers. The motivation behind this approach is that safety takes the precedence over fuel-efficiency when it comes to dealing with errors in the communication system.

Two emergency braking strategies, namely *Synchronized Braking* and *Adaptive Emergency Braking* have been proposed to enable fail-safe platooning. Both these braking strategies have been tested under dense traffic scenarios to examine their robustness. These braking strategies, along with two other state-of-the-art braking approaches, have been implemented in the PlatoonSAFE simulator. They are compared in terms of the capability of avoiding collisions and the distance traversed by the lead vehicle. Collision avoidance ensures the safety within the platoon, and the LV must traverse a shorter distance to avoid the hazard that triggered the emergency braking. Six different network configurations were generated with varying loads that offer six different channel busy ratios. It is observed that the delay of the braking notification message and the ACK delay can be considerably high when the channel busy ratio is high. All the braking strategies have also been simulated under these six configurations. The synchronized braking strategy can avoid collisions and bring the platoon into a fail-safe state fast by braking at a very high deceleration rate. The adaptive Emergency Braking approach also avoids collisions, and more importantly, it causes the LV to traverse a shorter distance by performing soft braking at a lower deceleration rate while notification messages are being delivered to all the other platooning vehicles. These two braking strategies exhibit fail-safe behaviour even in very dense traffic scenario while the state-of-the-art braking approaches fail to do so.

10.2 Future Work

The planned future works are as follows:

- So far, the runtime manager and cooperative emergency braking modules have been analyzed separately. In future, more scenarios will be simulated and investigated using these two modules together.
- Successful emergency braking with synchronized braking strategy largely depends on the prediction accuracy of the waiting time that the platooning vehicles should pursue before performing the braking. In future work, machine learning algorithms are planned to be employed to predict the communication delay based on the previous traffic patterns and their corresponding delays. Moreover, the emergency braking strategy of a multi-brand platoon will be investigated in which the platooning vehicles would have different braking capacities and different physical properties.
- In future work, remote control of the platoon in which the lead vehicle would be regulated from a control room will be considered. To this end, Cellular V2X (C-V2X) communication will be applied.
- The PlatoonSAFE simulator is planned to be made available to the research community.

List of Figures

List of Tables

Acronyms

AC	Access Category.
ACC	Adaptive Cruise Control.
ACK	Acknowledgement.
AEB	Adaptive Emergency Braking.
C-ITS	Cooperative-Intelligent Transportation System.
C2F	Connection-to-Front.
C2L	Connection-to-Leader.
CACC	Cooperative Adaptive Cruise Control.
CAMs	Cooperative Awareness Messages.
CASs	Collision Avoidance Systems.
CBR	Channel Busy Ratio.
CCH	Control Channel.
CDG	Constant Distance Gap.
CEB	Cooperative Emergency Braking.
CEBP	Coordinated Emergency Brake Protocol.
CSMA	Carrier Sense Multiple Access.
CSMA/CA	Carrier Sense Multiple Access/ Collision Avoidance.
CTG	Constant Time Gap.
CW	Contention Window.
DCC	Decentralized Congestion Control.
DENM	Decentralized Environmental Notification Message.
EDCA	Enhanced Distributed Channel Access.
ETSI	European Telecommunications Standards Institute.
FTA	Fault Tree Analysis.
FVs	Following Vehicles.

GA	Gap Adjustment.
GD	Gradual Deceleration.
GUI	Graphical User Interface.
ITS	Intelligent Transportation System.
ITS-Ss	Intelligent Transport System-Stations.
LV	Lead Vehicle.
MAC	Medium Access Control.
PCMs	Platoon Control Messages.
PHY	Physical.
RM	Runtime Manager.
SB	Synchronized Braking.
SCH	Service Channel.
STDMA	Self-organizing Time Division Multiple Access.
SUMO	Simulation of Urban Mobility.
TDMA	Time Division Multiple Access.
V2V	Vehicle-to-Vehicle.
V2X	Vehicle-to-Everything.
VANET	Vehicular Ad hoc Network.
VCPS	Vehicular Cyber Physical System.
WAVE	Wireless Access in Vehicular Environment.
WSMP	Wave Short Message Protocol.

Bibliography

[1] "Roadmap to decarbonising European cars." https://www.transportenvironment.org/publications/roadmap-decarbonising-european-cars. accessed: 23.09.2020.

[2] "Lorry CO2 – why Europe needs standards." https://www.transportenvironment.org/publications/lorry-co2-%E2%80%93-why-europe-needs-standards. accessed: 23.09.2020.

[3] "MEPs approve new CO2 emissions limits for trucks." https://www.europarl.europa.eu/news/en/press-room/20190412IPR39009/meps-approve-new-co2-emissions-limits-for-trucks. accessed: 14.08.2020.

[4] C. Bonnet and H. Fritz, "Fuel consumption reduction in a platoon: Experimental results with two electronically coupled trucks at close spacing," tech. rep., *SAE Technical Paper*, 2000.

[5] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen, "A survey on platoon-based vehicular cyber-physical systems," *IEEE communications surveys & tutorials*, vol. 18, no. 1, pp. 263–284, 2015.

[6] P. Kavathekar and Y. Chen, "Vehicle platooning: A brief survey and categorization," in *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 829–845, Citeseer, 2011.

[7] J. Axelsson, "Safety in vehicle platooning: A systematic literature review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1033–1045, 2016.

[8] A. Alam, A. Gattami, K. H. Johansson, and C. J. Tomlin, "Guaranteeing safety for heavy duty vehicle platooning: Safe set computations and experimental evaluations," *Control Engineering Practice*, vol. 24, pp. 33–41, 2014.

[9] R. Zheng, K. Nakano, S. Yamabe, M. Aki, H. Nakamura, and Y. Suda, "Study on emergency-avoidance braking for the automatic platooning of

trucks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1748–1757, 2014.

[10] R. Zheng, K. Nakano, S. Yamabe, and Y. Suda, "Safety evaluation of system failures in formation and separation processes of automatic platooning of trucks," in *20th ITS World Congress ITS Japan*, 2013.

[11] S. Yamabe, R. Zheng, K. Nakano, Y. Suda, T. Takagi, and S. Kawahara, "Analysis on behaviors of a driver in the system failure in forming automatic platooning of trucks from manual driving," in *19th ITS World Congress, ITS Europe*, 2012.

[12] M. Aki, R. Zheng, K. Nakano, S. Yamabe, S.-Y. Lee, Y. Suda, Y. Suzuki, and H. Ishizaka, "Evaluation of safety of automatic platoon-driving with improved brake system," in *19th ITS World Congress, ITS Europe*, 2012.

[13] P. Lemmen, H. Fagerlind, T. Unselt, C. Rodarius, E. Infantes, and C. van der Zweep, "Assessment of integrated vehicle safety systems for improved vehicle safety," *Procedia-Social and Behavioral Sciences*, vol. 48, pp. 1632–1641, 2012.

[14] X. Ma and I. Andréasson, "Estimation of driver reaction time from car-following data: Application in evaluation of general motor–type model," *Transportation research record*, vol. 1965, no. 1, pp. 130–141, 2006.

[15] E. van Nunen, D. Tzempetzis, G. Koudijs, H. Nijmeijer, and M. van den Brand, "Towards a safety mechanism for platooning," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 502–507, IEEE, 2016.

[16] R. Kiefer and L. Angell, "A comparison of the effects of an analog versus digital speedometer on driver performance in a task environment similar to driving," *Vision in vehicles*, vol. 4, pp. 283–290, 1993.

[17] C. Nowakowski, S. E. Shladover, X.-Y. Lu, D. Thompson, and A. Kailas, "Cooperative adaptive cruise control (cacc) for truck platooning: Operational concept alternatives," *PATH research report*, 2015.

[18] R. Horowitz and P. Varaiya, "Control design of an automated highway system," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 913–925, 2000.

[19] R. Rajamani, *Vehicle dynamics and control.* Springer Science & Business Media, 2011.

[20] R. Rajamani, H.-S. Tan, B. K. Law, and W.-B. Zhang, "Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 4, pp. 695–708, 2000.

[21] Q. Xu and R. Sengupta, "Simulation, analysis, and comparison of acc and cacc in highway merging control," in *IEEE IV2003 Intelligent Vehicles Symposium*, Columbus, OH, USA, June 2003, pp 237–242.

[22] C. Nowakowski, J. O'Connell, S. Shladover, and D. Cody, "Cooperative adaptive cruise control: Driver acceptance of following gap settings less than one second," *Proceedings of the Human Factors and Ergonomics Society*, vol. 3, pp. 2033–2037, 09 2010.

[23] S. E. Shladover, C. Nowakowski, X.-Y. Lu, and R. Ferlis, "Cooperative adaptive cruise control: Definitions and operating concepts," *Transportation Research Record*, vol. 2489, no. 1, pp. 145–152, 2015.

[24] "Intelligent transport systems (ITS); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service," Nov. 2014.

[25] "Intelligent transport systems (ITS); vehicular communications; basic set of applications; part 3: Specification of decentralized environmental notification basic service," Nov. 2014.

[26] J. C. Knight, "Safety critical systems: challenges and directions," in *Proceedings of the 24th International Conference on Software Engineering. ICSE 2002*, pp. 547–550, 2002.

[27] D. S. Herrmann, *A Practical Guide to Security Engineering and Information Assurance.* USA: CRC Press, Inc., 2001.

[28] T. Bijlsma and T. Hendriks, "A fail-operational truck platooning architecture," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1819–1826, 2017.

[29] P. Pop, D. Scholle, H. Hansson, G. Widforss, and M. Rosqvist, "The safecop ecsel project: Safe cooperating cyber-physical systems using wireless communication," in *2016 Euromicro Conference on Digital System Design (DSD)*, pp. 532–538, 2016.

[30] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. Lo Cigno, "PLEXE: A Platooning Extension for Veins," in *2014 IEEE Vehicular Networking Conference (VNC)*, Paderborn, Germany, December 2014, pp. 53–60.

[31] "Waymo CEO Sees Driverless Trucking Catching on Faster Than Taxis." `https://www.ttnews.com/articles/waymo-ceo-sees-driverless-trucking-catching-faster-taxis`. accessed: 13.08.2020.

[32] S. international, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," *SAE International,(J3016)*, 2016.

[33] F. Michaud, P. Lepage, P. Frenette, D. Letourneau, and N. Gaubert, "Coordinated maneuvering of automated vehicles in platoons," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 437–447, 2006.

[34] L. Xu, L. Y. Wang, G. Yin, and H. Zhang, "Communication information structures and contents for enhanced safety of highway vehicle platoons," *IEEE Transactions on vehicular Technology*, vol. 63, no. 9, pp. 4206–4220, 2014.

[35] J. Ploeg, D. P. Shukla, N. van de Wouw, and H. Nijmeijer, "Controller synthesis for string stability of vehicle platoons," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 854–865, 2014.

[36] S. E. Shladover, C. Nowakowski, and X.-Y. Lu, "Using cooperative adaptive cruise control (cacc) to form high-performance vehicle streams. definitions, literature review and operational concept alternatives," in *UC Berkeley: California Partners for Advanced Transportation Technology*, 2014.

[37] F. Browand, J. Mcarthur, and C. Radovich, "Fuel saving achieved in the field test of two tandem trucks," *UC Berkeley, Institute of Transportation Studies*, 01 2004.

[38] T. Robinson, E. Chan, and E. Coelingh, "Operating platoons on public motorways: An introduction to the sartre platooning programme," in *17th ITS World Congress*, vol. 1, p. 12, 2010.

[39] "Intelligent transport systems (ITS); Radiocommunications equipment operating in the 5.855 MHz to 5.925 MHz frequency band," Feb. 2017.

[40] I. C. S. L. S. Committee *et al.*, "IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11p*, 2007.

[41] "Intelligent transport systems (ITS); access layer specification for intelligent transport systems operating in the 5 GHz frequency band," Nov. 2012.

[42] L. Miao, K. Djouani, B. Van Wyk, and Y. Hamam, "A survey of ieee 802.11p mac protocol," *Journal of Selected Areas in Telecommunications*, 09 2011.

[43] S. Subramanian, M. Werner, S. Liu, J. Jose, R. Lupoaie, and X. Wu, "Congestion control for vehicular safety: Synchronous and asynchronous mac algorithms," *VANET'12 - Proceedings of the 9th ACM International Workshop on VehiculAr Inter-NETworking, Systems, and Applications*, 06 2012.

[44] T. V. Nguyen, F. Baccelli, K. Zhu, S. Subramanian, and X. Wu, "A performance analysis of csma based broadcast protocol in vanets," in *2013 Proceedings IEEE INFOCOM*, pp. 2805–2813, 2013.

[45] R. Vilzmann and C. Bettstetter, "A survey on mac protocols for ad hoc networks with directional antennas," in *EUNICE 2005: Networks and Applications Towards a Ubiquitously Connected World*, pp. 187–200, Springer, 2006.

[46] A. U. Haq and K. Liu, "Review of tdma-based mac protocols for vehicular ad hoc networks," in *2018 IEEE 18th International Conference on Communication Technology (ICCT)*, pp. 459–467, IEEE, 2018.

[47] K. Bilstrup, E. Uhlemann, E. Ström, and U. Bilstrup, "On the ability of the 802.11 p mac method and stdma to support real-time vehicle-to-vehicle communication," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, no. 1, p. 902414, 2009.

[48] K. Sjoberg Bilstrup, E. Uhlemann, and E. G. Strom, "Scalability issues of the mac methods stdma and csma of ieee 802.11p when used in vanets," in *2010 IEEE International Conference on Communications Workshops*, pp. 1–5, 2010.

[49] M. Segata, B. Bloessl, S. Joerer, C. Sommer, M. Gerla, R. Lo Cigno, and F. Dressler, "Toward communication strategies for platooning: Simulative and experimental evaluation," *IEEE Transactions on Vehicular Technology*, vol. 64, pp. 5411–5423, Dec 2015.

[50] L. Zhang, Z. Liu, R. Zou, J. Guo, and Y. Liu, "A scalable csma and self-organizing tdma mac for ieee 802.11 p/1609. x in vanets," *Wireless Personal Communications*, vol. 74, no. 4, pp. 1197–1212, 2014.

[51] G. M. Abdalla, M. A. Abu-Rgheff, and S. Senouci, "Space-orthogonal frequency-time medium access control (soft mac) for vanet," in *2009 Global Information Infrastructure Symposium*, pp. 1–8, 2009.

[52] D. N. M. Dang, H. N. Dang, V. Nguyen, Z. Htike, and C. S. Hong, "Hermac: A hybrid efficient and reliable mac for vehicular ad hoc networks," in *2014 IEEE 28th International Conference on Advanced Information Networking and Applications*, pp. 186–193, IEEE, 2014.

[53] J. Ploeg, B. T. M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Washington, DC, USA, October 5-7, 2011, pp 260–265.

[54] M. Segata, *Safe and Efficient Communication Protocols for Platooning Control.* Phd thesis, University of Innsbruck, 2 2016.

[55] S. E. Shladover, "Path at 20—history and major milestones," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 4, pp. 584–592, 2007.

[56] M. Di Bernardo, A. Salvi, and S. Santini, "Distributed consensus strategy for platooning of vehicles in the presence of time-varying heterogeneous communication delays," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 102–112, 2015.

[57] Xiangheng Liu, A. Goldsmith, S. S. Mahal, and J. K. Hedrick, "Effects of communication delay on string stability in vehicle platoons," in *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.01TH8585)*, pp. 625–630, 2001.

[58] P. Fernandes and U. Nunes, "Platooning with ivc-enabled autonomous vehicles: Strategies to mitigate communication delays, improve safety and traffic flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 91–106, 2012.

[59] L. Zhang and G. Orosz, "Motif-based design for connected vehicle systems in presence of heterogeneous connectivity structures and time delays," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1638–1651, 2016.

[60] S. Santini, A. Salvi, A. S. Valente, A. Pescapé, M. Segata, and R. L. Cigno, "A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 1985–1999, 2016.

[61] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," *IEEE Transaction Mobile Computing*, vol. 10, pp. 3–15, Jan 2011.

[62] C. Lei, E. M. van Eenennaam, W. K. Wolterink, G. Karagiannis, G. Heijenk, and J. Ploeg, "Impact of packet loss on cacc string stability performance," in *2011 11th International Conference on ITS Telecommunications*, pp. 381–386, 2011.

[63] "Plexe: The platooning extension for veins." `http://plexe.car2x.org/tutorial/`. Accessed: 31.03.2019.

[64] D. K. Murthy and A. Masrur, "Braking in close following platoons: The law of the weakest," in *2016 Euromicro Conference on Digital System Design (DSD)*, pp. 613–620, 2016.

[65] T. Taleb, K. Ooi, and K. Hashimoto, "An efficient collision avoidance strategy for its systems," in *2008 IEEE Wireless Communications and Networking Conference*, pp. 2212–2217, 2008.

[66] G. Jornod, T. Nan, M. Schweins, A. E. Assaad, A. Kwoczek, and T. Kurner, “Sidelink technologies comparison for highway high-density platoon emergency braking,” in *2018 16th International Conference on Intelligent Transportation Systems Telecommunications (ITST)*, pp. 1–7, 2018.

[67] C. Flores, P. Merdrignac, R. de Charette, F. Navas, V. Milanés, and F. Nashashibi, “A cooperative car-following/emergency braking system with prediction-based pedestrian avoidance capabilities,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1837–1846, 2018.

[68] J. Thunberg, N. Lyamin, K. Sjöberg, and A. Vinel, “Vehicle-to-vehicle communications for platooning: Safety analysis,” *IEEE Networking Letters*, vol. 1, no. 4, pp. 168–172, 2019.

[69] C. Bergenhem, K. Meinke, and F. Ström, “Quantitative safety analysis of a coordinated emergency brake protocol for vehicle platoons,” in *International Symposium on Leveraging Applications of Formal Methods*, pp. 386–404, Springer, 2018.

[70] H. Suzuki and T. Izawa, “Intent inference of driver’s deceleration maneuver and its safety impact on a platoon of vehicles,” in *2018 3rd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, pp. 286–290, IEEE, 2018.

[71] R. Adler, I. Schaefer, M. Trapp, and A. Poetzsch-Heffter, “Component-based modeling and verification of dynamic adaptation in safety-critical embedded systems,” *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 10, Jan. 2011.

[72] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. Henzinger, and K. G. Larsen, “Contracts for System Design,” *Research Report RR-8147, INRIA*, Nov. 2012.

[73] I. Sljivo, B. Gallina, J. Carlson, and H. Hansson, “Strong and weak contract formalism for third-party component reuse,” in *2013 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pp. 359–364, 2013.

[74] S. Girs, I. Sljivo, and O. Jaradat, “Contract-based assurance for wireless cooperative functions of vehicular systems,” in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, Beijing, China, 2017.

[75] I. Sljivo, B. Gallina, and B. Kaiser, “Assuring degradation cascades of car platoons via contracts,” in *6th International Workshop on Next Generation*

of System Assurance Approaches for Safety-Critical Systems, Springer, Magdeburg, Germany, Sept. 2017, pp 317–329.

[76] E. van Nunen, J. Ploeg, A. M. Medina, and H. Nijmeijer, "Fault tolerancy in cooperative adaptive cruise control," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pp. 1184–1189, IEEE, 2013.

[77] E. van Nunen, R. Koch, L. Elshof, and B. Krosse, "Sensor safety for the european truck platooning challenge," in *2016 ITS World Congress*, pp. 306–311, 2016.

[78] K. G. Lim, C. H. Lee, R. K. Y. Chin, K. B. Yeo, and K. T. K. Teo, "Simulators for vehicular ad hoc network (vanet) development," in *2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pp. 1–4, IEEE, 2016.

[79] M. B. Behrisch, J. L. Erdmann, and D. Krajzewicz, "SUMO-Simulation of Urban MObility: an overview in: SIMUL 2011," in *The Third International Conference on Advances in System Simulation*, 2011.

[80] M. Treiber and A. Kesting, "Car-following models based on driving strategies," in *Traffic Flow Dynamics*, pp. 181–204, Springer, 2013.

[81] "Verkehr In Stadten SIMulationsmodell (VISSIM)." Available at `http://www.vissim.de`, 2015.

[82] M. Fiore, J. Harri, F. Filali, and C. Bonnet, "Vehicular mobility simulation for vanets," in *40th Annual Simulation Symposium (ANSS'07)*, pp. 301–309, 2007.

[83] "NS-3 Network Simulator." Available at `https://www.nsnam.org/`.

[84] A. Varga, "The omnet++ discrete event simulation system," in *Proc. ESM'2001*, Prague, June 2001.

[85] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, Marseille, France, 2008.

[86] A. Virdis, G. Stea, and G. Nardini, "Simulating lte/lte-advanced networks with simulte," in *Simulation and Modeling Methodologies, Technologies and Applications*, pp. 83–105, Springer, 2015.

[87] "INET Framework." Available at `https://inet.omnetpp.org/`.

[88] "Omnet++ discrete event simulator." `https://omnetpp.org/documentation/`. Accessed: 04.09.2020.

[89] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "Traci: an interface for coupling road traffic and network simulators," in *Proceedings of the 11th communications and networking simulation symposium*, pp. 155–163, 2008.

[90] S. Hasan, A. Balador, S. Girs, and E. Uhlemann, "Towards emergency braking as a fail-safe state in platooning: A simulative approach," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pp. 1–5, 2019.

[91] N. An, J. Mittag, and H. Hartenstein, "Designing fail-safe and traffic efficient 802.11p-based rear-end collision avoidance," in *Proc. IEEE Vehicular Networking Conference*, Paderborn, Germany, Dec 2014, pp. 9-16.

[92] S. Hasan, M. A. Al Ahad, I. Sljivo, A. Balador, S. Girs, and E. Lisova, "A fault-tolerant controller manager for platooning simulation," in *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 1–6, 2019.

[93] L. Xiao and F. Gao, "Practical string stability of platoon of adaptive cruise control vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1184–1194, 2011.

[94] M. Segata, B. Bloessl, S. Joerer, C. Sommer, M. Gerla, R. L. Cigno, and F. Dressler, "Toward communication strategies for platooning: simulative and experimental evaluation," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5411–5423, 2015.

[95] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.