Design and Formal Verification of an Adaptive Cruise Control Plus (ACC^+) System

DESIGN AND FORMAL VERIFICATION OF AN ADAPTIVE CRUISE CONTROL PLUS (ACC$^+$) SYSTEM

BY

SASAN VAKILI, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

© Copyright by Sasan Vakili, November 2014

All Rights Reserved

Master of Applied Science (2014) McMaster University

(Computing and Software) Hamilton, Ontario, Canada

TITLE:	Design and Formal Verification of an Adaptive Cruise Control Plus (ACC^+) System
AUTHOR:	Sasan Vakili B.Sc., (Electrical Engineering) Shiraz University, Shiraz, Iran
SUPERVISOR:	Dr. Mark Lawford
NUMBER OF PAGES:	xi, 101

Abstract

Stop-and-Go Adaptive Cruise Control (ACC^+) is an extension of Adaptive Cruise Control (ACC) that works at low speed as well as normal highway speeds to regulate the speed of the vehicle relative to the vehicle it is following. In this thesis, we design an ACC^+ controller for a scale model electric vehicle that ensures the robust performance of the system under various models of uncertainty. We capture the operation of the hybrid system via a state-chart model that performs mode switching between different digital controllers with additional decision logic to guarantee the collision freedom of the system under normal operation. We apply different controller design methods such as Linear Quadratic Regulator (LQR) and H-infinity and perform multiple simulation runs in MATLAB/Simulink to validate the performance of the proposed designs. We compare the practicality of our design with existing formally verified ACC designs from the literature. The comparisons show that the other formally verified designs exhibit unacceptable behaviour in the form of mode thrashing that produces excessive acceleration and deceleration of the vehicle.

While simulations provide some assurance of safe operation of the system design, they do not guarantee system safety under all possible cases. To increase confidence in the system, we use *Differential Dynamic Logic* ($\mathsf{d}\mathcal{L}$) to formally state environmental assumptions and prove safety goals, including collision freedom. The verification

is done in two stages. First, we identify the invariant required to ensure the safe operation of the system and we formally verify that the invariant preserves the safety property of any system with similar dynamics. This procedure provides a high level abstraction of a class of safe solutions for ACC$^+$ system designs. Second, we show that our ACC$^+$ system design is a refinement of the abstract model. The safety of the closed loop ACC$^+$ system is proven by verifying bounds on the system variables using the KeYmaera verification tool for hybrid systems. The thesis demonstrates how practical ACC$^+$ controller designs optimized for fuel economy, passenger comfort, *etc.*, can be verified by showing that they are a refinement of the abstract high level design.

Acknowledgements

My experience as a graduate student has been an unforgettable journey made possible by Prof. Mark Lawford. I would like to express my sincere gratefulness and respect to Prof. Lawford for his generous support, trust, and advice throughout my graduate studies. I am proud to have Prof. Lawford as my academic mentor who has always inspired, valued and supported my learning unconditionally.

Special thanks to Dr. Neeraj Kumar Singh for his helpful technical discussions with regards to my work. My great appreciation goes to my friends Mrs. Sahar Kokaly and Ms. Golnaz Roshankar for their contribution in proofreading the written portion of my thesis. Many thanks to the McSCert Group, especially the NECSIS team, for their kindness and friendship.

Finally, my deepest thanks goes to my family wholeheartedly to whom I owe every accomplishment in my life.

Notation and abbreviations

Terms & Symbols

v_h:	The velocity of the host vehicle
v_l:	The velocity of the lead vehicle
a_h:	The acceleration of host vehicle
a_l:	The acceleration of lead vehicle
d_{gap}:	The relative distance between the host vehicle and lead vehicle
v_{set}:	The desired velocity of the host vehicle
B:	The absolute value of deceleration achieved by maximum brake force which depends upon the current vehicle weight and road conditions
h_{set}:	The desired following time gap between two successive vehicle (headway)
ϵ:	The maximum response delay from any actuators (ie. engine, brake etc.)

$f_{gap}(v_l, v_h, a_h)$: The distance it takes for the host vehicle to match the lead vehicle's velocity and be following at the desired headway h_{set} using acceleration a_h

$sc_{gap}(v_l, v_h)$: The distance at which ACC$^+$ system switches into safety critical mode

Contents

List of Figures

Chapter 1

Background and Literature Review

1.1 Introduction

In recent years, the automotive industry has become highly reliant on Software Engineering. Since software for embedded systems in the automotive domain is growing exponentially, the application of formal methods in specifying and designing such software is a promising yet challenging area (Bowen and Stavridou, 1993). Increasing demands for high quality and safety, and the shortcomings of the informal techniques applied in traditional software development have motivated the application of semi-formal or formal methods to facilitate a higher degree of automation and tool support for verification and validation purposes.

Every year, car crashes result in permanent disabilities and the loss of thousands of lives, leading to annual costs of billions of dollars in the United States only (Zaloshnja *et al.*, 2004). Although the majority of these car crashes are due to human error, failure in hardware or software components can also lead to accidents and unduly risk human life (Peters and Peters, 2003). While hardware failures are typically some

kind of *random failures* caused by different wear effects, such as corrosion, thermal stressing, *etc.*, software failures are *systematic failures* that may be introduced by human error during the system development. However, it is difficult to predict or detect the occurrence of all systematic software-related failures by classical means, such as testing and inspections (Parnas *et al.*, 1990).

Formal verification is an effective approach to help ensure the safety and reliability of complex automotive control systems, which can provide an additional level of confidence. This work contributes to the formal verification of automotive hybrid systems by using *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) to analyze the correctness of a high level Adaptive Cruise Control Plus (ACC^+) design. The verification method used ensures that the safety of the system is robust with respect to plant model variation. Formal development and verification of hybrid systems using $\mathsf{d}\mathcal{L}$ satisfies safety and performance requirements if the models used represent the system correctly. Using parametric constraints, we can find a region of safe operation for a continuous controller when we have an upper bound limit on response time in the presence of disturbances and uncertainties. In addition, the precise system descriptions, done through formal verification, can expose the problematic aspects of the system requirements. Based on the formal model of the system, the analysis techniques are required to establish the system correctness in accordance with the system requirements.

Another motivation of this work is to propose an implementation of an ACC^+ controller based on hierarchical design. The safety of practical ACC^+ controller designs, optimized for fuel economy, passenger comfort, *etc.*, can then be verified by showing their conformance to the high level design. Our ACC^+ hybrid system is implemented using *Simulink*[1] and is tested on a scale model Radio Controlled (RC) car test bench

[1] http://www.mathworks.com

forst described in (Breimer, 2013), to help ensure correctness and collision-freedom of the system under all possible scenarios in practice.

1.2 Adaptive Cruise Control Plus

In order to understand an ACC$^+$ system, one should first understand its predecessors, the Cruise Control (CC) and Adaptive Cruise Control (ACC) systems. The first generation of conventional Cruise Control (CC) systems, developed in the 1950s, could mechanically hold the throttle in a fixed position. Proportional feedback controllers were implemented in CC systems during the late 1950s to early 1960s. However, this system became more reliable and efficient than before when microchips became available in the market in the 1980s (Xiao and Gao, 2010). A CC system is beneficial in the absence of a lead vehicle or an obstacle in front of the car; however, it loses its effectiveness in traffic and congested roads.

The Adaptive Cruise Control (ACC) system is an improvement to CC system as it allows the controlling of the car's velocity and distance from the lead vehicle in different situations. In 1968, Fenton's group at Ohio State University captured the Functional requirements of highway automation for ACC systems (Xiao and Gao, 2010). These requirements are:

- Maintaining vehicle dynamic stability
- String stability
- Constant velocity in cruise state
- Driving comfort

- Minimizing traffic congestion to improve the traffic capacity

Maintaining vehicle dynamic stability of ACC systems means that the ACC system spacing response error converges to zero if the lead vehicle is travelling with a constant velocity, while this error may be non-zero if the leader's velocity is not a constant value (Xiao and Gao, 2010). String stability ensures that this spacing error does not amplify the space for the string of vehicles (Xiao and Gao, 2010). The main function of ACC is to maintain the desired speed that is set by the driver. For example, if the current speed of a lead vehicle is less than the speed of the host vehicle, the ACC system starts to control the speed of the host vehicle to maintain a desired safe distance between the two vehicles. Automatic adjustment of the host vehicle's acceleration allows the host vehicle to adjust its speed according to the traffic conditions without driver intervention. Any required braking action carried out by an ACC system will typically not exceed 30% of the host vehicle's maximum deceleration. When a stronger deceleration is needed, the driver is warned by an auditory signal and a warning message is displayed on a driver information screen. The driver can override the ACC system at any time to regain control of the vehicle.

According to (Vahidi and Eskandarian, 2003), a well-designed ACC system should improve the driving comfort, as well as traffic flow with safety considerations. There needs to be a guarantee that the designed ACC system will always behave correctly and safely while respecting the rules regarding passenger comfort (e.g., avoiding excessive changes in acceleration which would cause discomfort). Comfort has been considered in (Yi and Chung, 2001; Hoberock, 1976), as a parameter in terms of jerk or magnitude of acceleration, by defining a bounded criteria. Traffic conditions can

be considered in an ACC system design to help decrease traffic congestion by providing a smooth traffic flow. Improving traffic congestion using ACC systems has been investigated in the work of (Jerath and Brennan, 2010) through traffic flow simulations. They have concluded that increasing ACC systems in automobiles allows the traffic systems to operate in increased densities and flow. Furthermore, by introducing a full velocity and acceleration difference model for evaluating the impact of ACC systems on traffic flow, it was determined that traffic congestion varies significantly with acceleration in different situations (Zhao and Gao, 2005). In another work traffic flow has been studied in terms of the reaction time of the system and inter-vehicle gap by defining a relaxation equation for each vehicle (Tordeux *et al.*, 2010). Furthermore, (Kesting *et al.*, 2007) proposed an ACC system by considering a certain driving style for every traffic scenario. This system has been tested in afternoon traffic peak periods, as a worse case scenario, in order to measure the congestion improvement that might be achieved.

There is a compromise between traffic improvement and the distance between vehicles. Traffic is improved if the distance within a string of vehicles is reduced. However, shortening the distance between two vehicles threatens the safety of driving. The desired distance between two successive vehicles is defined in terms of their time gap and is referred to as *Time Headway*. The time gap is the time that it takes for a follower vehicle to reach a reference point passed by the lead vehicle at time zero. In the ACC domain, Time Headway has been preferably used over distance since it can guarantee string stability. Constant Time Headway (CTH) and Variable Time Headway (VTH) are two headway control approaches for ACC design. CTH can be stressful for passengers while VTH may not improve traffic congestion in some

cases (Santhanakrishnan and Rajamani, 2003; Hoedemaeker, 2000).

Considering all the functional requirements for ACC systems, as explained above, ACC systems, similar to other automated vehicle technologies, have a hierarchical architecture. This structure consists of a high-level controller and a low-level controller. The high-level controller is a finite state machine for decision-making in different mode-switching situations, which entails some necessary conditions that can be derived using kinematics equations. The low-level controller consists of a continuous controller for tracking the desired behaviour in a particular mode, which requires a good understanding of the vehicle dynamics.

A wide range of control methods, such as simple Proportional-Derivative (PD) feedback control, Model Predictive Control (MPC), Linear Quadratic Regulators (LQR) and Sliding Mode Control have been used for designing the lower and upper level controllers with respect to different objectives and constraints, from safety to comfort and fuel efficiency. Low-level control in most ACC systems applies the throttle and/or brake to achieve the required rate of acceleration/deceleration. Different mathematical models, such as sliding mode (Gerdes and Hedrick, 1997) and optimal dynamic back-stepping control (Lu *et al.*, 2001) have been used for deriving the desired acceleration by the supervisory controller. The LQR method has been used in the work of (Junaid *et al.*, 2005) to determine the ACC objectives while others have applied Sliding Mode Control for this purpose (Bin *et al.*, 2004; Hedrick and Yip, 2000; Dew, 2002). In addition, MPC for ACC has been proposed in a hierarchical architecture. The constraints and desired objectives have been formulated using MPC as high-level controller in (Naus *et al.*, 2008). Comfort and traffic requirements have also been considered in their design. In (Li *et al.*, 2011), it has been shown that

MPC is useful for designing a system to meet different criteria, such as fuel economy, tracking accuracy and driver comfort. In (Kural and B.A., 2010), MPC has been used to naturally integrate constraints into the optimization process. In the same work, the proposed ACC model has been tested for different traffic situations.

Another perspective on ACC design has been presented by (Shakouri and Ordys, 2011). The high-level controller in their hybrid system sends the desired velocity command to a low-level PI controller. Two non-linear models have been used for the vehicles dynamics depending on the application of throttle or brake. On the other hand, Girard and colleagues (Girard *et al.*, 2005) devised a hybrid system, which switches between the different modes of operation, in order to pass the desired acceleration to a low-level controller. The low-level controller chooses an appropriate action by manipulating the throttle or brake. PD controller, Adaptive Cruise Control, and Coordinated Adaptive Cruise Control have been used to achieve the desired set point velocity or distance based on the absence or presence of some conditions, such as the presence of a lead vehicle and/or wireless communication.

Another group of researchers proposed different features for ACC, such as the ability to maintain a constant velocity in the absence of a lead vehicle (Shigeharu *et al.*, 2010). Their system uses the throttle or the brake to decelerate the vehicle when it detects a lead vehicle travelling at a slower speed. It also tracks the headway time as the distance gap.

Although a wide range of approaches have been proposed in the literature to design various ACC systems with different objectives, ACC systems have a minimum speed threshold, such as 30 km/h, below which they stop operating; hence an ACC system does not deal with stop-and-go traffic (Shakouri and Ordys, 2011; Naranjo

et al., 2006).

Stop-and-Go Adaptive Cruise Control, also known as Adaptive Cruise Control Plus (ACC^+), is a system that operates at all velocities greater than or equal to 0 km/h. It is an extension of the ACC system that essentially consists of a superset of the features found in ACC. In particular, ACC^+ is designed to provide controllability at very low-speed driving scenarios.

An important component in any ACC^+ design is to obtain information about its environment, such as the speed of the lead vehicle, the user's desired speed, what constitutes a safe distance between the host and the lead vehicle, *etc.*, in order to meet its requirements. This can be achieved by using a set of sensors that monitor the environment at sufficiently high sampling rates to capture the continuous behaviour of the system as precisely as possible. For instance, a sensor can be mounted on the front of a vehicle to measure the distance to the nearest object within its zone of operation. Since the system must both accelerate and decelerate the host vehicle based on the information obtained from the environment, an ACC^+ system must be able to adjust the throttle and/or brake using appropriate control signals.

In the past two decades, there have been several studies on ACC+ design with respect to its various functional requirements. In the work of (Yamamura *et al.*, 2001) the design of an ACC^+ system was discussed, taking into account some of the challenges that arise from low-speed driving, such as smaller inter-vehicular spacing and frequent changes in velocity. In (Yi *et al.*, 2001), a control algorithm was developed using linear quadratic optimal control theory for an ACC^+ system. Their algorithm defined the desired acceleration of the follower vehicle based on its speed and distance. A throttle-brake control law was investigated, by applying a torque converter,

to track the desired acceleration. However, only a constant speed was considered for the lead vehicle in their simulations. A vehicle model in the form of a first-order differential equation has been proposed for the engine and transmission system of the vehicle by (Eizad and Vlacic, 2004). However, their experiments on electrical vehicles cannot verify the control algorithm presented for all speeds. Fuzzy logic theory was applied to ACC$^+$ design by (Naranjo *et al.*, 2006), where the input information was gathered from a Differential Global Positioning System (D-GPS) and a wireless area network. Driver behavior was incorporated into an ACC$^+$ design by (Persson *et al.*, 1999). (Persson *et al.*, 1999; Naranjo *et al.*, 2006) suggested that autonomous systems can increase safety if they act similar to driver behavior. However, this is not necessarily an appropriate conclusion because 90% of the accidents happen due to human error.

In addition, (Bin *et al.*, 2004) designed an ACC$^+$ system based on Model Matching Control (MMC) using a Sliding Mode Control (SMC) method to track the vehicle's desired acceleration. Robustness and good response time are the advantages of this design. Another robust ACC$^+$ control has been proposed by (Villagra *et al.*, 2009), where robustness has been investigated with respect to noise in the measurements of sensors by using nonlinear estimation methods. Although their focus was ACC$^+$ systems, it did not address the engine and brake dynamics nor did it take into account any uncertainties. (Martinez and Canudas-de Wit, 2007) proposed a nonlinear reference model-based longitudinal control with safety constraints and comfort specifications. Their system consists of an *inner force* control loop for acceleration and brake systems compensation and an *outer inter-distance* compensator for tracking the desired inter- vehicle distance. However, this model is very sensitive to the estimation

of the leader vehicle's acceleration resulting in a poor Signal to Noise Ratio (SNR) and the following vehicle has a larger jerk compared to the leader.

Sensitivity to vehicle parameters variation and uncertainty are two other factors that cannot be ignored in the design of ACC$^+$ system. No mathematical model can represent a physical system with 100% precision (Doyle *et al.*, 1992). Also "*The mass of a heavy duty vehicle can vary by as large as 400%*" (Vahidi and Eskandarian, 2003). Therefore, different types of uncertainty should be taken into account during the process of controller design. Considering all of these aspects leads to complexity in the system and difficulty to assure safety and correctness.

Safety is a crucial issue in ACC and ACC+ systems. The purpose of automotive vehicle control systems is to reduce workload and pressure on the driver, hence improve safe and collision free driving. However, such systems face challenges due to their impact on the driver behaviour in different situations. According to (Xiao and Gao, 2010), ACC systems can reduce the awareness, work-load and stress of the driver, but can also increase the mental workload needed to supervise the system. Another belief is that ACC reduces the stress of driving in dense traffic, while increasing the reaction time of the driver (Sathiyan *et al.*, 2013). This fact may cause failure in critical situations because such systems increase the reaction time of the driver in regaining the control of the vehicle during critical circumstances. Consequently, an unsafe system can increase the chances of collision and risk to human life. Therefore, the safety of these systems should be proven to ensure error reduction. In the comprehensive work of (Xiao and Gao, 2010), it has been indicated that the average safety improvements achieved by automatic vehicle control is 8%, 10%, 20%, 12% and 1% for lane change, obstacles, rear-end collision with queue, rear-end collision without

queue and road departure respectively. Proving safety and correctness of an ACC$^+$ system in any possible situation for any type of vehicle can increase the reliability of the system before its deployment into the market.

The goal of our work is to design a Robust Adaptive Cruise Control Plus system by considering all the mentioned aspects. In other words, we propose a robust ACC$^+$ system that provides sufficient assurances of safety for typical operating conditions. The ACC$^+$ presented in our work is a Stop-and-Go system that avoids uncomfortable jerks and other discomfort for passengers, except when in safety critical situations. Sensitivity to vehicle parameters variation and uncertainty has been considered in the design of the high-level and low-level controllers of our ACC$^+$ system using robust control methods. Finally, formal methods have been used in our work to provide safety assurance using *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) (Platzer, 2010).

1.3 Formal Verification

Automotive control is a broad and interesting area that has been studied by academic and industrial researchers in an effort to minimize the risk, and improve the safety of driving. Since these kind of systems deal with human life, even a small error or mistake in the design of these systems can lead to irreparable harm. Therefore, sufficient safety-assurance is necessary before deployment of any such system.

Several papers have reported work on the simulation of ACC (Verburg *et al.*, 2002; Gietelink *et al.*, 2009; Arioui *et al.*, 2009; Nehaoua *et al.*, 2008). However, these simulations are not enough to guarantee that the tested system is safe and collision-free under all traffic conditions.

Our ACC$^+$ design is a hybrid system. Hybrid systems integrate both continuous

and discrete dynamics, bringing together several research fields in order to address safe operations. Logic plays a significant role in formal verification of hybrid systems from reachability analysis to undecidability in theory and practice.

Several approaches have been proposed in the literature to verify safety properties of hybrid systems. An inductive method was proposed by (Abrahám-Mumm *et al.*, 2001) based on the PVS theorem prover to verify the required safety properties of the parallel hybrid systems. Decidability and complexity analysis for the verification of non-parametric reasonable linear hybrid automata was proposed by (Damm *et al.*, 2011), where an SMT solver was used to verify the safety properties, and time-bounded reachability. A counterexample-guided verification approach using a model checker has been used for verifying a cruise control system to reduce the computational cost. In this approach a sequence of abstractions was used to identify the unwanted behaviours (Stursberg *et al.*, 2004). In (Jairam *et al.*, 2008), a MEMS-based ACC system was verified using Simulink and semi-formal approaches. A case study was developed and the system was validated through a transformation-based approach. Another interesting work, done by (Ciobanu and Rusu, 2008) described the ACC system theoretically by process algebra (*timed distributed pi-calculus*), analyzed the informal requirements of the system, verified the properties (such as deadlocks) using the Mobility Workbench model checker.

In addition, Platzer *et al.* (Platzer, 2008, 2012) proposed a dynamic logic and proof calculus for verifying hybrid systems. Dynamic logic incorporates continuous evolutions during discrete behaviours and transitions between the states. Moreover, in the past few years Loos and Platzer in (Loos *et al.*, 2011, 2013) have published several papers addressing fundamental principles of ACC, including important safety properties

in various scenarios. However, none of their work provides a feasible solution for implementation purposes. For instance, one work discussed formal verification of ACC, examining the required distance for avoiding collision when an arbitrary number of cars are moving on a street including the case that a new car enters the lane (Loos *et al.*, 2011). In another work, Loos and colleagues (Loos *et al.*, 2013) proposed an ACC model based on different acceleration choices for different modes of operation using various conditions. However, this model has overlaps among its modes, which causes mode thrashing due to the improper guard conditions defined in the paper. The mode thrashing has the potential to result in acceleration changes that would be unacceptable in terms of driver comfort and fuel efficiency. In addition, the second and fourth controller modes in (Loos *et al.*, 2013) are unreachable regardless of the plant model. Moreover, (Loos *et al.*, 2013) proposed an acceleration formula for the third controller mode using the square root of some parameters such as communication time (τ). However, the optimal τ assumed as the maximum communication time, 3.2 seconds, is unrealistic for a real-time application due to slow functionality of the system. These two proposed solutions, (Loos *et al.*, 2013, 2011), for ACC have not taken any desired set point velocity or distance into account, which is required during the formalization of a system that serves the main purpose of ACC.

In response to the work discussed in (Loos *et al.*, 2011), in (Aréchiga *et al.*, 2012) a PID controller was proposed to maintain a desired distance between a host vehicle following a lead vehicle, describing acceleration in terms of the position and velocity of the vehicles. However, a large desired distance is attained in their controller design, which makes the system unrealistic, and the proposed system exits its safe boundaries if a small desired distance is considered. The problem of this system is that the

specified operating regime for the controller is outside its safe boundaries for small set point values and the controller will not take any action in some unsafe cases. Therefore, a large set point has to be considered to satisfy safety conditions.

In this work, we aim to provide a complete solution for a practical, generic ACC$^+$ system design that guarantees the safety properties outlined by (Loos *et al.*, 2011; Aréchiga *et al.*, 2012; Loos *et al.*, 2013). In addition, in our design we incorporate practical ACC$^+$ requirements, such as headway reference tracking, while respecting the user-specified maximum velocity constraint. Our proposed solution allows the host vehicle to maintain a desired velocity in the absence of a slower lead vehicle or obstacle, and to safely approach a slower lead vehicle within a desired safe distance. Furthermore, we investigate the safety of critical cases as a separate mode of operation in order to guarantee safety and collision-freedom.

1.4 Thesis Contributions

The contributions of this thesis are:

1. Proposing a new high-level design for ACC$^+$.
2. Formalization of the ACC$^+$ requirements using $\mathsf{d}\mathcal{L}$ (Platzer, 2010).
3. Formal verification of the new design's safety properties using the KeYmaera theorem prover (Platzer and Quesel, 2008).
4. A complete solution for a practical, generic ACC$^+$ system design that guarantees the safety properties.

5. Practical ACC$^+$ controller designs optimized for fuel economy, passenger comfort, *etc.*, can then be verified by showing their correspondence with the high level design.

6. Design and implementation of the proposed Robust Adaptive Cruise Control Plus (ACC$^+$).

Chapter 2

Preliminaries

In this chapter we provide preliminaries and backgrounds required for our work. The ACC$^+$ system proposed in our work has a hierarchy structure which consists of a Finite state machine as a high level controller and a low level continuous controller. The low level continuous controllers is designed to capture model uncertainties by using robust control theory. Therefore, in Section 2.1 we will first describe the foundation of robust control theory. Second, we will explain the interaction between the high level and low level controllers in terms of hybrid system in Section 2.2. Since our ACC$^+$ system is modelled in Matlab/Simulink, we will review Simulink briefly in Section 2.3. Finally, we aim to prove the safety of our ACC$^+$ system by using *differential dynamic logic (dℒ)*. Hence, the required background in dℒ and its tool is provided in Section 2.4 and Section 2.5 respectively.

2.1 Robust FeedBack Control Theory

The stability and performance of a control system can be described in terms of the size of a signal of interest. The signal size can be studied from the definition of Norms presented in (Doyle *et al.*, 1992):

Suppose $u(t)$ is a signal in the time domain mapping $(-\infty, \infty) \rightarrow \mathbb{R}$, then the norms of this signal are defined as follows:

1-Norm is the integral of the signal's absolute value:

$$\|u\|_1 := \int_{-\infty}^{\infty} |u(t)| \mathrm{d}t$$

2-Norm is the square root of signal's energy:

$$\|u\|_2 := (\int_{-\infty}^{\infty} u(t)^2 \mathrm{d}t)^{1/2}$$

∞-Norm is the least upper bound of the signal's absolute value:

$$\|u\|_\infty := \sup_t |u(t)|$$

The system Norms are used to evaluate the output signal based on the input and the system. Suppose G is a linear, time-invariant, and causal system. Then, the second and infinity norms of the transfer function $\hat{G}$ can be defined as follow (Note that G and $\hat{G}$ are mathematical representations of the system in the time and frequency domain, respectively).

2-Norm:

$$\|\hat{G}\|_2 := (\frac{1}{2\pi} \int_{-\infty}^{\infty} |\hat{G}(j\omega)|^2 \mathrm{d}\omega)^{1/2}$$

∞-Norm:

$$\|\hat{G}\|_\infty := \sup_\omega |\hat{G}(j\omega)|$$

Now, we can find out how big the output signal ($y(t)$) would be by computing the norms of the input signal ($u(t)$) and the system transfer function ($\hat{G}$). The results are represented in the Table 2.1 provided by (Doyle *et al.*, 1992).

	$u(t) = \delta(t)$	$u(t) = sin(\omega t)$	$\|u\|_2$	$\|u\|_\infty$
$\|y\|_2$	$\|\hat{G}\|_2$	∞	$\|\hat{G}\|_\infty$	∞
$\|y\|_\infty$	$\|\hat{G}\|_\infty$	$\|\hat{G}(j\omega)\|$	$\|\hat{G}\|_2$	$\|\hat{G}\|_1$

Table 2.1: Output norms for different inputs

Table 2.1 demonstrates how much the input signal, u, can affect the output, y, for a stable system G. This table shows the second and infinity norms of y for impulse and sinusoid signals as inputs in the first two columns. The third and fourth columns of this table provide the norms of y for unfixed signals which is bounded with some conditions, which turn out to be the least upper bound on the 2-norm and ∞-norm of the output ($\|y\|_2$ and $\|y\|_\infty$), while input signal (u) should be any signal of 2-norm ≤ 1 and/or ∞-norm ≤ 1, that is,

$sup\{\|y\|_2 : \|u\|_2 \leq 1\}$

$sup\{\|y\|_\infty : \|u\|_\infty \leq 1\}$

The ∞ in the other entries is valid for the case that there is some ω such that $\hat{G}(j\omega) \neq 0$ (Doyle *et al.*, 1992).

The performance specification of a control system is a good tracking of a reference signal. Perfect asymptotic tracking of a single signal is a common method in the field of control, such as designing PID controller for a step or ramp reference signal. However, this method is not applicable for maintaining a reasonable performance in

the presence of uncertainties and for a set of reference signals. Therefore, robust performance in terms of weighted norm bound can handle this issue. Assuming that the unity feedback system is internally stable and has open loop transfer function L, then the transfer function from reference signal r to tracking error e is called *sensitivity function* and can be found as $S := \frac{1}{1+L}$ (Doyle *et al.*, 1992). We suppose that a set of possible r values have the least upper bound amplitude ≤ 1. Therefore, good tracking, i.e. acceptable performance tracking, can be stated as $\|S\|_\infty < \epsilon$. This fact can be rewritten as $\|W_1 S\|_\infty < 1$ by considering the weighting function $W_1(s) = 1/\epsilon$ (Doyle *et al.*, 1992).

Besides this nominal performance for a set of reference signals, a set of plant models should be considered for the physical system. No mathematical model can represent the exact physical system (Doyle *et al.*, 1992). Uncertainty can be captured in terms of variation of some variables in the physical system or uncertainty in modelling the physical system as unmodeled dynamics. Therefore, among all structured and unstructured uncertainties we choose unstructured perturbation as disk-like multiplicative uncertainty to simplify our analysis. Suppose that the nominal plant transfer function is P, then we can model disk-like perturbation as $\tilde{P} = (1 + \Delta W_2)P$, where W_2 is a fixed stable transfer function (the weight) and Δ is a variable stable transfer function satisfying $\|\Delta\|_\infty < 1$ (ΔW_2 is the normalized plant perturbation). Also, it is considered that nominal transfer function P and its perturbed $\tilde{P}$ have the same unstable poles at the right half plane (Doyle *et al.*, 1992).

A controller is robust with respect to a characteristic if that characteristic holds for all the plant models in the set of perturbed plants . Therefore, a controller provides robust stability if it provides internal stability for every plant model in that mentioned

set. Stability analysis has been done by gain and phase margin for a nominal plant to measure the size of the plant model's gain and phase perturbation while the closed loop system's internal stability holds. As a result, it can be proven that *a controller provides robust stability if and only if* $\|W_2T\|_\infty < 1$, where T is nominal closed loop transfer function (called complementary sensitivity function) (Doyle *et al.*, 1992).

Given the information, we should consider the performance for a perturbed plant. A robust performance is to have internal stability and a specific performance for every plant in the set of perturbed plants. Having robust stability and nominal performance from the above information ($\|W_2T\|_\infty < 1$ and $\|W_1S\|_\infty < 1$), it can be shown that *a necessary and sufficient condition for robust performance is* $\||W_1S| + |W_2T|\|_\infty < 1$ (Doyle *et al.*, 1992).

Finally, we can design a controller based on the above conditions to achieve robust stability with respect to multiplicative perturbation and nominal performance. Another reasonable condition approximation to nominal performance, and robust stability, which can be interchangeably used is $\|(|W_1S|^2 + |W_2T|^2)^{1/2}\|_\infty < 1$ (Doyle *et al.*, 1992). More details, proofs, designs, and the complete theory of this area can be found in (Doyle *et al.*, 1992).

2.2 Hybrid Systems

A hybrid system refers to a dynamic system which combines continuous and discrete dynamic behaviours. A discrete system switches from one mode to another upon validation of the corresponding guard conditions, while state variables evolve continuously as a function of time. The guard conditions in a discrete event system describe the desired discrete state operation of the system while the differential equations in

continuous system depict the continuos operation of the system.

A continuous system can be modelled by a mathematical equation, which takes an input signal $x(t)$ and produces a result $y(t)$. The stability of a continuous system is defined in terms of the size of its output signal. A continuous system is stable if it produces bounded outputs for all bounded inputs. For example, the following equation is given as a definition for stability at time t:

$$M < \infty, |x(t)| \leq M \tag{2.1}$$

A discrete event system can be conceptualized with a finite state machine (FSM). A finite state machine is a set of reachable and countable states connected by a series of arrows called transitions. A transition is described by a label that consists of some conditions and transition actions. These conditions determine the active mode of operation. If a transition is triggered, its correspondent guard conditions must be valid in order to allow the transition to occur. The continuous states in a typical hybrid system evolve depending on the current discrete mode of operation. Therefore, any discrete mode of operation is paired with its corresponding continuous state variables.

2.3 Matlab and Simulink

Matlab Simulink is a block diagram environment for multi-domain simulation and model-based design which supports a list of libraries for developing different types of

discrete and continuous dynamic systems in different domains. It performs system-level design, simulation, automatic code generation, continuous test and verification[1].

Simulink develops a prototype for conceptualizing a hybrid system using continuous controller block components and Stateflow components. Continuous controller block components represent the continuous behaviour of the hybrid system, while Stateflow components depict the discrete behaviour of the system. This tool is used extensively to design a complete system, and produce the required test results.

2.4 Differential Dynamic Logic ($\mathsf{d}\mathcal{L}$)

Differential dynamic logic ($\mathsf{d}\mathcal{L}$) is a first-order dynamic logic for the specification and verification of hybrid systems. Program notation of hybrid systems, hybrid programs, with symbolic parameters have been used during the verification process of dynamic logic. Free variable sequential composition proof calculus with real arithmetic and quantifier elimination allows deductive verification of hybrid programs (Platzer, 2008, 2010, 2012).

The symbolic parameters of a system are represented by a set of logical variables in first-order logic, while the continuous behaviour of a system is described by dynamic logic. This logic can be used to verify the operation of a system with discrete and continuous state transitions by introducing hybrid programs with discrete assignments and differential actions and then applying a deductive method rather than using abstractions and exhaustive state space exploration as is typically done in model checking approaches (Platzer, 2007). The limited knowledge of $\mathsf{d}\mathcal{L}$ needed to understand this work is summarized below. More details are available in (Platzer,

[1] http://www.mathworks.com

2008, 2010).

Dynamic logic ($\mathsf{d}\mathcal{L}$) consists of nonlinear real arithmetic, real valued quantifiers, and modal operators, such as $<\alpha>$ or $[\alpha]$ for expressing reachable state conditions during system execution, where α presents the continuous evolution of a system. A set of logical variables V, a signature Σ, a set of real valued function and predicate symbols are used to define the well-formed terms and formulas that are given as follows[2]:

$$\theta ::= x \mid f(\theta_1, ..., \theta_n)$$

where $\theta_1, ..., \theta_n$ are terms, f is a function symbol of arity n, and x is a real-valued constant symbol.

$$\phi, \psi ::= p(\theta_1, ..., \theta_n) \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \rightarrow \psi \mid \forall x\phi \mid \exists x\phi$$

where ϕ and ψ are first-order formulas, θ_i are terms, p is a predicate symbol of arity n, and $x \in V$ is a logical variable.

Hybrid programs consist of discrete jump sets, systems of differential equations and a control structure. The discrete transitions assign values to the state variables, and the differential equations are used to express a continuous dynamic evolution of the system, which may change from one discrete state to another. The control structure plays an important role for combining the discrete and continuous transitions using regular expression operators, such as ($\cup$, *, ;). The grammar for designing the hybrid programs is given as follows:

[2]All the materials of this section are from (Platzer, 2010)

$$\alpha, \beta ::= x_1 := \theta_1, ..., x_n := \theta_n \mid x_1' = \theta_1, ..., x_n' = \theta_n \& \chi \mid ?\chi \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*$$

where α and β are hybrid programs, θ_i are terms, $x_i \in \Sigma$ are state variables, and χ is a formula of first-order logic. $x_1 := \theta_1, ..., x_n := \theta_n$ shows a discrete jump, in which θ_i assigns to state variables x_i. $x_1' = \theta_1, ..., x_n' = \theta_n \& \chi$ presents a list of differential equations for describing dynamic behaviour with additional first-order constraints χ. $?\chi$ and $\alpha \cup \beta$ are used to test the state variables and represent nondeterministic choice, respectively. $\alpha; \beta$ and α^* present sequential composition and nondeterministic repetition, respectively. Dynamic logic ($\mathsf{d}\mathcal{L}$) can be used to design other structures by combining the control structure operators ($\cup$, *, ;) with $?\chi$, such as in conditional statements like **if** χ **then** α **else** β, **while** χ **do** α. Formulas of dynamic logic ($\mathsf{d}\mathcal{L}$) based-on first-order logic together with some modal operators ($<\alpha>$ or $[\alpha]$) are defined as follows:

$$\phi, \psi ::= p(\theta_1, ..., \theta_n) \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \rightarrow \psi \mid \forall x \phi \mid \exists x \phi \mid [\alpha]\phi \mid <\alpha> \phi$$

where ϕ, ψ are dynamic logic ($\mathsf{d}\mathcal{L}$) formulas, θ_i are terms, p is a predicate symbol of arity n, $x \in V$ is a logical variable, and α is a hybrid program. The syntax of dynamic logic ($\mathsf{d}\mathcal{L}$) allows real arithmetic predicate expressions, negation, conjunction, disjunction, implication, universal and existential quantification, and modalities to express the validity of formula ϕ for any terminating execution of hybrid program α ($[\alpha]\phi$) or at least one terminating execution of hybrid program α ($<\alpha> \phi$).

2.5 Verification Tool: KeYmaera

KeYmaera (Platzer and Quesel, 2008) is a hybrid verification tool integrated with an automated and an interactive theorem prover to formalize and verify hybrid systems. It supports dynamic logic ($\mathsf{d}\mathcal{L}$), and combines different methods, such as deductive logic, real algebraic and computer algebraic rules. Moreover, KeYmaera also supports nonlinear discrete jumps, nonlinear differential equations, differential-algebraic equations, differential inequalities, and nondeterministic discrete or continuous input for hybrid systems to express the functional behaviours. KeYmaera allows decomposition of hybrid system specifications into symbolic form and into subsystems to simplify the proof strategy. However, a bottom-up approach employing compositional verification allows KeYmaera to verify large, complex systems by proving the required properties of the sub-systems and then the main system.

Chapter 3

A High Level Safety Concept Abstraction for ACC$^+$ Systems

In this chapter, we propose a general solution for preserving collision-freedom of any ACC$^+$ system design. In other words, we want to identify the invariant required to ensure the safe operation of any ACC$^+$ system design, such that it can be formally verified that the provided invariant preserves the safety property of any system with similar dynamics. Therefore, safety of any practical ACC$^+$ controller design can then be verified by showing the compliance of its behaviour with the safety property.

ACC$^+$ systems can be formalized using *differential dynamic logic* ($\mathsf{d}\mathcal{L}$), (Platzer, 2010), to state and prove safety properties and performance requirements by capturing the system constraints together with the desired behaviours and controller designs. We first propose an abstract safety model for such a system in order to meet the required safety property. Then, we will formalize the functionality of the proposed system using $\mathsf{d}\mathcal{L}$ and will prove collision-freedom of the system while the proposed safety property invariant is preserved.

3.1 Safety Model of the ACC$^+$ Systems

A high level conceptual safety design of ACC$^+$ is proposed in this section, where we consider that the host vehicle is equipped with ACC$^+$, and the host vehicle follows a lead vehicle in the same lane. We use a high level conceptual model of the ACC$^+$ system to formalize an abstraction of the system requirements to satisfy the desired safety properties. According to (Loos *et al.*, 2011; Aréchiga *et al.*, 2012), collision-freedom for these kinds of systems can be achieved if and only if there is always a safe distance between two successive vehicles. This distance, which we will denote by sc_{gap}, can be derived from Newton's formula of motion as in Eq. 3.1, where B is the absolute value of maximum deceleration achieved by maximum brake force, and v_l and v_h are lead and host vehicles' velocities respectively.

$$sc_{gap}(v_l, v_h) = \frac{v_h^2 - v_l^2}{2 \times B} \tag{3.1}$$

The length of $sc_{gap}(v_l, v_h)$ should be such that the host vehicle can fully stop at the rear end of the lead vehicle or the end of $sc_{gap}(v_l, v_h)$ in the worst case scenario when the lead vehicle may itself be suddenly using the same maximum brake force to come to a full stop. In the case when the relative distance between the two vehicles is less than or equal to this safe distance, the host vehicle has no choice but to use its maximum braking power to exit the critical zone in order to make the system collision free. This fact is critical to the safe, collision-free operation of any ACC or ACC$^+$ design.

In addition, the system uses sensors to provide required values for the control system; however, there is some lag associated with acquiring sensor reading, the

controller needs some time to react to any new sensor values, and the actuators take some time to react. Therefore, a safety margin should be taken into account related to the maximum delays in the system. This extra padding distance can be determined by the following formula (Eq. 3.2) according to (Loos *et al.*, 2011).

$$margin_{sc_{gap}}(v_h) = (\frac{A_{max}}{B} + 1)(\frac{A_{max}}{2} \times \epsilon^2 + \epsilon \times v_h) \tag{3.2}$$

Here A_{max} is the maximum acceleration of the host vehicle and ϵ is the worst case delay time, which is close to zero. Eq. 3.2 is considered as the worst case scenario where the host vehicle is traveling with maximum acceleration (A_{max}) when the ACC$^+$ system requests the maximum negative acceleration B. The host vehicle will continue to accelerate at A_{max}, increasing its velocity v_h for ϵ seconds before it starts to decelerate at $-B$. Therefore, the extra distance given in Eq. 3.2 is required for acceleration $-B$ to return the host vehicle to what was its initial velocity, v_h, when the negative deceleration was first requested. Consequently, $margin_{sc_{gap}}(v_h)$ is the total of these two distances that the host vehicle travels during the ϵ delay. Thus, the ACC$^+$ system can react safely if the relative distance between host and lead vehicle ($d_{gap} = x_l - x_h$) is always greater than the sum of $sc_{gap}(v_l, v_h)$ and $margin_{sc_{gap}}(v_h)$ as in Eq. 3.3.

$$d_{gap} > sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h) \tag{3.3}$$

Therefore, the High Level Safety Concept Abstraction for ACC$^+$ can be specified as shown in Fig. 3.1. This figure depicts an independent safety system that intervenes only when necessary. This system monitors the relative distance to the lead

vehicle (d_{gap}) and sets the host vehicle acceleration a_h to $-B$ whenever the relative distance is less than or equal to safety distance ($d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)$). Effectively it activates a *Safety_Critical* mode that applies the maximum brake force. Four components have been considered for this purpose: a *guard condition* block, a switching block, *safety_Critical*, and *Other*. The *guard condition* block checks the validity of the safety condition and the switching block changes the active mode from *Other* (normal ACC$^+$) functionalities to *Safety_Critical* in critical cases when $d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)$ holds. A tabular representation of this decision making structure is shown in Table 3.1, where *Other* is used to consider all other behaviour ACC$^+$ could have in different scenarios and *Safety_Critical* is used to apply maximum brake.

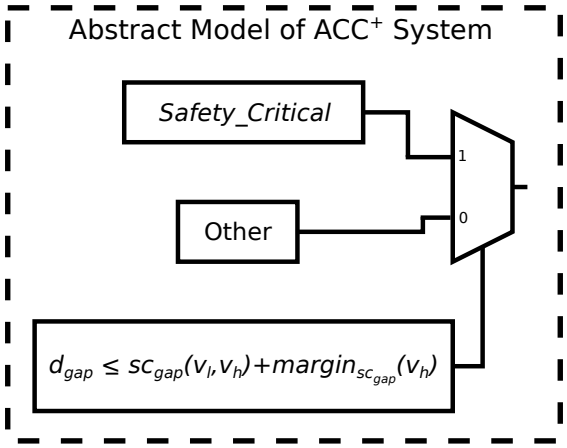


Figure 3.1: High level abstract conceptual design block diagram

$d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)$	$-B$	*Safety_Critical*
$d_{gap} > sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)$	$[-B, A_{max}]$	*Other*

$\boldsymbol{a_h}$	*Mode*

Table 3.1: Decision making structure of abstract ACC$^+$

3.2 Verification

Since some of the system parameters come from the environment or are yet to be determined by a more detailed design, we need to define symbolic constraints for some parameters like vehicles' acceleration. Before doing this, we first define the state variables of the host and lead vehicles that will be used to model their continuous behaviour:

$$host = (x_h, v_h, a_h) \tag{3.4}$$

$$leader = (x_l, v_l, a_l) \tag{3.5}$$

where x_h is position, v_h is velocity, and a_h is acceleration of the host vehicle, and x_l is position, v_l is velocity, and a_l is acceleration of the lead vehicle. These variables can then be used to specify the dynamics of a real-time system, where the relationships between position, velocity and acceleration are $x'_h = v_h$ and $v'_h = a_h$ for the host vehicle, and are $x'_l = v_l$ and $v'_l = a_l$ for the lead vehicle.

The velocity of the host (lead) vehicle changes continuously according to the current acceleration of the host (lead) vehicle. We assume the maximum acceleration for both the host and lead vehicles is $A_{max} > 0$, and similarly the maximum deceleration due to braking with the maximum braking force is $-B$ where $B > 0$. Therefore,

$$-B \leq a_h \leq A_{max} \ \ \& \ \ -B \leq a_l \leq A_{max} \tag{3.6}$$

The complete formalization of our abstract ACC$^+$ is presented in **Model 1**. The model contains both discrete and continuous dynamic behaviours. **Model 1** can be

derived in a similar fashion to (Loos *et al.*, 2011), where they also define an abstract model for an autonomous vehicle. However, **Model 1** presented here is simpler and more abstract than the model in (Loos *et al.*, 2011). The *Local Lane Control* of the ACC system in their work always sets the acceleration of the host vehicle to zero in the case that its velocity is zero. Thus once stopped, the ACC system remains stopped regardless of the behaviour of the lead vehicle. Also, in their work *Local Lane Control* chooses a nondeterministic brake value within a particular range for the safety critical situation, which makes the system more complicated than a safety concept abstraction. Despite its complexity, *Local Lane Control* in (Loos *et al.*, 2011) is more realistic than a basic safety concept abstraction since it might not always be possible to achieve $-B$, for example when the road is wet.

The host and lead vehicles can repeatedly choose an acceleration from the range $[-B, A_{max}]$ in **Model 1**. This behaviour is specified by the nondeterministic repetition $*$ in (1). The host and lead vehicles operate in parallel as defined in (2). The lead vehicle is free to use brake or acceleration at any time; so, a_l is assigned non-deterministically in (3), and the model continues if a_l is within its accepted range $[-B, A_{max}]$.

The host vehicle's movement depends on the distance between the host vehicle and the lead vehicle. The most crucial functionality of ACC$^+$ is formalized as successive actions to capture the decision on entering the safety critical mode as the last action in (4) before the system's continuous state is updated. The safety following distance ($sc_{gap}(v_l, v_h)$) and the extra safety margin for delays ($margin_{sc_{gap}}(v_h)$) are calculated in (5). The last line in (5) assigns the relative distance to d_{gap} . The host vehicle can choose any arbitrary acceleration value in the valid range $-B$ to A_{max} for the *Other*

mode in (6) to capture all dynamic behaviours of possible ACC$^+$ system designs. The safety requirement that the system applies maximum brake force when the host vehicle is within the safe following distance is formalized as the overriding action of the *Safety_Critical* mode in (7). The continuous state of the system then evolves over time which is measured by a clock variable t. The sampling time of the system has been considered as the delay of the system $t \leq \epsilon$ where slope is considered as $t' = 1$. Therefore, system is piecewise continuous and the physical laws for movement are formalized by simplified versions of Newton's formula, are all presented in (8).

Model 1: Formalization of abstract model for ACC$^+$ systems

$$ACC^+ \equiv (Vehicle; Drive)^* \quad (1)$$

$$Vehicle \equiv host \,||\, leader; \quad (2)$$

$$leader \equiv a_l := *; ?(-B \leq a_l \leq A_{max}) \quad (3)$$

$$host \equiv Calc\_sc_{gap};\ Other;\ Safety\_Critical; \quad (4)$$

$$Calc\_sc_{gap} \equiv sc_{gap}(v_l, v_h) := \tfrac{v_h^2 - v_l^2}{2 \times B}; \quad margin_{sc_{gap}}(v_h) := (\tfrac{A_{max}}{B} + 1)(\tfrac{A_{max}}{2} \times \epsilon^2 + \epsilon \times v_h); \quad d_{gap} := x_l - x_h; \quad (5)$$

$$Other \equiv a_h := *; ?(-B \leq a_h \leq A_{max}); \quad (6)$$

$$Safety\_Critical \equiv \text{if } \big(d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)\big) \text{ then } a_h := -B \text{ fi}; \quad (7)$$

$$Drive \equiv t := 0; (x_h' = v_h \wedge v_h' = a_h \wedge x_l' = v_l \wedge v_l' = a_l \wedge t' = 1 \wedge v_h \geq 0 \wedge v_l \geq 0 \wedge t \leq \epsilon) \quad (8)$$

With the system dynamics specified, we can now use the KeYmaera (Platzer and Quesel, 2008) tool to verify the required collision-freedom safety property.

Property 1: *If the host vehicle is following at a safe distance behind the lead vehicle, then the vehicles will never collide in any operation when the host vehicle controllers follow the defined dynamics under given safety constraints.*

In KeYmaera this property will take the form:

$$\text{Controllability Condition} \rightarrow [\text{Abstract ACC}^+] \;\; x_h < x_l \tag{3.7}$$

The controllability condition will be given below in equation (3.8). We now explain how we arrive at the appropriate precondition for the safety property. To complete (3.7), we must establish a precondition that says that the host vehicle is behind the lead vehicle and both vehicles are moving in a forward direction. The relation (3.7) indicates that for all iterations of the hybrid program in **Model 1** the position of the host vehicle is always less than the lead vehicle's position ($x_h < x_l$) if the given controllability condition is satisfied. In other words, relative distance between the vehicles is always greater than zero ($d_{gap} > 0$) if the precondition holds. One of the most important condition is the safe distance formula, which is an invariant during the proof of this hybrid program. This condition can be considered as a controllability property and must be always satisfied by every operation of the ACC$^+$ system.

3.2.1 Controllability

The controllability formula states that for every possible evolution of the ACC$^+$ system, it can satisfy the safety property by applying maximum brake before it has passed the *Safety_Critical* distance. The vehicle is *controllable* if there is enough distance in order to fully stop the car by the rear end of lead vehicle or exit the critical zone. The assumption is that both vehicles only move forward (i.e. their velocity

is greater than or equal to zero). Therefore, the ACC$^+$ will be safe if it can satisfy condition (3.8), which is an invariant for the defined system dynamics of **Model 1**. This controllability property in condition (3.8) is a safety concept invariant not only for ACC$^+$ systems, but also for any kind of system with similar continuous motion dynamics.

$$x_l > x_h \wedge v_h^2 - v_l^2 < 2 \times B \times d_{gap} \wedge v_l \geq 0 \wedge v_h \geq 0 \tag{3.8}$$

An important fact in this verification is that there must be a required distance to be physically possible to stop the host vehicle by the rear end of an instantaneous obstacle. This has been formally presented in (3.8) as $v_h^2 - v_l^2 < 2 \times B \times d_{gap}$. The system checks whether it can satisfy the safety property in case of detecting any obstacle and once it gets in to the critical zone it uses maximum brake until the safety property holds again.

This model has been written in the KeYmaera theorem prover (Platzer and Quesel, 2008) and the required safety property (3.7) has been successfully proven. In this abstract model of ACC$^+$ system, we considered a viable range of accelerations for the host vehicle that admits a variety of desired behaviour for a concrete ACC$^+$ system in different scenarios.

3.3 Summary

The focus of this chapter was on demonstrating the desired behaviour of any ACC$^+$ concrete model in the safety critical case that is required to guarantee the safety requirement of collision freedom. Therefore, the required safe, collision free, distance between two successive vehicles was derived. A general controllability invariant was

also given along with the formalized abstract model of ACC$^+$ using *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) (Platzer, 2010). Finally, the general abstract model of ACC$^+$ system was proved to be collision-free preserving the required collision-freedom safety property.

In chapter 4, this system will be refined with respect to other requirements to create a more realistic concrete ACC$^+$ design, whose safety has already been proven if it can be shown that the new ACC$^+$ design refines this abstract ACC$^+$ safety concept.

Chapter 4

Refinement of Abstraction into a Practical ACC$^+$ Model

In this chapter, we aim to refine the abstract ACC$^+$ model described in Chapter 3 into a practical ACC$^+$ system. By considering some assumptions and requirements in Section 4.1, we will design our ACC$^+$ system with a hierarchy structure. In Section 4.2, different modes of operation will be defined to capture corresponding scenarios that the vehicle might undergo. A mode switching system will be designed to capture these scenarios in Section 4.3, while Section 4.4 will present the design of low level continuous controllers. Section 4.5 will demonstrate the simulation results of a test case to further evaluate the performance of the proposed ACC$^+$ system. At the end, we will provide a formalization of our ACC$^+$ system in Section 4.6 for the purpose of proving safety. Finally, in Section 4.7, we will further investigate the refinement relation between our abstract model defined in Chapter 3 and the practical ACC$^+$ system. This chapter will end with a summary in Section 4.8.

4.1 Assumptions & Requirements

An ACC$^+$ design requires information about the host vehicle's continuous state (velocity, acceleration, etc.), as well as information about the presence and behaviour of the lead vehicle. While the most important requirement for ACC$^+$ systems is to safely adjust the host vehicle's speed in the presence of a lead vehicle, some additional functional requirements and assumptions have to be considered in their design. Assumptions can help to make the design more reliable and practical. Also, understanding additional functional requirements can allow us to scope the design and verification effort.

Assumptions:

1. The ACC$^+$ system will never be operating when the vehicle is moving backwards ($velocity < 0$).

2. The driver is responsible for steering the host vehicle in a safe manner.

3. It is assumed that the maximum range of the sensors for detecting objects in front of the host vehicle is always greater than the safety gap obtained in the Section 3.1 ($d_{range} > sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)$).

4. Errors will be detected by a separate subsystem, a Fault Detection System, that will alert the driver to intervene in the case of a fault.

Given Requirements:

1. The user has the ability to override the ACC$^+$ system settings such as desired

velocity v_{set} and desired headway h_{set}, at any point in the system's operation except in safety critical cases.

2. The accessible parameters of the ACC$^+$ system, such as desired velocity v_{set} and desired headway h_{set}, should be restricted to an acceptable range in order to meet the assumptions and limitations of the design.

3. The ACC$^+$ system must regulate the velocity of the host vehicle to maintain the user's expected velocity in the absence of a slower lead vehicle.

4. The ACC$^+$ system must slow down the host vehicle's velocity and maintain the desired headway when approaching a slower lead vehicle.

5. The acceleration of the system must be restricted to a comfortable range. Therefore, rapid de-acceleration should not be applied during the normal operation of ACC$^+$ system.

6. The ACC$^+$ system should return the operation of the vehicle to the user in the presence of any failure in the system or when throttle/brake is pressed.

Among all these requirements, we consider the implementation of the first to fifth one in our design. The third and fourth requirements, which are not typically discussed in related work such as (Loos *et al.*, 2011, 2013), play a major role in our ACC$^+$ design. The restriction on v_{set}, as described by the second requirement, is derived in Sections 4.2 and 4.3. The required restriction on h_{set} can be derived in a similar fashion to v_{set}. The sixth requirement is not directly addressed in our work. It can be designed in a separate block by using fault diagnosis techniques as in (Mohammadi, 2009). The ACC$^+$ system controls the speed of the host vehicle according to the

different scenarios that are considered during the high level design. Fig. 4.1 depicts a high level design of the ACC$^+$ that contains four components: a *low-level (continuous) controller*, an extended *finite state machine* (FSM), a *sensor*, and the *host vehicle*. We consider the fifth component, the *lead vehicle*, as being external to the ACC$^+$ system. All the components of the ACC$^+$ system are connected by arrows that represent the system data flow. Thus this block diagram shows the flow of information that is required to design the ACC$^+$ system, providing the relationship between the ACC$^+$ subsystems and the lead vehicle. *Mode* is the value of the current state of the FSM that is used by the low-level controller to select a particular continuous controller. The value of *Mode* belongs to the set {*Cruise*, *Follow*, *Safety_Critical*}. Signal v_{ref} is a reference signal for the target velocity for the continuous controller selected inside the low-level controller. A list of the other symbols for describing vehicle behaviour is given in Table 4.1.

Term	**Definition**
Specification Terms	
v_h	velocity of the host vehicle
v_l	velocity of the lead vehicle
a_h	acceleration of host vehicle
a_l	acceleration of lead vehicle
d_{gap}	relative distance between the host vehicle and lead vehicle
Controller Terms	
v_{set}	desired velocity of the host vehicle
B	absolute value of deceleration achieved by maximum brake force which depends upon the current vehicle weight and road conditions
h_{set}	desired following time gap between two successive vehicle (headway)
ϵ	maximum response delay from any actuators (ie. engine, brake etc.)
$f_{gap}(v_l, v_h, a_h)$	the distance it takes for the host vehicle to match the lead vehicle's velocity and be following at the desired headway h_{set} using acceleration a_h
$sc_{gap}(v_l, v_h)$	the distance at which ACC$^+$ system switches into safety critical mode

Table 4.1: Terms used in ACC$^+$ Specification and Controller Design

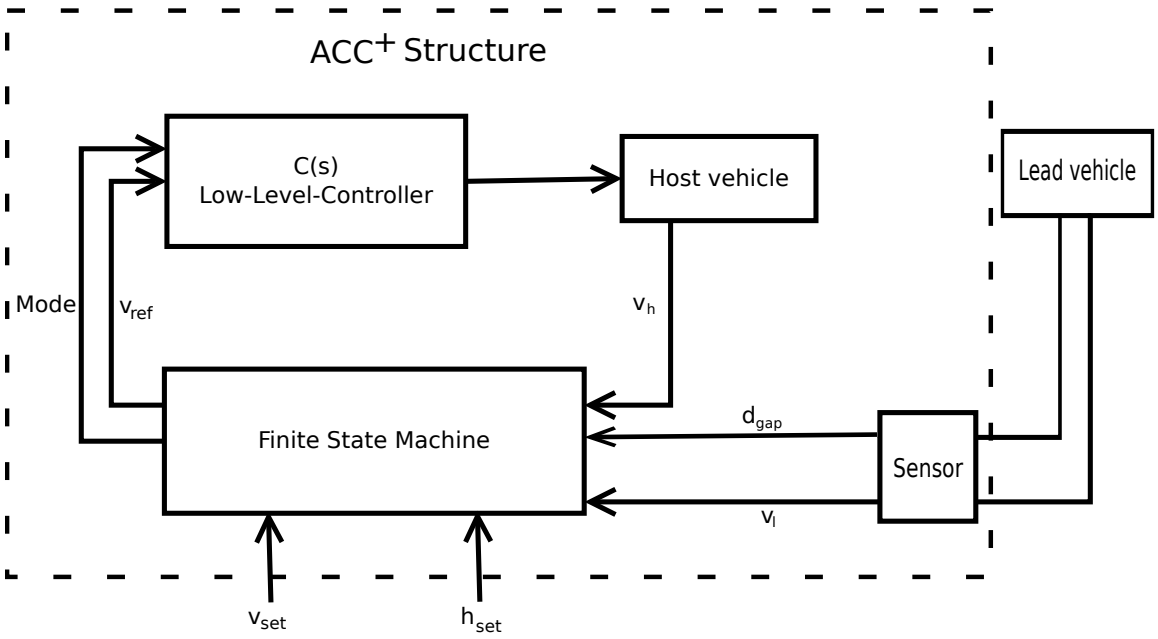


Figure 4.1: High level conceptual design block diagram

4.2 Controller Modes

There are three main operational modes of ACC$^+$ (see Fig.4.4). These modes are:

Cruise which implements standard cruise control system (CC) when no lead vehicle is detected or the lead vehicle exceeds the desired maximum velocity of the host vehicle (v_{set}),

Follow which tries to match the lead vehicle's velocity at distance $h_{set} \times v_l$, and

Safety_Critical where the vehicle has to apply maximum braking force to avoid a collision as discussed in Section 3.1.

Fig. 4.2 and Fig. 4.3 show the headway diagrams describing the possible scenarios. The first mode is similar to a conventional cruise control system (CC) that regulates the speed of the host vehicle to the desired set point (v_{set}) within acceleration limits

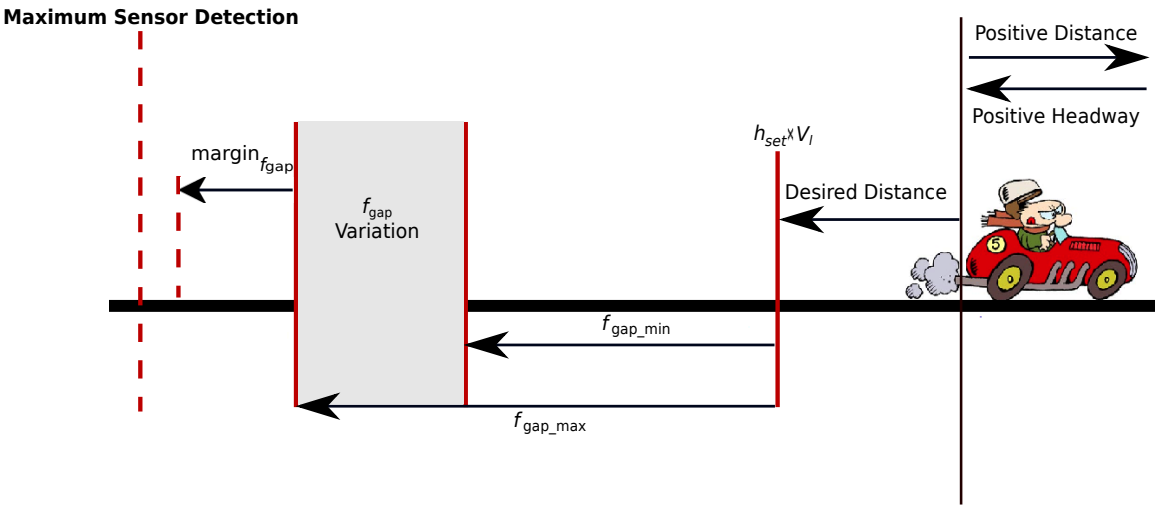


Figure 4.2: Required distance for approaching the lead vehicle in *Follow* mode

based on the requirements such as comfort and fuel efficiency. If the host vehicle detects a leader or an other object, the system determines whether or not the sensed object is going faster than v_{set}. If the lead vehicle is travelling faster than v_{set} and is outside of the safety critical zone ($d_{gap} > sc_{gap}(v_l, v_h)$) then the ACC$^+$ system will not change its operating mode.

The second mode, *Follow*, becomes active when the host vehicle follows a slower lead vehicle outside the safety critical zone. In this situation, the objective is to maintain the desired headway gap $h_{set} \times v_l$ while various aspects such as driver comfort, fuel economy, *etc.*, are considered. When a slower lead vehicle is present, the goal is to reduce the host vehicle's velocity as it approaches the lead vehicle, matching the lead vehicle's velocity when the gap closes to the desired headway $h_{set} \times v_l$. To achieve this behaviour the system picks a negative acceleration for the host vehicle ($a_h < 0$). An additional restriction for the host vehicle's acceleration a_h is the maximum available deceleration B available by applying full brake force, i.e. $a_h \geq -B$. For a chosen value of a_h in this range, the distance required to reduce the host vehicle's velocity to match

the lead vehicle's velocity at the desired following distance $h_{set} \times v_l$ is $f_{gap}(v_l, v_h, a_h)$. The size of $f_{gap}(v_l, v_h, a_h)$ is derived from Newton's formula of motion as follows:

$$f_{gap}(v_l, v_h, a_h) = \frac{v_h^2 - v_l^2}{-2 \times a_h}, \qquad (-B \leq a_h < 0) \tag{4.1}$$

Eq. 4.1 describes the distance required for the host vehicle to achieve v_l as its new velocity, where $-B \leq a_h < 0$ is the deceleration of the host vehicle. According to Eq. 4.1, if the host vehicle wants to use a negative, constant acceleration a_h to achieve the leader's velocity by the time it reaches distance $h_{set} \times v_l$, it has to start decelerating at distance $f_{gap}(v_l, h_v, a_h)$. Note that once the host vehicle achieves the leader's velocity the size of $f_{gap}(v_l, v_h, a_h)$ will become zero (Eq. 4.1).

Based on the constraints on acceleration and the maximum range of the distance sensor, there are constraints on possible values for f_{gap} to a value between f_{gap_min} and f_{gap_max}. As shown in Fig. 4.2, the system bounds on $f_{gap}(v_l, v_h, a_h)$ by restricting the values of a_h that the ACC$^+$ system will use. The upper bound f_{gap_max} and lower bound f_{gap_min} will be defined based-on the upper and lower bound of $a_h < 0$. An upper bound of a_h is the minimum deceleration that the ACC$^+$ system will use by, for example, easing up on the throttle at the current vehicle velocity, while a lower bound is achieved by the maximum braking deceleration B the vehicle can generate based on the current vehicle weight and road conditions (i.e., $a_h \geq -B$).

To make the system more realistic, another safety margin has been taken into account related to the system delay ϵ that is required to respond to messages from the ACC$^+$ system to the engine and brake controllers and the time they require to activate their respective actuators and have them respond. This margin can be determined by the following formula:

$$margin_{f_{gap}}(v_h, a_h) = (\frac{A_{max}}{-a_h} + 1)(\frac{A_{max}}{2} \times \epsilon^2 + \epsilon \times v_h) \tag{4.2}$$

The size of this margin for the response delay (Eq. 4.2) can be derived in similar fashion to the derivation of $margin_{sc_{gap}}(v_h)$ (Eq. 3.2) by replacing the maximum braking deceleration B with the deceleration a_h. The value of a_h is considered to be negative in all of the formulae given when approaching the lead vehicle, assuming a slower lead vehicle. Once the host vehicle reaches the leader's velocity, it will attempt to track the lead vehicle's velocity and those formulae are not required anymore. Finally, the *Follow* mode will be activated if relative distance between the vehicles, d_{gap}, is less than or equal to $f_{gap}(v_l, v_h, a_h) + margin_{f_{gap}}(v_h, a_h) + (h_{set} \times v_l)$ but greater than the safety critical distance. Note that the value of $f_{gap}(v_l, v_h, a_h)$ becomes negative in the case when the leader's velocity is greater than the host vehicle's velocity ($v_l > v_h$). Therefore, the system always chooses the $\max(f_{gap}(v_l, v_h, a_h), 0)$ for system safety. Although $h_{set} \times v_l$ converges to zero as v_l goes to zero, $margin_{f_{gap}}(v_h, a_h) > 0$ ensures a minimum following distance.

$$d_{gap} \leq max(f_{gap}(v_l, v_h, a_h), 0) + margin_{f_{gap}}(v_h, a_h) + (h_{set} \times v_l) \tag{4.3}$$

The velocity of the host vehicle should be in a range such that the right side of Eq. 4.3 is within the maximum range of the *Sensor*, as shown in Fig. 4.2. Assume the *Sensor* component of the ACC$^+$ system shown in Fig. 4.1 which measures the velocity and position of the lead vehicle relative to the host vehicle has a maximum range of d_{range} meters. Then the maximum v_{set} that can be employed by the system assuming a realistically comfortable deceleration a_h as 30% of maximum deceleration

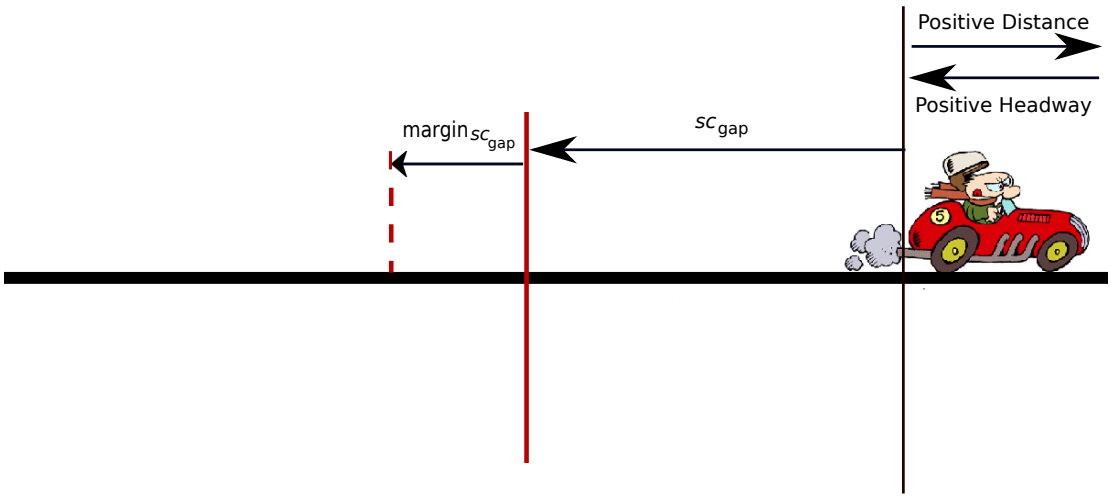


Figure 4.3: Minimum required distance in *Safety_Critical* mode

B ($a_h = -0.3B$), time delay ϵ in the response of the system, and a worst case zero velocity of lead vehicle ($v_l = 0$) results in the following equation:

$$v_{set} \leq \sqrt{2 \times 0.3B \times (d_{range} - margin_{f_{gap}}(v_{set}, -0.3B))} \tag{4.4}$$

Note that Eq. 4.4 represents an approximation of v_{set} because $margin_{f_{gap}}(v_h, a_h)$ has been considered to be a fixed value.

The third mode is the *Safety_Critical* mode that activates when a vehicle suddenly cuts in the lane or an obstacle appears in front of the vehicle, and the relative distance is less than or equal to the minimum stopping distance for the vehicle when full braking power is applied. In this case the the host vehicle has no choice but to use its maximum braking power to exit the critical zone where $d_{gap} \leq sc_{gap}(v_l, v_h)$ (see Fig. 4.3). Although this situation should not normally occur when the host vehicle is in the follow state, it may happen in critical situations such as a cut-in scenario. Fig. 4.3 illustrates $sc_{gap}(v_l, v_h)$. The size of this zone and the margin for the response

delay has been derived in Eq. 3.1 and Eq. 3.2 of Section 3.1. According to the discussion in Section 3.1, this mode implies safety and collision-freedom of any ACC$^+$ system. Note that in this scenario, we consider same maximum braking deceleration for both the host and the lead vehicles.

4.3 Mode Switching

It may, in fact, be the case that the lead and host vehicles have different values for the maximum brake deceleration; hence, the equation for the distance $sc_{gap}(v_l, v_h)$ (Eq. 3.1) may be changed accordingly. Let us assume that the maximum negative acceleration due to maximum brake force for host and lead vehicles are B and b, respectively, then the value of $sc_{gap}(v_l, v_h)$ becomes:

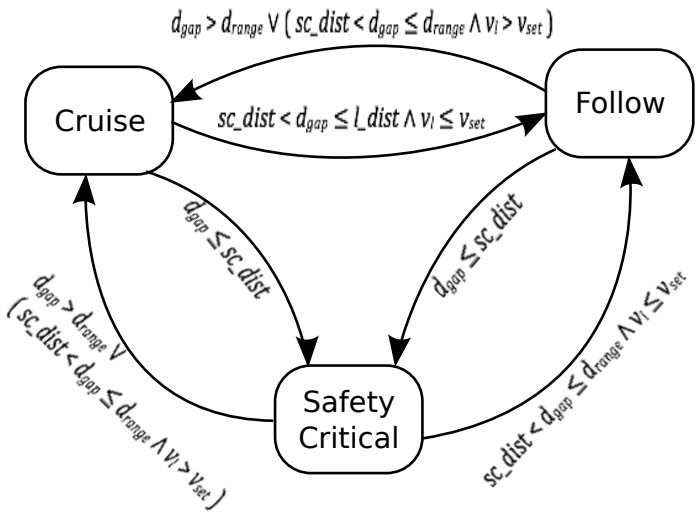
$$sc_{gap}(v_l, v_h) = \frac{v_h^2}{2 \times B} - \frac{v_l^2}{2 \times b} \tag{4.5}$$

The procedure for deriving this version of sc_{gap} is trivial. The ACC$^+$ system switches to *Safety_Critical* mode when the relative distance becomes less than or equal to the value of $sc_{gap}(v_l, v_h)$ as formalized in Eq. 4.6.

$$d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h) \tag{4.6}$$

According to Eq. 3.1 in Section 3.1, the safety critical gap shown in Fig.4.3 converges to zero when the host vehicle attains the same velocity as lead vehicle (i.e., $sc_{gap}(v_l, v_h) \rightarrow 0$). This is under the assumption that both vehicles have the same maximum braking deceleration. In the case when the maximum braking deceleration

differs (Eq. 4.5), $sc_{gap}(v_l, v_h) > 0$ when $v_h = v_l$ and $B < b$, providing the extra required braking margin due to the lesser maximum deceleration of the host vehicle. In the case when $B > b$, $sc_{gap}(v_l, v_h) < 0$ when $v_h = v_l$ so we take the maximum of $sc_{gap}(v_l, v_h)$ and 0.



where

$$l\_dist := max(f_{gap}(v_l, v_h, a_h), 0) + margin_{f_{gap}}(v_h, a_h) + (h_{set} \times v_l)$$
$$sc\_dist := max(sc_{gap}(v_l, v_h), 0) + margin_{sc_{gap}}(v_h)$$

Figure 4.4: Finite State Machine

Fig. 4.4 and Table 4.2 are alternative representations of the high level design of our ACC$^+$ system. Fig. 4.4 shows the finite state machine (FSM) with the three major modes as separate states. Guard conditions are attached to transitions in this figure. The tabular representation of transition from one state to another is given in Table 4.2.

The actual value of deceleration applied in the *Follow* state could be chosen to be comfortable for the user while achieving a high level of fuel economy, traffic flow, and safety. This deceleration can be described as minimizing a_h. The point that the

Let
$l\_dist := max(f_{gap}(v_l, v_h, a_h), 0) + margin_{f_{gap}}(v_h, a_h) + (h_{set} \times v_l)$
$sc\_dist := max(sc_{gap}(v_l, v_h), 0) + margin_{sc_{gap}}(v_h)$ in

					Mode
$d_{gap} \leq d_{range}$	$d_{gap} \leq sc\_dist$				*Safety_Critical*
	$d_{gap} > sc\_dist$	$v_l > v_{set}$			*Cruise*
		$v_l \leq v_{set}$	$d_{gap} \leq l\_dist$		*Follow*
			$d_{gap} > l\_dist$	$Mode_{-1} \neq Cruise$	*Follow*
				$Mode_{-1} = Cruise$	*Cruise*
$d_{gap} > d_{range}$					*Cruise*

Table 4.2: Decision making structure of ACC$^+$

vehicle switches to the *Follow* mode, can be determined by an optimization process selecting a_h and then desired velocity reference will be computed using Eq. 4.7 and sent from the FSM to the low-level controller.

$$v_{ref} = \sqrt{v_l^2 - 2 \times a_h \times (d_{gap} - v_l \times h_{set})} \tag{4.7}$$

Eq. 4.7 can be derived based on the Eq. 4.3 for the normal following action where $-B \leq a_h < 0$. This velocity reference signal is defined for the case that host vehicle detects a slower lead vehicle and the ACC$^+$ system needs to decrease the velocity such that the host vehicle can achieve leader's velocity by the desired headway. The term under the square root in Eq. 4.7 will not be negative as long as the velocities are greater than or equal to zero. In Eq. 4.7 $d_{gap} - v_l \times h_{set}$ represents the distance it takes for the host vehicle to achieve the leader's velocity. If this term is less than zero ($V_l^2 - 2 \times a_h \times (d_{gap} - v_l \times h_{set}) < 0$), it means that the host vehicle should move backward which is in contradiction with the first assumption of ACC$^+$ system. Therefore, system always picks a maximum value between this term ($V_l^2 - 2 \times a_h \times (d_{gap} - v_l \times h_{set})$) and zero (i.e., $max(V_l^2 - 2 \times a_h \times (d_{gap} - v_l \times h_{set}), 0)$).

In a cut-in scenario when the velocity of the lead vehicle (v_l) and relative distance (d_{gap}) change abruptly from one set of specific values to another, there is no continuous trajectory for the parameters of the system. It is possible that the *Mode* of the system was *Follow* before the change and remains in *Follow* after updating the sensor values for the new leader. For example, the host vehicle is decreasing the velocity of the host in the *Follow* mode to achieve a leader's velocity by the desired headway when suddenly another leader vehicle l_{new} with velocity less than v_{set}, i.e. $v_{l_{new}} < v_{set}$, cuts in the lane. However, the host vehicle's velocity is less than this new leader's velocity ($v_h < v_{l_{new}}$) and the relative distance between the host vehicle and new leader is not less than safety critical distance (i.e., $d_{gap_{new}} > max(sc_{gap}(v_l, v_h), 0) + margin_{sc_{gap}}(v_h)$). In this example, the ACC$^+$ system will not change the mode of operation and will remain in *Follow*. Therefore, the ACC$^+$ system may accelerate to match the new leader's velocity by the desired headway. After updating different parameters such as v_l, and d_{gap}, the range of a_h for this purpose can be between 0 to A_{max}. Finally, Eq. 4.7 can be used as v_{ref} with $0 \leq a_h \leq A_{max}$. The optimization process should find a valid value for a_h from this range based on the relative distance, the distance takes for the host vehicle to achieve the leader's velocity, and the desired headway. However, it should be bounded so that the system will not thrash between *Follow* and *Safety_Critical*. This is a subset of the behaviour we are formally verifying. Although, test cases do not reveal any thrashing in this scenario, further analysis and formal verification is necessary to ensure that the system is free of thrashing. This formal analysis is left to future work that could be done using techniques such as those of (Tabuada, 2009).

In the *Safety_Critical* state the desired velocity v_{ref} is set to zero by the FSM.

The goal is to continuously apply the maximum brake force to get the host vehicle travel out of this critical zone. Therefore, the maximum brake command is passed to the low-level controller. In this case, the reference signal will be zero velocity with the *Safety_Critical* mode signal from the FSM to the Low-Level Controller being interpreted as "apply the maximum brake power and close the throttle".

Mode	v_{ref}
Cruise	v_{set}
Follow	$\sqrt{max(V_l^2 - 2 \times a_h \times (d_{gap} - v_l \times h_{set}), 0)}$
Safety_Critical	0

Table 4.3: Velocity reference signal with respect to the state

Typically, it is important in hybrid systems design to avoid rapid mode switching. In the ACC$^+$ system designed, mode switching between states could cause rapid de-acceleration which is not comfortable for passengers. Thus, in Table 4.2, we avoid rapid mode switching by using hysteresis. When $v_l \leq v_{set}$ and $d_{gap} > l\_dist$, the table checks the previous value of the FSM state, denoted $Mode_{-1}$. The system remains in *Cruise* if the previous value of *Mode* is *Cruise* ($Mode_{-1} = Cruise$), otherwise ($Mode_{-1} \neq Cruise$) it remains in *Follow* or switches from *Safety_Critical* to *Follow*. This behaviour is similarly defined in the finite state machine Fig. 4.4. Once the current state becomes *Follow* or *Safety_critical*, the FSM will switch to *Cruise* only in the case that leader is traveling faster than v_{set} or in the absence of a lead vehicle or object ($d_{gap} > d_{range} \vee (sc\_dist < d_{gap} \leq d_{range} \wedge v_l > v_{set})$). Consequently, the

FSM changes state from *Cruise* to *Follow* only in the case that $d_{gap} \leq l\_dist$. The only other potential source of mode thrashing is between *Follow* and *Safety_Critical*. According to Table 4.3, the reference velocity signal in *Follow* mode is defined as in Eq. 4.7. This v_{ref} ensures that in a typical following of a slower leader, the FSM does not switch back and forth between *Follow* and *Safety_Critical*. In the case that a leader vehicle cuts in the lane, once the host vehicle exits the critical distance, the ACC$^+$ system switches from *Safety_Critical* to *Follow* through the guard condition $sc\_dist < d_{gap} \leq d_{range} \wedge v_l \leq v_{set}$ (Fig. 4.4) and the system does not fall back in to *Safety_Critical* due to the definition of v_{ref} in *Follow* mode (Eq. 4.7). Finally, we can provide v_{ref} for the continuous controller based upon the *Mode* of operation (Fig. 4.1). Table 4.3 defines the value of v_{ref} for each *Mode*.

The desired objective in this mode switching is to avoid sudden application of full brake with the resulting severe jerk in non-critical scenarios. The system should not switch to *Safety_Critical* mode unless a leader vehicle cuts in the lane and there is not enough distance between the host and lead vehicle. There is a particular circumstance under which the mentioned desired objective may be violated during the "normal" functioning of our ACC$^+$ system. This scenario happens when the host vehicle is traveling with reference velocity v_{set} and the ACC$^+$ system's current *Mode* is *Cruise*. If there is a slower lead vehicle in the lane (i.e., $v_l < v_{set}$), but the ACC$^+$ system has not yet changed its *Mode* to *Follow*, we expect the system to switch from *Cruise* to *Follow* when $d_{gap} \leq l\_dist$. However, if the host vehicle's driver decides to change the value of v_{set} to a new value that is less than v_l (i.e., $v_{set_{new}} < v_l$), then according to Fig. 4.4 and Table 4.2, the ACC$^+$ system will not switch the *Mode* from *Cruise* to *Follow* after $d_{gap} \leq l\_dist$. Therefore, the ACC$^+$ system will try to maintain the

new desired velocity $v_{set_{new}}$ in *Cruise* mode. In this situation, for a fixed acceleration $a_h < 0$, the required distance for the host vehicle to slow to $v_{set_{new}}$ can be obtained from the following formula:

$$dist_v = \frac{v_h^2 - v_{set_{new}}^2}{-2 \times a_h}, \qquad (-B \leq a_h < 0) \tag{4.8}$$

If this required distance is greater than or equal to the difference between d_{gap} and the safety critical distance $sc_{gap}(v_l, v_h)$, the system will eventually transition directly from *Mode Cruise* to *Safety_Critical*. Therefore, some additional functionality should be defined in *Cruise* to avoid this undesired behaviour. The system should restrict the driver to choosing the set point velocity from a range of values which do not lead to a full brake in *Safety_Critical* mode. This range of values can be derived from the above explanation, and is formulated in Eq. 4.9.

$$d_{gap} - sc_{gap}(v_l, v_h) > dist_v \tag{4.9}$$

Note that in Eq. 4.9, $margin_{sc_{gap}}(v_h)$ is not considered for simplicity. A lower bound for $v_{set_{new}}$ can be calculated by replacing $sc_{gap}(v_l, v_h)$ and $dist_v$ in Eq. 4.9 with their formulas. Eq. 4.10 demonstrates the lower bound of v_{set} in *Cruise* mode when the velocity of the leader vehicle (v_l) is lower than the initial set point velocity v_{set} and the driver decides to change v_{set} to a value lower than v_l.

$$v_{set_{new}} > \sqrt{v_h^2(\frac{B - a_h}{B}) + 2a_h(d_{gap} + \frac{v_l^2}{2B})} \tag{4.10}$$

This lower bound for v_{set} in Eq. 4.10 is only for avoiding *Safety_Critical* mode in normal operation of the ACC$^+$ system. Although the continuous controller in *Cruise*

mode manipulates the throttle to decrease or increase the velocity, some percentage of brake can be added to the control action in *Cruise* mode. Therefore, a_h can be picked, for instance, as 10% of the maximum deceleration achieved by full brake B ($a_h = -0.1B$). Eq. 4.11 depicts the lower bound for v_{set} by considering 10% of B.

$$v_{set_{new}} > \sqrt{1.1v_h^2 - 0.1v_l^2 - 0.2 \times B \times d_{gap}} \tag{4.11}$$

Note that, if the term under the square root in Eq. 4.10 or Eq. 4.11 becomes negative, it means that $v_{set_{new}}$ can be any value greater than zero ($v_{set_{new}} \geq 0$). Therefore, the lower bound, derived in Eq. 4.11, can be defined by the maximum function in Eq. 4.12.

$$v_{set_{new}} > \sqrt{max(1.1v_h^2 - 0.1v_l^2 - 0.2 \times B \times d_{gap}, 0)} \tag{4.12}$$

As a conclusion, *Cruise* mode operation should be refined based on the following conditions:

No leader / Faster leader: v_{set} can be defined in the interval from zero to the upper limit in Eq. 4.4.

Slower leader: v_{set} can be defined in the interval from the lower limit in Eq. 4.12 to the upper limit in Eq. 4.4.

The implementation of this conditioning operation in *Cruise* mode has been left for future work.

4.4 Continuous Controller Design

Other than the case when we are in *Safety_Critical* mode, the Low-Level (continuous) controller can implement a standard continuous feedback controller designed to meet tracking and disturbance rejection performance requirements. In *Cruise* mode we can use a simple Single Input Single Output (SISO) controller to try to have v_h tracking v_{ref}. In the case when we are in *Follow* mode, we have to use a Multiple Input Multiple Output (MIMO) controller in order to have v_h tracking the v_l at a distance of d_{gap}. A typical performance specification of a control system is "good tracking" of the reference signal(s). This is usually interpreted as asymptotic tracking of a single step or ramp reference signal and is commonly met with a standard design such as a PID controller. However, PID controllers typically do not maintain reasonable performance in the presence of uncertainties in the plant model and set of reference signals. Therefore, robust controller design techniques have been developed to achieve performance in terms of a weighted norm bound that result in strong performance in the presence of plant uncertainties, such as weight of the loaded vehicle, friction of the road, *etc.* Among all possible structured and unstructured uncertainties, we choose simple disk-like multiplicative uncertainty to simplify our analysis. We then design our Low-Level Controller based on the Loopshaping analysis technique of (Doyle *et al.*, 1992). As a result, our ACC$^+$ system attains reliable performance in the presence of plant uncertainty for a variety of reference signals (Table 4.3).

Let us briefly explain the robust feedback control analysis technique of (Doyle *et al.*, 1992). The nominal transfer function of the Remote Control (RC) car motor, with throttle signal as its input and velocity as its output, is derived in (Breimer,

2013). This model is represented in Eq. 4.13.

$$P(s) = \frac{\frac{818.4}{30}.s + \frac{22158}{30}}{s^2 + 93.24.s + 1467} \tag{4.13}$$

Sensitivity (S) and complementary sensitivity (T) functions are used in deriving the robust stability and performance of the closed-loop system. The sensitivity function can be determined as the transfer function between input and error, while the complementary sensitivity function is determined as the closed-loop transfer function between input and output. By considering W_1 and W_2 as the weighting functions, (Doyle *et al.*, 1992) have defined the robust stability and performance.
Robust stability:

$$\| W_2.T \|_\infty < 1 \tag{4.14}$$

Nominal performance:

$$\| W_1.S \|_\infty < 1 \tag{4.15}$$

A necessary and sufficient condition for robust performance is:

$$\| |W_1.S| + |W_2.T| \|_\infty < 1 \tag{4.16}$$

While a reasonable condition approximation to nominal performance, and robust stability is:

$$\| (|W_1.S|^2 + |W_2.T|^2)^{\frac{1}{2}} \|_\infty < 1 \tag{4.17}$$

The RC car model (Eq. 4.13) is stable and has no zeros at the right half plane. According to (Doyle *et al.*, 1992), the set of stabilizing controllers for a stable plant

without any zeros at the right half plane can be defined by Eq. 4.18:

$$C = \frac{Q}{1 - P.Q} \quad \& \quad W_1.S = W_1(1 - P.Q) \tag{4.18}$$

where P is the plant transfer function as in Eq. 4.13. Therefore, we should define Q such that it is stable and proper to obtain the required nominal performance (Eq. 4.15). According to (Doyle *et al.*, 1992), the best definition for Q is:

$$Q = P^{-1}.J \quad \& \quad J(s) = \frac{1}{(\tau.s + 1)^k} \tag{4.19}$$

For a large k and a small τ, Q will be proper and the ∞-norm of the nominal performance (Eq. 4.15) will be less than one. Since the model is in discrete time (Z domain), we need to consider *Zero Order Hold* (ZOH) as a cascade to the plant transfer function $P(s)$ (Eq. 4.13). Hence:

$$P(s) = P(s) \times \frac{T_s}{1 + \frac{T_s.s}{2}} \tag{4.20}$$

In Eq. 4.20, T_s is the sampling time of the system. Since the RC car has been modeled with 3 milliseconds, T_s is equal to this value ($T_s = 0.003$) (Breimer, 2013). The relative degree of $P(s)$ is 2 as shown in Eq. 4.20. The degree of $J(s)$ is derived based on the relative degree of $P(s)$ as shown in Eq. 4.21.

$$J(s) = \frac{1}{(\tau.s + 1)^2} \tag{4.21}$$

By computing some iterations of the nominal performance, we can find the value of τ which satisfies Eq. 4.22. Table 4.4 shows the computed ∞-norm of Eq. 4.22 for

decreasing τ.

$$\| W_1.(1 - J(s)) \|_\infty < 1 \tag{4.22}$$

τ	∞ - *Norm*
0.01705	0.9994
0.01704	0.9988
0.016	0.9378
0.01	0.8554
...	...
0.0001	0.4954

Table 4.4: Nominal Performance

Although $\tau = 0.0001$ reasonably satisfies the condition defined by Eq. 4.22 (e.g. 0.4954), this value leads to an unstable system as it violates the relation defined in Eq. 4.14. Among all possible values, $\tau = 0.01$ provides the best accuracy. Finally, the continuous transfer function of the controller is:

$$C(s) = \frac{P(s)^{-1}.J(s)}{1 - J(s)} \tag{4.23}$$

By using Zero-Pole matching technique, a good approximation of this controller can be derived in discrete time domain.

4.5 Simulation Result

In this section, a test case is presented in Fig. 4.5 to evaluate the behaviour of the proposed ACC$^+$ system design. Although all the possible scenarios cannot be captured with one test case, we try to capture the most significant behaviour to examine the performance of our ACC$^+$ system.

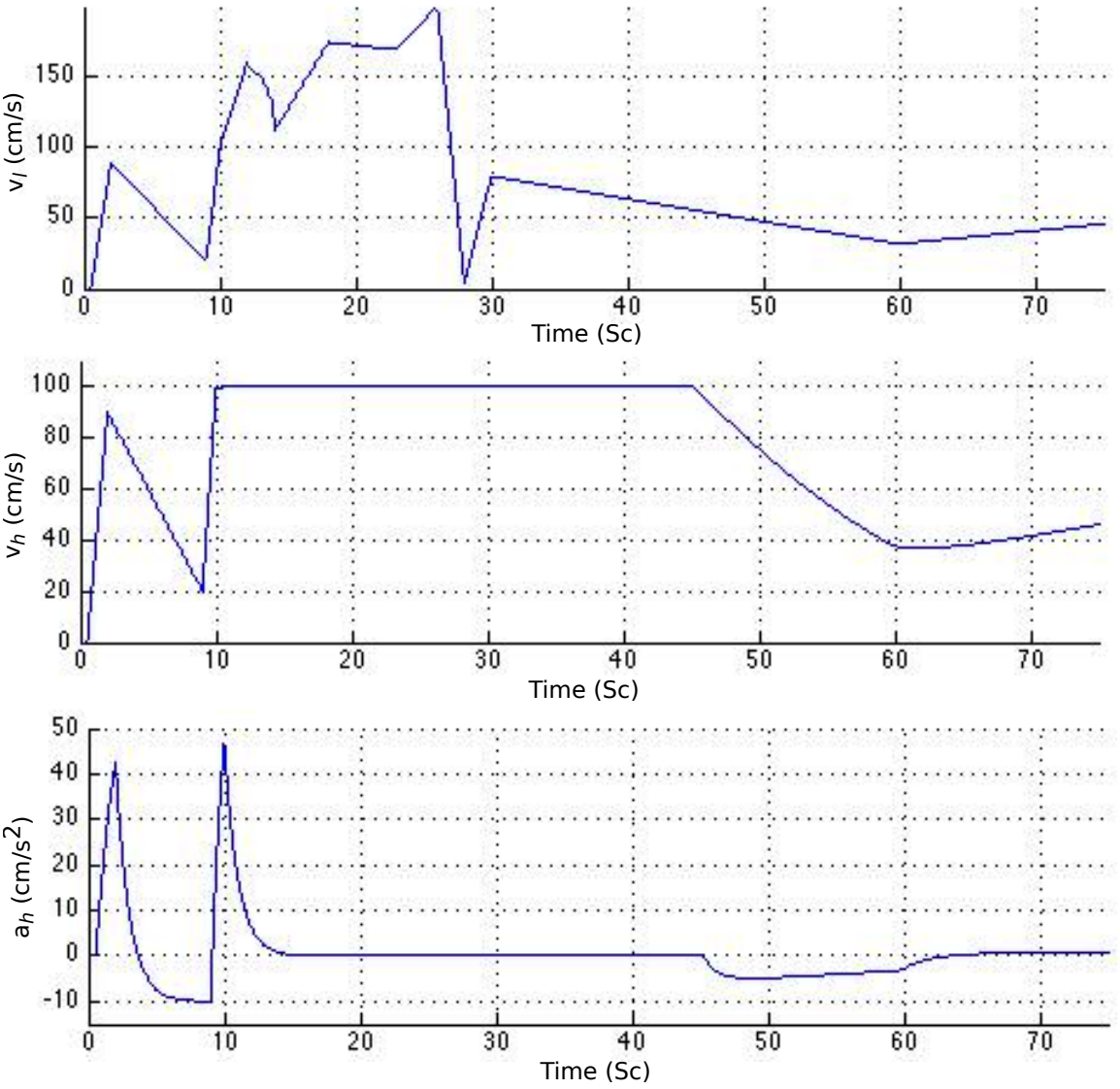


Figure 4.5: Simulation Results

Fig. 4.5 illustrates the behaviour of the host vehicle controlled by our ACC$^+$ system in various conditions such as when a lead vehicle is present at varying velocities or absent. The first plot in Fig. 4.5 depicts the velocity behaviour of the leader vehicle (v_l) which is measured in centimetre per second (cm/s). According to this plot, the

leader starts from an initial velocity of zero ($v_l = 0$) and changes its velocity as shown. The host vehicle starts at a certain desired headway at the beginning of the simulation. Therefore, the host vehicle tracks the leader's velocity in *Follow* mode until the leader travels faster than the host vehicle's set point velocity (v_{set}) at time $t = 10$. This behaviour can be seen in the second plot of Fig. 4.5, where the host vehicle's velocity is shown. The host vehicle's set point velocity is defined as 100 cm/s ($v_{set} = 100$). Therefore, the second plot of Fig. 4.5 shows the host vehicle tracks the leader's velocity until it exceeds $v_{set} = 100$ cm/s and the ACC$^+$ mode of operation is changed from *Follow* to *Cruise*. As shown in the first plot of Fig. 4.5, the leader vehicle decreases its velocity to lower values than the host's v_{set} at approximately $t = 27$. However, the ACC$^+$ system does not immediately change its mode of operation and the host vehicle continues with $v_h = 100$ cm/s in *Cruise* mode as long as the relative distance between the two vehicles is greater than the required distance for following the leader. The ACC$^+$ system changes the mode from *Cruise* to *Follow* only when d_{gap} becomes less than or equal to *l_dist*. This occurs around $t = 45$ when the host vehicle decreases its velocity in order to attain the leader's velocity by the desired headway. Consequently, the host vehicle matches its velocity with that of the lead vehicle while maintaining the desired headway.

The third plot of Fig. 4.5 shows the host vehicle's acceleration. According to this plot, the acceleration increases when the host vehicle is accelerating its velocity at first. The acceleration increases from zero to approximately 40 cm/s^2 at first, while the host vehicle's velocity increases from zero to 90 cm/s. The acceleration does not become negative right after the host vehicle starts decreasing its velocity from 90 cm/s to 20 cm/s. The reason for the host vehicle to still have a positive

acceleration, even when the lead vehicle is decelerating, is the presence of an integral term in the continuous controller. When initially developing the safety verification of the refined ACC$^+$ controller that appears in the following section (Section 4.6), the specification stated that if $v_l < v_h$ in *Follow* mode, then the acceleration of the host vehicle chosen by the controller, $a_h < 0$. Clearly a reasonable linear control system design, such as the one simulated in Fig. 4.5, does not satisfy this property. The verification in the following section was modified to allow positive accelerations even in the case when $v_l < v_h$ precisely for this reason. The lesson here is that one has to be careful to make sure that the formal model that is verified faithfully models the actual system (validation).

The host vehicle changes its acceleration in order to maintain a required velocity. However, due to the nature of the continuous controller, the acceleration control it generates does not change instantaneously due to the continuous dynamic of the system. This ACC$^+$ system attains the leader's velocity by a desired headway if the leader travels slower than the host vehicle's set point velocity. In addition, this system will continue to track the leader' velocity after the desired headway is achieved. In the absence of a slower leader, the system's objective is to track a desired set point velocity. As shown in Fig. 4.5, our ACC$^+$ system behaves safely and will not switch to *Safety_Critical* mode in this particular normal operation scenario. This control structure is not conservative because the required safety constraints are considered as a separate mode of operation and do not affect the operation of *Cruise* and/or *Follow* modes. Therefore, the required safety constraints and the desired performance could be obtained simultaneously by this design.

4.6 Verification

In this section we provide a formalization for the refined mode switching of the ACC$^+$ system described in Section 4.3 using *differential dynamic logic* ($\mathsf{d}\mathcal{L}$). This formalization is presented in **Model 2**. We use the dynamic operations of the host and lead vehicles as defined in Section 3.2 (Eq. 3.4 & Eq. 3.5). The leader vehicle behaviour is the same as in **Model 1** in Section 3.2, where acceleration can be chosen from the valid range (Eq. 3.6, Line(3)). The nondeterministic repetition $*$ and parallel operation of the host and leader vehicle has been already defined in **Model 1** (Line (1-2)). The *Other* functionality of the ACC$^+$ system in **Model 1** is now formalized as successive actions to capture other driving modes. The host vehicle controller takes action in a more restricted manner. Safety constraints related to relative distance, velocities and the selected velocity for *Cruise* mode must be satisfied. The host vehicle has three different operating modes that are represented sequentially in Line (4).

Three operating modes *Cruise*, *Follow*, and *Safety_Critical* always use the current value of $sc_{gap}(v_l, v_h)$, $margin_{sc_{gap}}(v_h)$, and d_{gap} in Line (5) to randomly choose the desired acceleration non-deterministically within the valid range to control the speed of the host vehicle under given safety margins.

The *Cruise* operating mode states that if d_{gap} is not less than or equal to $sc_{gap}(v_l, v_h)+$ $margin_{sc_{gap}}(v_h)$ and the speed of the lead vehicle is greater than v_{set}, then either the acceleration of the host vehicle a_h can be assigned non-deterministically from $-B \leq a_h \leq A_{max}$ when speed of the host vehicle v_h is less than or equal to v_{set}, or a_h can be assigned non-deterministically from $-B \leq a_h \leq A_{max}$ when the speed of the host vehicle v_h is greater than v_{set}. This operating mode is formalized in Line (6), where the speed of the host vehicle is always maintained according to the selected

driver speed, considering the speed of lead vehicle and safety margins. The range of deceleration is formalized based on the band limit of the continuous controller. For instance, if the continuous controller has an integral term, the acceleration might continue for some time with a value greater than zero even when $v_h > v_{set}$; hence, when performing the verification, the range of a_h cannot be restricted to a value between $-B$ to 0 in the case when $v_h > v_{set}$. By a similar reasoning, the range of a_h cannot be restricted to a value between 0 to A_{max} in the case that $v_h \leq v_{set}$. Thus, we consider the range of $-B \leq a_h \leq A_{max}$ for both mentioned cases. In the work of (Loos *et al.*, 2013), the behaviour of the continuous controller is not taken into account. The verified ACC controller in (Loos *et al.*, 2013) may not apply to certain controllers, such as PID controller, since the band limit of the continuous controller is not included in the verification. This concept was explained in detail in Section 4.5. Although the host vehicle's acceleration can be formalized based on the continuous controller (i.e., for a PID controller $a_h := k_1 + k_2 \times \int v_h(t).dt + k_3 \times \frac{\mathrm{d}}{\mathrm{d}t} v_h(t)$), we use the possible physical range ($-B \leq a_h \leq A_{max}$) to capture a class of possible continuous controllers. Accordingly, the continuous controller can be an arbitrary design without any concern about the safety properties of ACC$^+$ system.

The *Follow* operating mode is applicable only when the speed of the lead vehicle is less than or equal to the driver selected desired speed, (i.e. $v_l \leq v_{set}$). The *Follow* operating mode is specified in Line (7) of **Model 2** that specifies that if d_{gap} is not less than or equal to $sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)$ and the speed of lead vehicle is less than or equal to v_{set}, then the acceleration of the host vehicle a_h can be assigned non-deterministically based on the current status of the host and lead vehicles' velocities (v_h and v_l). If the host's velocity is greater than the leader's velocity ($v_h > v_l$)

then a_h should be negative ($-B \leq a_h < 0$). This case happens normally when the ACC$^+$ system detects a slower leader vehicle and should decrease its current velocity gradually in order to maintain v_l. Although a_h should be negative in this case, we considered its value between $-B$ and A_{max} ($-B \leq a_h \leq A_{max}$) in the formalization. This demonstrates that the continuous controller may work with a positive acceleration for a short time interval until obtaining a negative value. Therefore, $-B \leq a_h \leq A_{max}$ is not in contradiction with the expected behaviour in *Follow* mode by the reasoning provided in Section 4.5. After maintaining v_l, the ACC$^+$ system should track the leader's behaviour. Therefore, if the leader accelerates, and $v_h \leq v_l$, then of the possible values in $-B \leq a_h \leq A_{max}$, one would expect the controller to trend towards values of $a_h \geq 0$. As mentioned earlier, a_h cannot be switched from a negative to a positive value instantly due to the continuous behaviour of the system. Therefore, a_h cannot be formalized between 0 to A_{max} in the last case when $v_h \leq v_l$, otherwise significant jerk may occur in the system. As a result, $-B \leq a_h \leq A_{max}$ is a reasonable range to be considered for any arbitrarily continuous controller. As a result, the formal model corresponds to the actual behaviour of the practical control system.

The ACC$^+$ system can make any of these two choices according to the situation. Furthermore, the current values of $f_{gap}(v_l, v_h, a_h)$ and $margin_{f_{gap}}(v_h, a_h)$ are calculated sequentially, where system must be satisfied by $d_{gap} - (h_{set} \times v_l) \leq f_{gap}(v_l, v_h, a_h) + margin_{f_{gap}}(v_h, a_h)$. The test checks that the host vehicle is within $f_{gap}(v_l, v_h, a_h)$ to make sure that the transition to *Follow* is done properly and there is enough distance to maintain v_l as the new velocity. If the test condition does not hold ($d_{gap} - (h_{set} \times v_l) > f_{gap}(v_l, v_h, a_h) + margin_{f_{gap}}(v_h, a_h)$), then the execution will fail. Therefore,

this assertion forces the system operation to maintain enough distance for taking an appropriate action in *Follow* mode. Although this test does not have any impact on the proof of safety and collision-freedom of **Model 2**, we have defined this assertion to allow the conformity of this formalization to the actual mode switching system of Section 4.3.

In the case that this test cannot be satisfied, there can be two subsequent cases. First there is the normal behaviour in the presence of a slower leader when the sensor detects a slower leader, but the host vehicle is still able to travel at $v_h = v_{set}$ until it comes within $f_{gap}(v_l, v_h, a_h)$ of the lead vehicle. The second case for this violation is the opposite problem where there is not enough of a gap to reduce the host vehicle to v_l by the time the host vehicle is within the desired headway h_{set}. However the $f_{gap}(v_l, v_h, a_h)$ distance is always greater than the minimum stopping distance $sc_{gap}(v_l, v_h)$ in the presence of a slower leader vehicle. Further, the maximum delay for the ACC$^+$ system to react to a change, ϵ, has also been taken into consideration during the calculation to estimate the additional safe distance margin for $f_{gap}(v_l, v_h, a_h)$ in order to provide sufficient time for the controllers to react. Line (7) formalizes the behaviour of the *Follow* mode in the case when the ACC$^+$ system starts to decrease the host vehicle's velocity to match a slower leader's velocity by the time the host vehicle reaches distance $h_{set} \times v_l$, and then track the leader's velocity at an appropriate distance as long as the leader does not travel faster than v_{set}. This formalization also captures the behaviour of the *Follow* mode after the host vehicle starts to track the leader's velocity. Line (8) formalizes the *Safety_Critical* mode as defined previously in **Model 1**. The sampling time and dynamic evolution of the system are defined in Line (9), similar to **Model 1**.

Model 2: Formalization of refined ACC$^+$ system

$$ACC^+ \equiv (Vehicle; Drive)^* \quad (1)$$

$$Vehicle \equiv host \,||\, leader; \quad (2)$$

$$leader \equiv a_l = *; ?(-B \leq a_l \leq A_{max}) \quad (3)$$

$$host \equiv Calc\_sc_{gap};\ Cruise;\ Follow;\ Safety\_Critical; \quad (4)$$

$$\begin{aligned} Calc\_sc_{gap} \equiv\ & sc_{gap}(v_l, v_h) := \tfrac{v_h^2 - v_l^2}{2 \times B}; \\ & margin_{sc_{gap}}(v_h) := (\tfrac{A_{max}}{B} + 1)(\tfrac{A_{max}}{2} \times \epsilon^2 + \epsilon \times v_h); \\ & d_{gap} := x_l - x_h; \end{aligned} \quad (5)$$

$$\begin{aligned} Cruise \equiv\ & \text{if } \big(\neg(d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)) \wedge v_l > v_{set}\big) \text{ then} \\ & \big(?(v_h \leq v_{set}); a_h := *; ?(-B \leq a_h \leq A_{max})\big) \bigcup \\ & \big(?(v_h > v_{set}); a_h := *; ?(-B \leq a_h \leq A_{max})\big) \\ & \text{fi}; \end{aligned} \quad (6)$$

$$\begin{aligned} Follow \equiv\ & \text{if } \big(\neg(d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)) \wedge v_l \leq v_{set}\big) \text{ then} \\ & \big(?(v_h > v_l); a_h := *; ?(-B \leq a_h \leq A_{max})\big) \bigcup \\ & \big(?(v_h \leq v_l); a_h := *; ?(-B \leq a_h \leq A_{max})\big) \\ & f_{gap}(v_l, v_h, a_h) := \tfrac{v_h^2 - v_l^2}{-2 \times a_h}; \\ & margin_{f_{gap}}(v_h, a_h) := (\tfrac{A_{max}}{-a_h} + 1)(\tfrac{A_{max}}{2} \times \epsilon^2 + \epsilon \times v_h); \\ & ?(d_{gap} - (h_{set} \times v_l) \leq f_{gap}(v_l, v_h, a_h) + margin_{f_{gap}}(v_h, a_h)) \\ & \text{fi}; \end{aligned} \quad (7)$$

$$\begin{aligned} Safety\_Critical \equiv\ & \text{if } \big(d_{gap} \leq sc_{gap}(v_l, v_h) + margin_{sc_{gap}}(v_h)\big) \text{ then} \\ & a_h := -B \\ & \text{fi}; \end{aligned} \quad (8)$$

$$\begin{aligned} Drive \equiv\ & t := 0; (x_h' = v_h \wedge v_h' = a_h \wedge x_l' = v_l \wedge \\ & v_l' = a_l \wedge t' = 1 \wedge v_h \geq 0 \wedge v_l \geq 0 \wedge t \leq \epsilon) \end{aligned} \quad (9)$$

The main purpose of the refined ACC$^+$ formalization given in **Model 2** is to investigate the safety of the ACC$^+$ system in the presence of a leader in front of the host vehicle. Additionally we also want to ensure that the vehicle behaves safely when switching between the different modes of operation. The host vehicle's behaviour

when the lead vehicle is out of range of the sensor is the same as conventional cruise control systems. The *Cruise* mode controls the speed of the host vehicle on behalf of the driver. In the current formal model of the refined ACC$^+$ system, d_{range} has not been defined (i.e., we assume that the sensor range is effectively infinite). It is left as future work to prove the correctness of the system with a limited range sensor under the conditions outlined in (Eq. 4.4).

For now, we consider that there is a leader vehicle in the same lane as the host vehicle in **Model 2**. The system checks whether it can satisfy the safety property in the case when an obstacle or lead vehicle is detected. Once the path is cleared from any obstacle or there is no longer a lead vehicle, then it can switch back to the *Cruise* mode to maintain the desired speed (v_{set}).

The proposed ACC$^+$ design has three operating modes, where the system is switching from one mode to another according to desired situation considering safety constraint. The safe distance formula is the most important invariant that must be always satisfied by the ACC$^+$ system in all the operating modes as stated by the Controllability property (Eq. 3.8) in Section 3.2. We wrote **Model 2** in the KeYmaera theorem prover's input language to further demonstrate that this ACC$^+$ system design is safe and collision free as long as the safety critical distance condition (Eq. 3.8) has not been violated.

$$\text{Controllability Condition (3.8)} \rightarrow [\text{Refined ACC}^+] \;\; x_h < x_l \tag{4.24}$$

The precondition for Formula 4.24 is similar to that of Formula 3.7. It indicates that for all iterations of the Refined ACC$^+$ (**Model 2**), the system is collision free ($x_h < x_l$) if the controllability condition (3.8) is satisfied. This fact confirms that the

ACC$^+$ system in **Model 2** is a refinement of **Model 1**. Therefore any system, such as **Model 2**, will be safe as long as the controllability condition (3.8) is maintained. We will further investigate the refinement and refactoring relations between **Model 1** and **Model 2** in the next section. The safety of a complex model, such as **Model 2**, can be proved based on an abstract model, such as **Model 1**. The refactoring relation makes the proof procedure easier than the procedure we have done for proving relation 4.24. Additionally, the refinement relation allows designers to add new requirements to a system and/or change some parts of the system without violating the required safety properties. Consequently, it can be shown that **Model 2** is derived form the abstract model of Section 3.2 (**Model 1**) by adding some new states and refining the system's behaviour while preserving the required safety properties.

4.7 Safe Refactoring

Direct proof of safety and other properties of a complex cyber physical system is often difficult, if not impossible, due to the complex interaction between software and hardware models. Different approaches have been investigated to overcome this fact such as over-approximating the reachable set of states and defining an abstract model in order to reduce the complexity (Johnson *et al.*, 2012). The abstract model then can be verified for safety purposes. However, an important part of this method, which is typically disregarded in this area, is to prove that the original, complex system model is a property preserving refinement of the proposed abstraction. After verifying safety of an abstract model of a cyber physical system, any update in any part of that model requires reverification of the whole new system. Refinement reasoning makes the reverification process easier by assuring that the new additional part of

the system does not violate the safety of the whole system. Platzer and his coworkers recently proposed a refinement relation for systems described in *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) in (Mitsch *et al.*, 2014). There, they introduced two notions of refinement "*Projective Relational Refinement*" and "*Partial Projective Relational Refinement*". According to their work:

> "***Projective Relational Refinement:*** *Let $V \subseteq \Sigma$ be a set of variables. Let $|_V$ denote the projection of relations or states to the variables in V. We say that hybrid program α refines hybrid program γ w.r.t the variables in V ($\alpha \sqsubseteq^V \gamma$) iff $\rho(\alpha)|_V \subseteq \rho(\gamma)|_V$.*"

where ρ is the transition relation used to specify reachable states.

Although we used KeYmaera (Platzer and Quesel, 2008) to prove safety property (Eq. 4.24) of **Model 2** in Section 4.6, we want to further investigate refinement reasoning. We defined a safe abstract model of any ACC or ACC$^+$ system in Chapter 3 and proved the collision-freedom property of that model. In this section we want to show that **Model 2** refines **Model 1**. The Projective Relational Refinement definition holds for **Model 2** with respect to **Model 1** since the reachable states of **Model 2** are a subset of the reachable states of **Model 1**. We can thus conclude that **Model 2** refines **Model 1** with respect to the variables of these models (**Model 2** $\sqsubseteq^V$ **Model 1**). Therefore, **Model 2** inherits the collision freedon safety property from **Model 1**. We will use refactoring methods from (Mitsch *et al.*, 2014) to demonstrate the validity of this claim.

(Mitsch *et al.*, 2014) developed "*proof-aware refactoring*" and proposed some rules with associated proof obligations to define a refinement relation in terms of refactoring. Two refactorings, *Structural* and *Behavioral*, are defined in (Mitsch *et al.*, 2014).

"*Structural refactoring changes the structure of a hybrid program without changing its reachable states*"; while, "*Behavioral refactoring partially changes the reachable states*". Therefore, some auxiliary proof obligations are necessary to demonstrate inheritance of safety or correctness properties in behavioral refactoring. We use "safety relational refinement" and "auxiliary safety proof" from (Mitsch *et al.*, 2014) for refinement reasoning.

> "*Safety relational refinement. Prove that all reachable states from the refactored model α are already reachable in the original model γ.*"

> "*Auxiliary safety proof. Prove that a refactored model α satisfies some safety properties under the assumption of an existing proof about the original model γ. The auxiliary safety proof patches this proof w.r.t. the changes made by the refactoring. Let $\forall^\gamma$ quantify universally over all variables that are changed in γ. The intuition is that, assuming $\models \forall^\gamma(\phi \rightarrow [\gamma]\phi)$ (ϕ is an inductive invariant of γ), we can close the identical parts in the proof from the assumption by axiom and only need to show correctness for the remaining, new parts of the refactored model. For auxiliary safety use an invariant of $\mathcal{I}(\phi) \equiv (\phi \wedge \forall^\gamma(\phi \rightarrow [\gamma]\phi))$ for the refactored program α to prove $(F \wedge \mathcal{I}(\phi)) \rightarrow [\alpha^*]\psi$.*"

where F is some formula based on the definition of partial projective relational refinement. A hybrid program α is a partial refinement of γ with respect to some variables in the set of variables V and some formula F ($\alpha \sqsubseteq_F^V \gamma$) if and only if $(?F;\alpha) \sqsubseteq_F^V (?F;\gamma)$. In the case that $F \equiv true$, this partial refinement relation becomes a total refinement

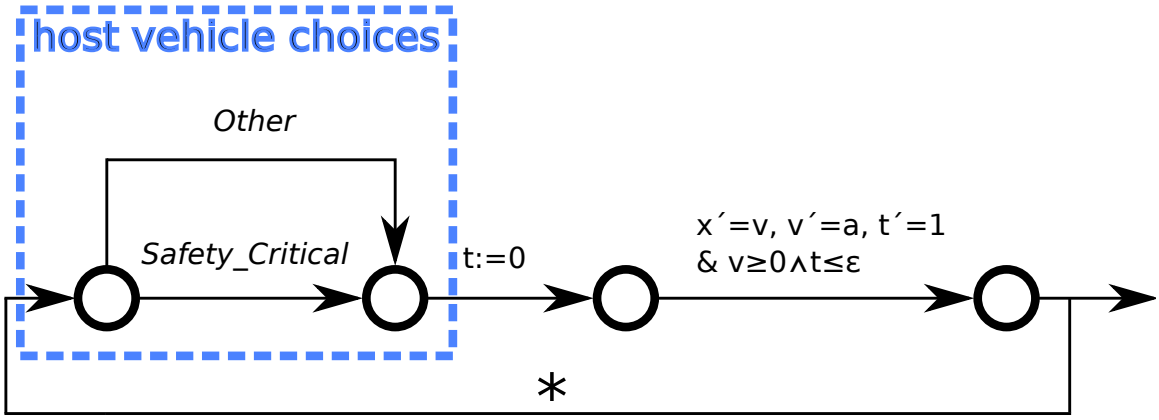


Figure 4.6: Time-Triggered Architecture of Abstract ACC$^+$

relation ($\alpha \sqsubseteq^V \gamma$ iff $\alpha \sqsubseteq^V_{true} \gamma$). Therefore, F in $(F \wedge \mathcal{I}(\phi)) \rightarrow [\alpha^*]\psi$ is an additional condition for partial refinement cases.

According to the above definitions from (Mitsch *et al.*, 2014), we want to show that if abstract model, **Model 1**, guarantees a safety property, e.g., collision-freedom, then this safety property can be proven for a refactored model, **Model 2**. We translate our problem using the auxiliary safety proof method as shown below:

- α is "Refined ACC$^+$" (**Model 2**)
- γ is "Abstract ACC$^+$" (**Model 1**)
- ϕ is "Condition (3.8)"
- ψ is $x_h < x_l$

We already proved that: $(\phi \rightarrow [\mathit{Abstract\ ACC^+}]\ x_h < x_l)$ as in Eq. 3.7. We want to show the same collision-freedom $(x_h < x_l)$ is valid for the refactored model, in this case the refined ACC$^+$ $(\phi \rightarrow [\mathit{refined\ ACC^+}]\ x_h < x_l)$. Therefore, we should strengthen the inductive invariant of **Model 1**, the Controllability Condition (3.8),

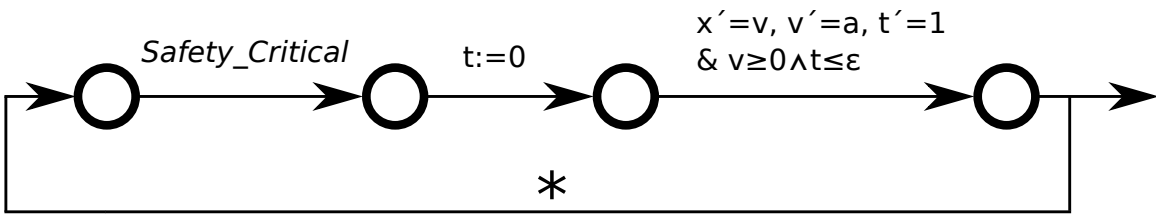


Figure 4.7: Removed branch model

with the safety approved assumption for the abstract model.

$$\mathcal{I}(\phi) \equiv (\phi \wedge \forall x \forall v(\phi \rightarrow [\text{Abstract ACC}^+]\ \phi))$$

We want to formally prove that: $\mathcal{I}(\phi) \rightarrow [\text{Refined ACC}^+]\ x_h < x_l$

The *Event- to time-triggered architecture* refactoring changes a hybrid program from event-triggered to timed triggered. This refactoring process separates the continuous evolution of the system from control choices. Fig. 4.6 shows the time-triggered architecture of **Model 1** (Abstract ACC$^+$) as a state transition system. The procedure of deriving this architecture can be found in (Mitsch *et al.*, 2014).

Fig. 4.7 demonstrates removing one branch (*Other*) from the original model (Fig. 4.6) while the safety property is still preserved, and then Fig. 4.8 introduces two new branches (*Cruise* and *Follow*) to Fig. 4.7 without changing the *Safety_Critical* branch. Both figures Fig. 4.7 and Fig. 4.8 depict the "*Introduce Control Path*" refactoring, which is defined under the category of "*Behavioral Refactorings*" in (Mitsch *et al.*, 2014). Finally, Fig. 4.8 shows the time-triggered architecture of **Model 2** (Refined ACC$^+$). The safety proof procedure of this refactored model can then be constructed from Fig. 4.7 and Fig. 4.8. The details of this proof are left as future work. Although this procedure provides easier steps in the safety proof of the refined system, we want

to further demonstrate that one transition is split into two transitions. The two new transitions cover the same guard condition while each transition has a subset of the old transition's behaviour. Therefore, we want to further prove the case splitting of the *Other* (old transition), which means that *Cruise* and *Follow*, as new transitions, result in a subset of the behaviour of *Other*.

Another use of refinement of these two models (i.e. **Model1** and **Model2**) is to demonstrate that the abstract model can be improved and adapted to a more complex model in order to meet new requirements. We want to use a refactoring from (Mitsch *et al.*, 2014) to prove a safety property of a refined model based on the abstract one. However, "*proof-aware refactoring*" in (Mitsch *et al.*, 2014) does not propose any path-split (case splitting) refactoring. In other words, we need to add an additional refactoring proof in *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) to show that a transition in the abstract model can be split in to two or more new branches. In this case the *Other* transition is split without touching the *Safety_Critical* case in order to preserve safety of the whole new system.

We want to show that *Other* mode in **Model1**, Fig.3.1, is split into two new modes *Cruise* and *Follow* in **Model2**, Fig. 4.4, without touching the *Safety_Critical* mode. This fact can be established by proving that new branches apply in the same situations as the old branches and each will not violate the acceptable range for parameters of the old branch. In our example, a_h in *Cruise* and *Follow* will not be out of the acceptable range which has been already defined in *Other* ($-B \leq a_h \leq A_{max}$). Therefore, another notion of refactoring and refinement (path-split) can be introduced in the proof refactoring of *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) that can be shown to preserve safety properties.

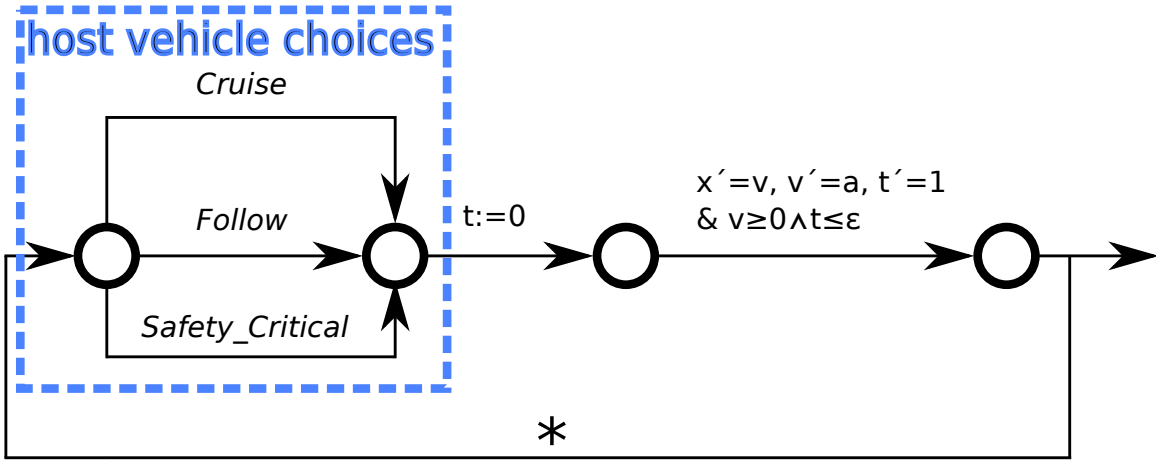


Figure 4.8: Time-Triggered Architecture of Refined ACC$^+$

Consequently, the whole refactoring procedure with path-split refinement can be done more easily than the steps which we have done based on "*proof-aware refactoring*". Using this technique we can deduce Fig. 4.8 from Fig. 4.6 directly, without the intermediate step shown in Fig. 4.7. Providing the formal syntax and semantics of path-split refinement is, again, left as future work.

4.8 Summary

In this chapter, an ACC$^+$ system was developed in the form of a hybrid system. Three different modes of operation were defined based on the system requirements. The desired system behaviour was captured via the interaction between a finite state machine and low level continuous controllers. The proposed ACC$^+$ system was also formalized using *differential dynamic logic* (d$\mathcal{L}$) to prove the system's collision freedom. It was proven that the vehicle behaves safely when switching between the different modes of operation. The safety invariant of the proposed system is similar to the abstract model described in Chapter 3.

The refinement relation for the *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) was investigated at the end to formally demonstrate that the proposed ACC$^+$ system refines the abstract model while preserving the safety concept. In addition, some future works were proposed in this chapter, such as formal proof of the thrash freedom of the mode switching system. Lastly, a new concept of refinement (path-split) was introduced as another future work to be included in the proof refactoring of *differential dynamic logic* ($\mathsf{d}\mathcal{L}$). As a conclusion, the formalized ACC$^+$ model is not only safe, but also can capture all the physically possible continuous controllers.

Chapter 5

Evaluation and Results

The motivation behind providing a modularized ACC$^+$ system approach is to further prove safety and scalability for future development, and to fully define the required system actions. Our aim in designing the presented ACC$^+$ system is to modularize the system in the most understandable and scalable method possible to avoid any arbitrary behaviours, and to be able to expand the system easily in future. In this chapter a comparison is made between our work and related work in this area. We perform this comparison by examining some mathematical and simulation results from each work.

5.1 ACC$^+$ Design vs. ACC Design

In (Breimer, 2013), testing and simulation have revealed some unreachable states as a result of unnecessary zones of operation, which also makes the system too complicated. As a result, this implementation does not satisfy the desired safety properties. Fig. 5.1 shows the design of ACC by (Breimer, 2013).

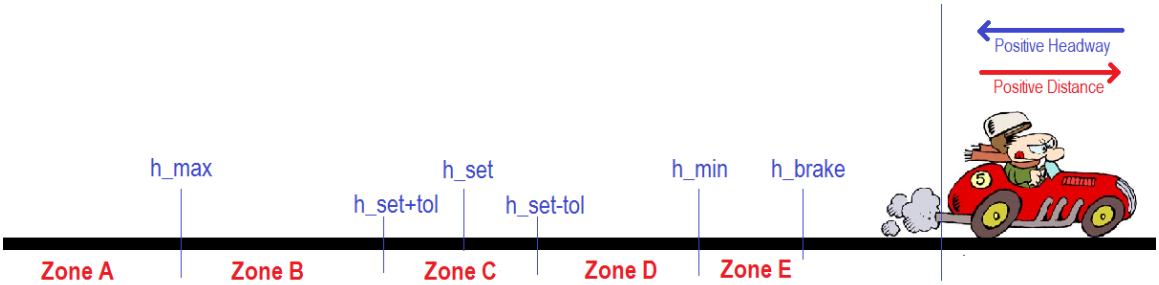


Figure 5.1: Diagram of Headway Zones in ACC Design
(Figure provided by (Breimer, 2013))

In this design, an ACC system was proposed by describing different zones of operations. As the figure 5.1 shows, the headway is divided into five zones from Zone A to Zone E, where unnecessary zones of operation have been defined. Although the system tries to cover all possible scenarios that can occur, some of the zones, such as Zone B, are not useful in the decision-making process. The host vehicle's next action, following a leader vehicle, starts when the vehicle reaches the desired headway or when it is in Zone D. However, this action can be too late in practice due to the vehicle's dynamics. As a result, the system may need to use a severe brake, despite its purpose being to manipulate the brake. Another issue in this design is that the size of the zones has been defined using fixed values rather than a mathematical definition. Testing and simulations have demonstrated some unreachable states in the system as a result of operation overlapping in different zones. Thrashing and collision are inevitable consequences of this design and thus the desired safety properties cannot be guaranteed.

In addition, (Breimer, 2013) has proposed four active states named *Cruise*, *Close Gap*, *Maintain Headway*, *Apply Brakes* to capture all of the mentioned zones of operation. However, the functionality of *Close Gap* and *Maintain Headway* are closely

related. *Close Gap* is used to maintain the headway requested by the driver, while *Maintain Headway* has been defined for tracking the leader's velocity to maintain a constant headway. Having two states for one goal can result in thrashing as we have discovered in different test cases in both Simulink and Test Bench. These two states can be merged into one state even for a simple PID controller. Moreover, the study does not provide any definition about how safety critical cases can be determined and controlled in *Apply Brakes*. One of the thrashing instances can be quickly observed when the host vehicle's velocity is slightly greater than that of the lead vehicle and moves from Zone C to Zone D. The host vehicle repeatedly moves into Zone D when operating in a *Cruise* state and then falls into Zone C when switching controllers to operate in its next state. Fig. 5.2 shows the host vehicle's velocity and relative distance between vehicles for the top lead vehicle's velocity behaviour.

In Fig. 5.2 the first plot (**A**) shows the lead vehicle's velocity ("**Leader's Velocity**"), which starts from velocity zero ($v_l = 0\ cm/s$) and moves on to 52 cm/s at the end. The second plot (**B**) demonstrates the ACC behaviour in (Breimer, 2013) ("**Host's Velocity**"). In this graph, mode thrashing is apparent from the oscillations in the host's velocity. This mode thrashing is due to some unreachable states as a result of the mentioned unnecessary zones of operation. At the start of the simulation time, the host vehicle is in Zone C and can choose *Cruise* or *Maintain Headway* in this zone. These two choices lead to thrashing not only in Zone C but also between Zone C and Zone D. The host's velocity is dropped and ramped quickly after switching from one state to another which makes a poor control on headway as illustrated plot (**C**): "**Relative Distance**" in Fig. 5.2. Plot (**C**) demonstrates the relative distance between the leader and the host vehicles. The relative distance becomes negative in

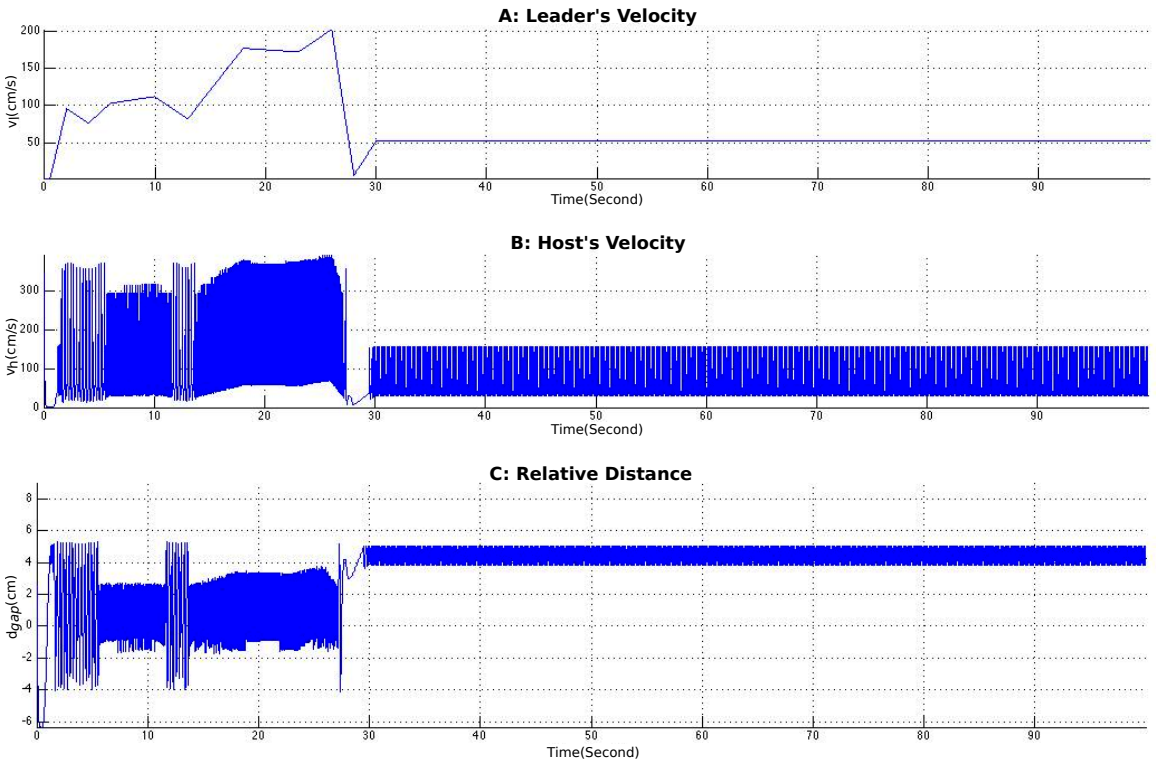


Figure 5.2: ACC Simulation Results

some of the time periods. The mentioned undesirable mode thrashing resulting in oscillation in plot (**B**) of Fig. 5.2 cannot attain the desired and safe relative distance between vehicles and collisions can be seen as in plot (**C**) of Fig. 5.2. Thus, this ACC system cannot control the system properly. Collision and unsafe behaviour can be captured in this model which makes the system unsafe and impractical.

In order to remove this unwanted behaviour, an ACC or ACC$^+$ system should be designed taking into consideration dynamic behaviour and discrete behaviour together to remove any arbitrary operation. In our ACC$^+$ system, the finite state machine (discrete behaviour) has been designed considering the longitudinal dynamic physics of movement. Therefore, the mentioned unwanted mode switching behaviour has been eliminated. Fig. 5.3 shows the host vehicle's velocity of our ACC$^+$ implementation

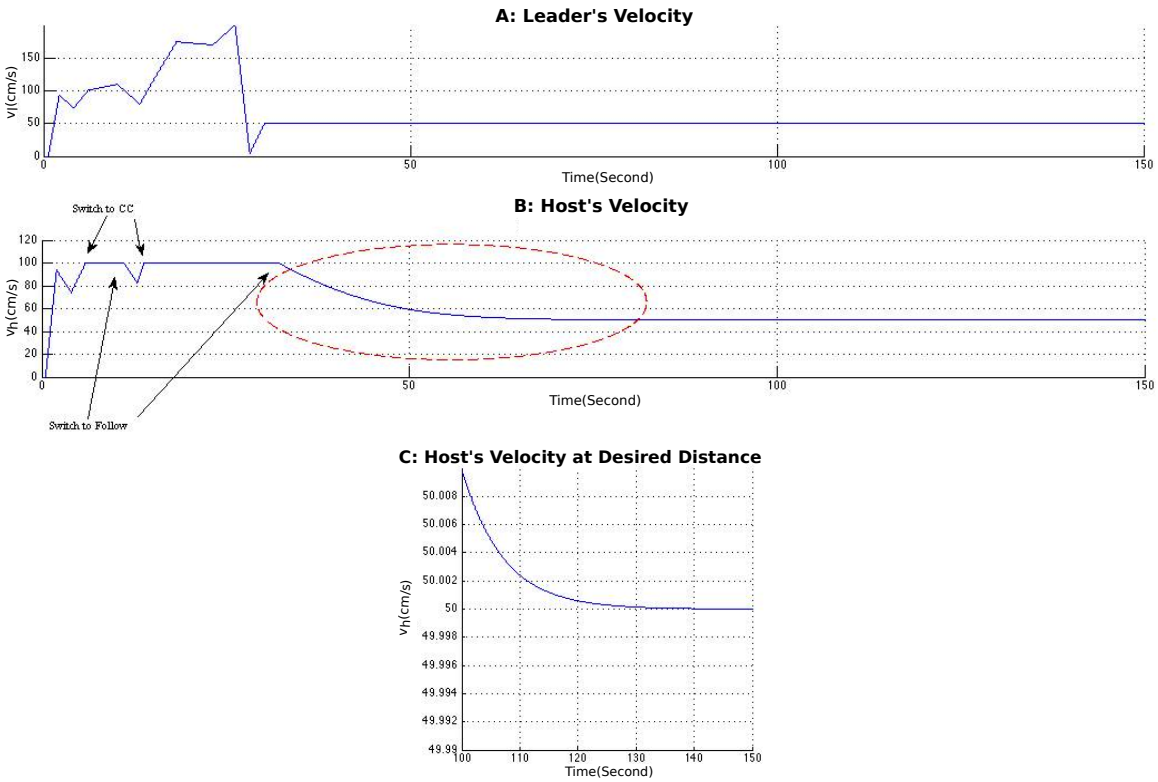


Figure 5.3: ACC+ Simulation Results

for a similar test-case leader as the top plot (**A**) in Fig. 5.2. In plot (**A**) of Fig. 5.3, "**Leader's Velocity**", the velocity behaviour of the leader vehicle is shown, which reaches a terminal speed of 50 cm/s in the examined time interval.

We can focus on a unique desired behaviour while tracking the leader's velocity, which will lead the host vehicle to a desired and safe distance from the leader as shown in Fig. 5.3 and Fig. 5.4. The ACC$^+$ controller, plot (**B**) in Fig. 5.3 (i.e. "**Host's Velocity**"), tracks the lead vehicle smoothly while it maintains a safe distance from the leader without any thrashing. This safe distance can be interpreted from the relative distance between the host and lead vehicle according to plot (**A**) of Fig. 5.4.

Plot (**B**) in Fig. 5.3 demonstrates that at the starting point, the host vehicle tracks

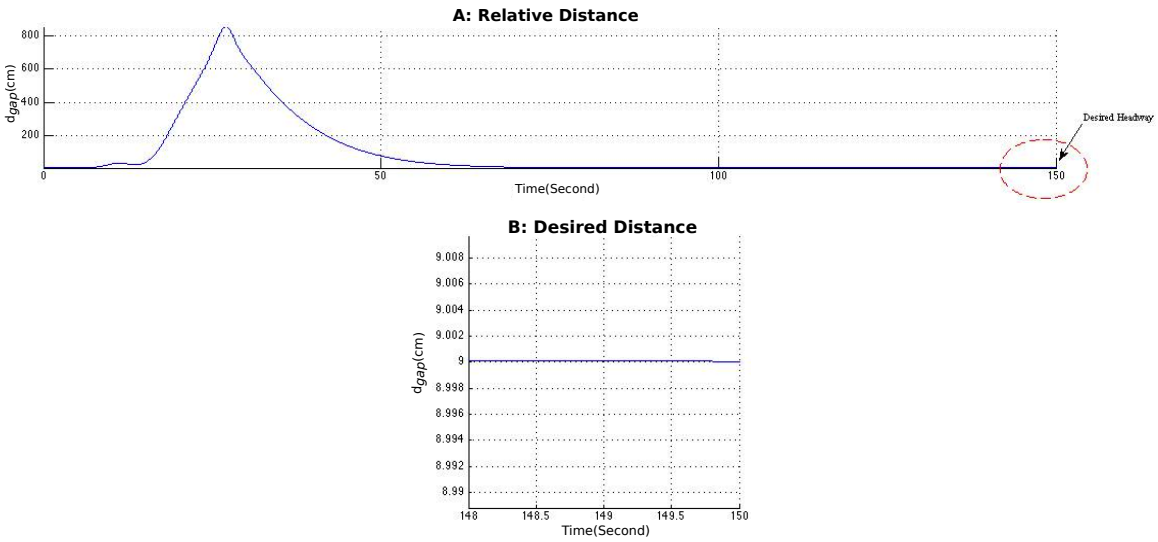


Figure 5.4: Host and Leader Relative Distance

the leader's behaviour in the *Follow* mode since relative distance is exactly the same as the desired distance. The ACC$^+$ system switches to *Cruise* mode after the leader goes faster than the driver's set point velocity and maintains 100 cm/s as desired velocity. One mode switching can be observed between *Cruise* to *Follow* and again to *Cruise* due to decreasing and increasing the leader's velocity from the upper limit velocity of host vehicle. These facts can be seen in plot (**B**) in Fig. 5.3 as indicated by the arrows. At the end, the leader decreases its velocity and travels at 50 cm/s after a while (plot (**A**) in Fig. 5.3), but the host vehicle is too far from the leader. Therefore, no action will take place. In this test case, we defined maxim deceleration due to maximum brake as 10 ($-B = -10\ cm/s^2$) and the desired deceleration as 7 ($a_h = -7\ cm/s^2$). After hitting the point of *l_dist*, the host vehicle starts to decelerate. The red dotted oval in plot (**B**) in Fig. 5.3 shows the deceleration in *Follow* mode.

The innovative part of this design is that the host vehicle will achieve 50 cm/s (leader's velocity) when the relative distance becomes the desired distance ($h_{set} \times v_l =$

9 cm). However, this headway time interval should not be too short such that it causes discomfort to the passengers. We consider a constant desired distance, 9 cm, for this simulation to make it simpler. We can show that the host vehicle can maintain leader's velocity (50 cm/s) by end of $h_{set} \times v_l$, which is the desired distance (9 cm for this test case). This is shown by a red dotted oval and an arrow in plot (**A**) of Fig. 5.4. Therefore, the host vehicle will have the same behaviour as the leader right after it has reached a 9 cm distance from the leader. The precision of both objectives (i.e. achieving leader's velocity by desired distance) in *Follow* mode is shown in plots (**C**) and (**B**) of Fig. 5.3 and Fig. 5.4, respectively. The absolute error in the system response for both velocity and distance is less than 0.001 according to plot (**C**) of Fig. 5.3 and the plot (**B**) of Fig. 5.4.

5.2 ACC$^+$ Verification vs. ACC Verification

(Loos *et al.*, 2013) recently presented an adaptive cruise control system. Their system conforms the safety property discussed in (Loos *et al.*, 2011). However, none of these publications have represented the exact behaviour of an ACC system. In the latest work (Loos *et al.*, 2013) has proposed an ACC based on different acceleration choices for different modes of operation. The idea of proposing such a system is in Eq. 5.1. The equation is provided by earlier work (Loos *et al.*, 2011), which says that the system is considered safe if the distance between two successive vehicles is greater than or equal to the distance required for the host vehicle to fully stop bumper to bumper if both vehicles fully stop using the same maximum brake. In this part, all the notations are the same as (Loos *et al.*, 2013) for consistency. The host vehicle is called follower in (Loos *et al.*, 2013); hence, x_h, v_h and a_h are replaced with x_f, v_f

and a_f respectively.

$$x_f + \frac{v_f^2}{2B} \leq x_l + \frac{v_l^2}{2B} \tag{5.1}$$

Eq. 5.1 demonstrates the safety property required for the ACC system to be considered collision free, where x_f & v_f are the position and velocity of the follower (host vehicle), x_l & v_l are the position and velocity of the leader, and B is the maximum deceleration due to maximum brake for both vehicles. The proposed ACC system, consists of a_f as the control action. a_f can accept various values based on the condition and situation of the follower vehicle. We provide a detailed review of their system and discuss some of its issues.

$$a_f(v_f, v_l, D, \tau) = \begin{cases} A & \text{if} \quad a \geq A \\ 0 & \text{if} \quad v_f = 0 \wedge a \leq 0 \\ a & \text{if} \quad a \geq \frac{-v_f}{\tau} \wedge -B \leq a \\ b & \text{if} \quad a < \frac{-v_f}{\tau} \wedge -B \leq b \\ -B & \text{o.w.} \end{cases} \tag{5.2}$$

In Eq. 5.2, τ is the time that the follower can proceed without violating the safety property in Eq. 5.1.

The evaluation of this system should be done in two steps. First, a description behind the values defined in Eq. 5.2 should be determined. Second, the transition conditions should be specified. In Eq. 5.2, A is the maximum acceleration of the follower vehicle, while $-B$ is the maximum deceleration it can afford using maximum brake. a and b are defined in Eq. 5.3 and Eq. 5.4 respectively.

$$a = \frac{\sqrt{B^2\tau^2 - 4Bv_f\tau + 8BD + 4v_l^2} - B\tau - 2v_f}{2\tau} \quad \text{where} \quad D = x_l - x_f \tag{5.3}$$

$$b = \frac{-v_f^2}{2(D + \frac{v_l^2}{2B})} \tag{5.4}$$

Eq. 5.4 can be derived from Eq. 5.5, where the absolute value of b ($|b|$) is less than the maximum brake B ($-B \leq b$). D is the relative distance between two successive vehicles. Eq. 5.4 is used to interpret the fact that if the ACC system can stop the follower (host vehicle) safely with less deceleration ($-b$) than maximum brake (B), then the ACC system has to use b as its negative acceleration.

$$x_f + \frac{v_f^2}{2 \times -b} \leq x_l + \frac{v_l^2}{2B} \tag{5.5}$$

In Eq. 5.2, transitions are based on the value of a. If a is more than the maximum acceleration then A should be chosen. One issue in Eq. 5.2 is in the stopping situation where v_f is zero ($v_f = 0$). It is trivially clear from Eq. 5.3 that if $v_f = 0$ then a is greater than or equal zero ($a \geq 0$). Therefore the reason for having the second condition is not clear and it makes the second and third conditions valid at the same time for the case that $a = 0$ and in other cases ($a > 0$) the second condition will never be valid.

In the third and fourth conditions, a has been compared to $\frac{-v_f}{\tau}$. The reason is to assure a is not going to cause the system move backwards. As mentioned earlier, a cannot be negative when $v_f = 0$. If a is less than $\frac{-v_f}{\tau}$ ($a < \frac{-v_f}{\tau}$) the follower vehicle

can not be fully stopped before the stopping point of the leader vehicle. Therefore, a is always greater than or equal to $\frac{-v_f}{\tau}$ ($a \geq \frac{-v_f}{\tau}$); otherwise, collision can occur. As a result, the fourth condition ($a < \frac{-v_f}{\tau} \wedge -B \leq b$) is never valid due to the first term ($a < \frac{-v_f}{\tau}$) in it.

It seems that the ACC system in (Loos *et al.*, 2013) is redundant and hence mode switching is a noticeable issue in this design. Another issue is in a, Eq. 5.3, which shows that $x_{f,af}(t_{stop}) = x_{l,-B}(t_{stop})$. One can derive a by solving this equation. It means that the follower vehicle can travel safely with a_f for τ seconds while maintaining the safety property in Eq. 5.1. However, Eq. 5.3 cannot cover all cases due to missing leader's acceleration and velocity as represented below.

$$\frac{1}{2}a_f\tau^2 + v_f\tau + x_f + \frac{(a_f\tau + v_f)^2}{2B} \leq x_l + \frac{v_l^2}{2B} \tag{5.6}$$

Eq. 5.3 can be derived by solving Eq. 5.6. The missing part is shown in Eq. 5.7.

$$\frac{1}{2}a_f\tau^2 + v_f\tau + x_f + \frac{(a_f\tau + v_f)^2}{2B} \leq \frac{1}{2}a_l\tau^2 + v_l\tau + x_l + \frac{(a_l\tau + v_l)^2}{2B} \tag{5.7}$$

By solving Eq. 5.7 a bound for a_f will be achieved as in Eq.5.8.

$$\frac{-\sqrt{B^2\tau^2 - 4Bv_f\tau + 8BD + 4v_l^2 + 4Ba_l\tau^2 + 8Bv_l\tau + 4a_l^2\tau^2 + 8a_lv_l\tau} - B\tau - 2v_f}{2\tau} \leq a_f \leq \frac{\sqrt{B^2\tau^2 - 4Bv_f\tau + 8BD + 4v_l^2 + 4Ba_l\tau^2 + 8Bv_l\tau + 4a_l^2\tau^2 + 8a_lv_l\tau} - B\tau - 2v_f}{2\tau} \tag{5.8}$$

Since the lowest bound should be $-B$ the non-equality in Eq. 5.8 should be as in Eq. 5.9.

$$-B \leq a_f \leq \frac{\sqrt{B^2\tau^2 - 4Bv_f\tau + 8BD + 4v_l^2 + 4Ba_l\tau^2 + 8Bv_l\tau + 4a_l^2\tau^2 + 8a_lv_l\tau} - B\tau - 2v_f}{2\tau} \tag{5.9}$$

It can be shown that the upper bound in Eq. 5.9 is always greater than or equal $-B$ as in Eq. 5.10.

$$-B \leq \frac{\sqrt{B^2\tau^2 - 4Bv_f\tau + 8BD + 4v_l^2 + 4Ba_l\tau^2 + 8Bv_l\tau + 4a_l^2\tau^2 + 8a_lv_l\tau} - B\tau - 2v_f}{2\tau} \tag{5.10}$$

The non-equality in Eq. 5.10 will lead to Eq. 5.11.

$$v_f^2 \leq v_l^2 + 2BD + Ba_l\tau^2 + 2Bv_l\tau + a_l^2\tau^2 + 2a_lv_l\tau \tag{5.11}$$

We rename the term $Ba_l\tau^2 + 2Bv_l\tau + a_l^2\tau^2 + 2a_lv_l\tau$ in Eq. 5.11 as P ($P := Ba_l\tau^2 + 2Bv_l\tau + a_l^2\tau^2 + 2a_lv_l\tau$). Eq. 5.12 shows the result of the renaming.

$$v_f^2 \leq v_l^2 + 2BD + P \tag{5.12}$$

P is always a positive value because $v_l \geq 0$ and $a_l \geq -B$. This shows that the safety property in Eq. 5.1 will be satisfied by the bound for a_f (Eq. 5.9).

It can therefore be concluded that the bound for a_f in Eq. 5.9 is both safe and valid. The next step is to relate τ and D to a desired headway. Although, (Loos *et al.*,

2013) proposed an optimal sampling time as 3.2 *seconds* ($\tau = 3.2$ *seconds*), this large sampling time is unrealistic for a real time system. Another requirement in the ACC design is achieving a desired velocity, which has not been discussed in (Loos *et al.*, 2013). Following behaviour is the only consideration in their work. Desired headway and desired velocity have been missed in their design. As mentioned in previous chapters, our ACC+ system provides the generalized solution and formalizes the system requirements to satisfy the desired safety properties discussed in both (Loos *et al.*, 2013) and (Loos *et al.*, 2011).

5.3 Summary

As a summary, some mathematical and simulation evaluations from design process to formal verification of ACC and ACC$^+$ system were made between our proposed model and related works in this area.

It was shown that achieving and tracking leader's velocity will lead the host vehicle to a desired and safe distance from the leader. This performance can be maintained by considering the interactions between continuous and discrete behaviour in the design process of the system. Real-time response of the system illustrated the accuracy and performance of our ACC$^+$ system while in a related work, (Breimer, 2013), undesired behaviour was captured. This unsafe behaviour of the ACC system of (Breimer, 2013) was due to the lack of assessment of the host and lead vehicles' continuous dynamics in the design of the state transitions. Therefore, some unreachable and redundant states of operation caused the poor performance in the mentioned work.

On the other hand, it was later shown that an ACC or ACC$^+$ system can be defined to conform safety property. However, the system might not correspond to

the required engineering principles. The example of such system was discussed from the work of (Loos *et al.*, 2013). This ACC system proposed in (Loos *et al.*, 2013) was described to have mode switching due to incorrect definitions for both state transitions and action as acceleration. Consequently, although the ACC system of the second related work is safe their proposed system is not realistic and makes dramatic discomfort for passengers.

Overall, a system designed without any proof of safety is not reliable. Also, it is not useful to define a system that can be proven safe without considering its practicality. Accordingly, safety and desired requirements should be considered throughout the design process and formal verification of the system. It is important to validate that the formal model is an accurate representation of its actual implementation. The formal model should precisely represent the practical control system.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Cruise Control (CC) and Adaptive Cruise Control (ACC) have been used by several car makers to regulate the speed of the car under traffic situation above some minimum speed. However, these methods are not suitable for very low speeds with frequent stops such as in a traffic jam. ACC$^+$ extends CC and ACC features to provide automatic speed regulation of the car in a traffic jam environment. Formal verification is a promising approach to verify the ACC$^+$ system behaviour with respect to the system requirements and guarantee that the system is collision free and safe under all possible scenarios. We have presented a general solution for preserving collision-freedom in any ACC$^+$ system. Therefore, any practical ACC$^+$ controller design optimized for fuel economy, passenger comfort, *etc.*, can then be verified by showing its correspondence with this general high level solution.

In addition, we have provided a complete solution for a practical ACC$^+$ design by formalizing the system's requirements. We have demonstrated a formal verification

method for this real-world hybrid control system using *Differential Dynamic Logic* ($\mathsf{d}\mathcal{L}$) to guarantee the collision freedom under typical environmental conditions with the aid of the formal verification tool KeYmaera. This model clearly presents safety of a hybrid system by defining system constraints in terms of bounds on the corresponding variables. As such, the formal model of the hybrid control system is verified, and this verified model is not only feasible, but also useful for improving the existing hybrid control systems.

6.2 Future Works

Possilbe further avenues for research in this area include:

- Implementing the proposed ACC$^+$ system on a realistic hardware platform to capture and validate the system requirements and safety properties of the system in practice.

- Determining an optimal value for acceleration of the vehicle with respect to various constraints, such as the relative distance between the host and lead vehicle, fuel economy, passenger comfort, traffic congestion, *etc.* This procedure can be done by modelling an optimization problem while defining the requirements as constraints.

- Proposing a formal verification method for "mode stability" analysis of the proposed hybrid automata. Therefore, an additional level of confidence can be achieved beyond the typical simulation and test cases that the proposed hybrid system does not have any rapid mode switching (thrashing).

- Proposing syntax and semantics for spilt-path refinement in *Differential Dynamic Logic* ($\mathsf{d}\mathcal{L}$). Therefore, any ACC$^+$ model can be easily extended in the future while the safety property has been already assured for the whole system.
- Extending the ACC$^+$ system to capture lateral dynamics.The ACC$^+$ system proposed in our work is only for longitudinal dynamics. Thus, with this extension, ACC$^+$ would be able to handle cornering.

Appendix A

Appendix

A.1 Calculating Safety Critical Distance

In this section we provide the procedure of deriving $sc_{gap}(v_l, v_h)$ in section 3.1. First, we show that $\frac{v^2}{2\times B}$ is the stopping distance by considering v as the velocity and B as the deceleration. The two following equations Eq. A.1 and Eq.A.2 show the Newton's formula of motion where x, v, a are position, velocity and acceleration respectively.

$$v_1 = a \times t + v_0 \tag{A.1}$$

$$x_1 = \frac{1}{2} \times a \times t^2 + v1 \times t + x_0 \tag{A.2}$$

We can derive the required time for full stop by replacing $v_1 = 0$ and $a = -B$ in Eq. A.1. Therefore:

$$t = \frac{v_0}{B} \tag{A.3}$$

By replacing Eq. A.3 in to Eq. A.2 and also considering $v_1 = 0$ and $a = -B$, we can derive the stopping distance with respect to the velocity and deceleration as follows:

$$\Delta x = \frac{v^2}{2 \times B} \tag{A.4}$$

Finally, we can conclude that the minimum stopping distance for the host vehicle to fully stop by the rear end of the leader vehicle is the difference between their stopping distances as in Eq. A.5.

$$sc_{gap}(v_l, v_h) = \frac{v_h^2 - v_l^2}{2 \times B} \tag{A.5}$$

A.2 Calculating the margin for Safety Critical Distance

The following discussion explains how the $margin_{sc_{gap}}(v_h)$ can be obtained in section 3.1. Suppose that the host vehicle is traveling with max acceleration (A_{max}) when the ACC$^+$ system requests the maximum negative acceleration B. If we consider the delay of the system as ϵ then the host vehicle will continue to accelerate at A_{max}, increasing its velocity v_h for ϵ seconds. Consequently, the host vehicle's velocity will increase as in Eq. A.6.

$$v_{h_{new}} = A_{max} \times \epsilon + v_h \tag{A.6}$$

Also, the distance that the host vehicle will travel for ϵ seconds with acceleration A_{max} will be derived from the following formula (Eq. A.7):

$$\Delta x_1 = \frac{1}{2} \times A_{max} \times \epsilon^2 + v_h \times \epsilon \tag{A.7}$$

However, the maximum negative acceleration B is requested; hence, acceleration $-B$ should be applied after ϵ seconds to return the host vehicle to its initial velocity, v_h. The required distance can be interpreted from similar steps as Appendix A.1. Therefore, the distance will be:

$$\Delta x_2 = \frac{v_{h_{new}}^2 - v_h^2}{2 \times B} \tag{A.8}$$

By replacing $v_{h_{new}}$ from Eq. A.6 into Eq. A.8, Δx_2 will be:

$$\Delta x_2 = \frac{A_{max}^2 \times \epsilon^2 + 2 \times A_{max} \times \epsilon \times v_h}{2 \times B} \tag{A.9}$$

Finally, $margin_{sc_{gap}}(v_h)$ is the total of the two distances $(\Delta x_1 + \Delta x_2)$:

$$margin_{sc_{gap}}(v_h) = (\frac{A_{max}}{B} + 1)(\frac{A_{max}}{2} \times \epsilon^2 + \epsilon \times v_h) \tag{A.10}$$

A.3 Calculating the maximum Desired Velocity v_{set} for a Practical ACC$^+$ Design

In this section we provide a calculation of formula 4.4 in section 4.2. The velocity of the host vehicle should be in a practical range such that a realistic sensor can detect the leader vehicle and ACC$^+$ system can take proper action as required in *Follow* and/or *Safety_Critical*.

Assume d_{range} is the maximum range of the sensor for this purpose. If d_{range} is

greater than or equal to the right side of Eq. 4.3 then ACC$^+$ system would be able to take an action under some required circumstances. This fact is in Eq. A.11.

$$d_{range} \geq max(f_{gap}(v_l, v_h, a_h), 0) + margin_{f_{gap}}(v_h, a_h) + (h_{set} \times v_l) \tag{A.11}$$

According to Eq. 4.1, $f_{gap}(v_l, v_h, a_h)$ can be replaced in Eq. A.11 considering a worst case scenario that the host vehicle's velocity is v_{set} ($v_h = v_{set}$) and zero velocity of lead vehicle ($v_l = 0$). Also, $margin_{f_{gap}}(v_h, a_h)$ is considered to be a fixed value in order to simplify the calculations. Therefore, Eq. A.11 can be rewritten as follows.

$$d_{range} \geq \frac{v_{set}^2}{-2 \times a_h} + margin_{f_{gap}}(v_{set}, a_h) \tag{A.12}$$

Solving the above non-equality (Eq. A.11) with respect to v_{set} with a realistic comfortable deceleration a_h as 30% of maximum deceleration B ($a_h = -0.3B$) leads to the following upper bound for v_{set} (Eq. A.13).

$$v_{set} \leq \sqrt{2 \times 0.3B \times (d_{range} - margin_{f_{gap}}(v_{set}, -0.3B)} \tag{A.13}$$

Bibliography

Abrahám-Mumm, E., Hannemann, U., and Steffen, M. (2001). Verification of hybrid systems: Formalization and proof rules in PVS. In *Engineering of Complex Computer Systems, 2001. Proceedings. Seventh IEEE International Conference on*, pages 48–57. IEEE.

Aréchiga, N., Loos, S. M., Platzer, A., and Krogh, B. H. (2012). Using theorem provers to guarantee closed-loop system properties. In *American Control Conference (ACC), 2012*, pages 3573–3580. IEEE.

Arioui, H., Hima, S., and Nehaoua, L. (2009). 2 DOF Low Cost Platform for Driving Simulator: Modeling and Control. In *Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on*, pages 1206–1211.

Bin, Y., Li, K., Lian, X., Ukawa, H., Handa, M., and Idonuma, H. (2004). Longitudinal acceleration tracking control of vehicular stop-and-go cruise control system. In *Networking, Sensing and Control, 2004 IEEE International Conference on*, volume 1, pages 607–612. IEEE.

Bowen, J. and Stavridou, V. (1993). Safety-critical systems, formal methods and standards. *Software Engineering Journal*, **8**(4), 189–209.

Breimer, B. (2013). *Design of an Adaptive Cruise Control Model for Hybrid Systems Fault Diagnosis*. Master's thesis, McMaster University.

Ciobanu, G. and Rusu, S. (2008). Verifying adaptive cruise control by π-calculus and mobility workbench. Technical report.

Damm, W., Ihlemann, C., and Sofronie-Stokkermans, V. (2011). Decidability and complexity for the verification of safety properties of reasonable linear hybrid automata. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*, pages 73–82. ACM.

Dew, M. (2002). *Coordinated Adaptive Cruise Control: Design and Simulation*. Ph.D. thesis, University of California, Berkeley.

Doyle, J. C., Francis, B. A., and Tannenbaum, A. (1992). *Feedback control theory*, volume 1. Macmillan Publishing Company New York.

Eizad, Z. and Vlacic, L. (2004). A control algorithm and vehicle model for stop & go cruise control. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 401–406. IEEE.

Gerdes, J. C. and Hedrick, J. K. (1997). Vehicle speed and spacing control via coordinated throttle and brake actuation. *Control Engineering Practice*, **5**(11), 1607–1614.

Gietelink, O., Ploeg, J., Schutter, B. D., and Verhaegen, M. (2009). Development of a driver information and warning system with vehicle hardware-in-the-loop simulations. *Mechatronics*, **19**(7), 1091 – 1104. Special Issue on Hardware-in-the-loop simulation.

Girard, A. R., Spry, S., and Hendrick, K. (2005). Intelligent cruise control applications: real-time embedded hybrid control software. *Robotics Automation Magazine, IEEE*, **12**(1), 22 – 28.

Hedrick, J. K. and Yip, P. (2000). Multiple sliding surface control: Theory and application. *Journal of Dynamic Systems, Measurement, and Control*, **122**(4), 586–593.

Hoberock, L. (1976). A survey of longitudinal acceleration comfort studies in ground transportation vehicles. Technical report.

Hoedemaeker, M. (2000). Driving behaviour with ACC and the acceptance by individual drivers. In *Intelligent Transportation Systems, 2000. Proceedings. 2000 IEEE*, pages 506–509. IEEE.

Jairam, S., Lata, K., Roy, S., and Bhat, N. (2008). Verification of a MEMS based adaptive cruise control system using simulation and semi-formal approaches. In *Electronics, Circuits and Systems, 2008. ICECS 2008. 15th IEEE International Conference on*, pages 910–913.

Jerath, K. and Brennan, S. (2010). Adaptive cruise control: Towards higher traffic flows, at the cost of increased susceptibility to congestion. In *Proceedings of AVEC*, volume 10.

Johnson, T. T., Green, J., Mitra, S., Dudley, R., and Erwin, R. S. (2012). Satellite rendezvous and conjunction avoidance: Case studies in verification of nonlinear hybrid systems. In *FM 2012: Formal Methods*, pages 252–266. Springer.

Junaid, K. M., Shuning, W., Usman, K., and Naveed, R. (2005). LQR autonomous longitudinal cruise control with a minimum state observer. In *Proceedings of the Eighth IASTED International Conference: Intelligent Systems and Control*.

Kesting, A., Treiber, M., Schönhof, M., and Helbing, D. (2007). Extending adaptive cruise control to adaptive driving strategies. *Transportation Research Record: Journal of the Transportation Research Board*, **2000**(1), 16–24.

Kural, E. and B.A., G. (2010). Model predictive adaptive cruise control. In *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pages 1455 –1461.

Li, S., Li, K., Rajamani, R., and Wang, J. (2011). Model predictive multi-objective vehicular adaptive cruise control. *Control Systems Technology, IEEE Transactions on*, **19**(3), 556 –566.

Loos, S., Witmer, D., Steenkiste, P., and Platzer, A. (2013). Efficiency analysis of formally verified adaptive cruise controllers. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 1565–1570.

Loos, S. M., Platzer, A., and Nistor, L. (2011). Adaptive cruise control: hybrid, distributed, and now formally verified. In *Proceedings of the 17th international conference on Formal methods*, FM'11, pages 42–56, Berlin, Heidelberg. Springer-Verlag.

Lu, X.-Y., Tan, H.-S., Shladover, S. E., and Hedrick, J. K. (2001). Nonlinear longitudinal controller implementation and comparison for automated cars. *Journal of dynamic systems, measurement and control. Vol. 123, no. 2*.

Martinez, J.-J. and Canudas-de Wit, C. (2007). A safe longitudinal control for adaptive cruise control and stop-and-go scenarios. *Control Systems Technology, IEEE Transactions on*, **15**(2), 246–258.

Mitsch, S., Quesel, J.-D., and Platzer, A. (2014). Refactoring, refinement, and reasoning. In *FM 2014: Formal Methods*, pages 481–496. Springer.

Mohammadi, R. (2009). *Fault diagnosis of hybrid systems with applications to gas turbine engines*. Ph.D. thesis, Concordia University.

Naranjo, J. E., González, C., García, R., and De Pedro, T. (2006). ACC$^+$ Stop & Go maneuvers with throttle and brake fuzzy control. *Intelligent Transportation Systems, IEEE Transactions on*, **7**(2), 213–225.

Naus, G., Van den Bleek, R., Ploeg, J., Scheepers, B., van de Molengraft, R., and Steinbuch, M. (2008). Explicit MPC design and performance evaluation of an ACC stop-&-go. In *American Control Conference, 2008*, pages 224–229. IEEE.

Nehaoua, L., Mohellebi, H., Amouri, A., Arioui, H., Espie, S., and Kheddar, A. (2008). Design and control of a small-clearance driving simulator. *Vehicular Technology, IEEE Transactions on*, **57**(2), 736–746.

Parnas, D. L., van Schouwen, A. J., and Kwan, S. P. (1990). Evaluation of safety-critical software. *Communications of the ACM*, **33**(6), 636–648.

Persson, M., Botling, F., Hesslow, E., and Johansson, R. (1999). Stop and go controller for adaptive cruise control. In *Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on*, volume 2, pages 1692–1697. IEEE.

Peters, G. A. and Peters, B. J. (2003). *Automotive Vehicle Safety*. CRC Press.

Platzer, A. (2007). Differential dynamic logic for verifying parametric hybrid systems. In *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 216–232. Springer.

Platzer, A. (2008). Differential dynamic logic for hybrid systems. *Journal of Automated Reasoning*, **41**(2), 143–189.

Platzer, A. (2010). *Logical Analysis of Hybrid Systems - Proving Theorems for Complex Dynamics*. Springer.

Platzer, A. (2012). The complete proof theory of hybrid systems. In *Logic in Computer Science (LICS), 2012 27th Annual IEEE Symposium on*, pages 541–550. IEEE.

Platzer, A. and Quesel, J.-D. (2008). Keymaera: A hybrid theorem prover for hybrid systems (system description). In *Automated Reasoning*, pages 171–178. Springer.

Santhanakrishnan, K. and Rajamani, R. (2003). On spacing policies for highway vehicle automation. *Intelligent Transportation Systems, IEEE Transactions on*, **4**(4), 198–204.

Sathiyan, S. P., Kumar, S. S., and Selvakumar, A. I. (2013). A comprehensive review on cruise control for intelligent vehicles. *International Journal of Innovative Technology and Exploring Engineering (IJITEE) Volume-2, Issue-5*.

Shakouri, P. and Ordys, A. (2011). Application of the state-dependent nonlinear model predictive control in adaptive cruise control system. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 686 –691.

Shigeharu, M., Takashi, N., Sei, K., Tomoji, I., Hisayoshi, N., Akira, Y., Hitomi, N., and Shin, T. (2010). Improvement of adaptive cruise control performance. *EURASIP Journal on Advances in Signal Processing*, **2010**.

Stursberg, O., Fehnker, A., Han, Z., and Krogh, B. H. (2004). Verification of a cruise control system using counterexample-guided search. *Control Engineering Practice*, **12**(10), 1269–1278.

Tabuada, P. (2009). *Verification and control of hybrid systems: a symbolic approach*. Springer.

Tordeux, A., Lassarre, S., and Roussignol, M. (2010). An adaptive time gap car-following model. *Transportation research part B: methodological*, **44**(8), 1115–1131.

Vahidi, A. and Eskandarian, A. (2003). Research advances in intelligent collision avoidance and adaptive cruise control. *Intelligent Transportation Systems, IEEE Transactions on*, **4**(3), 143–153.

Verburg, D., Van der Knaap, A., and Ploeg, J. (2002). Vehil: Developing and testing intelligent vehicles. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 537–544 vol.2.

Villagra, J., D'Andrea-Novel, B., Choi, S., Fliess, M., and Mounier, H. (2009). Robust stop-and-go control strategy: an algebraic approach for non-linear estimation and control. *International Journal of Vehicle Autonomous Systems*, **7**(3), 270–291.

Xiao, L. and Gao, F. (2010). A comprehensive review of the development of adaptive cruise control systems. *Vehicle System Dynamics*, **48**(10), 1167–1192.

Yamamura, Y., Tabe, M., Kanehira, M., and Murakami, T. (2001). Development of an adaptive cruise control system with stop-and-go capability. Technical report.

Yi, K. and Chung, J. (2001). Nonlinear brake control for vehicle CW/CA systems. *Mechatronics, IEEE/ASME Transactions on*, **6**(1), 17–25.

Yi, K., Hong, J., and Kwon, Y. (2001). A vehicle control algorithm for stop-and-go cruise control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, **215**(10), 1099–1115.

Zaloshnja, E., Miller, T., Council, F., and Persaud, B. (2004). Comprehensive and human capital crash costs by maximum police-reported injury severity within selected crash types. In *Annual proceedings / Association for the Advancement of Automotive Medicine. Association for the Advancement of Automotive Medicine*, volume 48, pages 251–263.

Zhao, X. and Gao, Z. (2005). A new car-following model: full velocity and acceleration difference model. *The European Physical Journal B-Condensed Matter and Complex Systems*, **47**(1), 145–150.