Graduation Assignment

Advanced cruise control development and validation for in-wheel motor-based powertrain

e-Traction

Systems and Software Engineering department

Author: Kevin MBENZA MBUAMBUA

Apeldoorn, Netherlands, 30. 08. 2018

Czech Technical University in Prague
Faculty of Mechanical Engineering
Department of Automotive, Combustion Engine, and Railway Engineering

Diploma work:

Advanced cruise control development and validation for in-wheel motor-based powertrain

Master:

Master of Automotive Engineering

Author:

Kevin MBENZA MBUAMBUA

Supervisor (Czech Technical University):	Supervisor (e-Traction):
Rastislav TOMAN	Geert KWINTENBERG

Abstract

When driving a vehicle, we must pay attention to our surrounding environment, such as the state of our vehicle, the vehicles nearby, pedestrians, and drivers' behavior. Because, these factors are one of the main causes of road accidents. Aware of these, the automotive industry has introduced, and developed embedded systems called Advanced Driver Assistance Systems (ADAS), to improve driving safety and convenience, and eco – driving. The most commonly used ADAS systems are the Airbag, Anti -Lock Braking System (ABS), and Cruise Control (CC), which is the topic of this master's thesis.

Between 1935 and 1945, the American engineer Ralph Teetor invented the Cruise Control (CC). It is the most common ADAS system used in modern vehicles for driver comfort, and long – distance travels. It works as an automatic speed control for vehicles. It maintains a desired speed throughout a drive cycle, reduces driver tasks, avoids violation of speed limit, and thus, improves driving comfort and safety. A variant of the CC called Adaptive Cruise Control (ACC), adjusts the vehicle speed to maintain a certain distance from the vehicle ahead. However, this variant is not the research topic of this master's thesis.

The CC shall maintain or get a desired speed set by the driver. To achieve this target, it shall produce the desired torque output to maintain the set speed with minimum amount of speed overshoot or undershoot for the supported vehicle types. However, the climbing, the rolling, the drag, and other disturbances impact the vehicle speed. Therefore, these external disturbances could influence the performance of the CC. Therefore, when designing a CC, a method to cancel the effect of these external disturbances should be considered.

The aim of this master's thesis is to develop, model, and verify an Advanced Cruise Control software module, for an in – wheel motor – based powertrain of a city bus and delivery truck. This software shall deliver a torque request to maintain a set speed.

The involved master's thesis uses a Model Based Design (MBD) methodology, and MATLAB/Simulink software.

This master's thesis deals with longitudinal vehicle dynamics control, and online parameters estimation, involving the Extended Kalman Filter (EKF), and Luenberger Observer (LO) to estimate the vehicle mass for one, and the road grade for the other. This online parameter estimation is used to develop a control strategy that can handle external parameters variation.

The performance of the controller and the estimators have been tested and validated by applying the Unit Testing (UT), and the Model in the Loop (MIL). The controller has been tested by performing test cases to verify the requirements set to design the controller. For each test cases, the results have shown that the controller follows the set speed within a range of 1-2 % error using the estimated vehicle mass, and road grade. According to the results, the EKF estimates the vehicle mass within a range of 5-10 % error. Regarding the LO observer, the observer reaches and tracks the true slope with an acceptable accuracy.

For both estimators, a logic has been implemented to run them under some conditions. When these conditions are not satisfied, the estimators stop to estimate the parameters. For the LO observer, the observer runs when the brake is not applied, and the velocity is above a threshold value. It has been seen that when the observer is paused, and then, restarts the estimation from the previous estimates, there were some overshoot that led to a non-smooth signal of the estimated road grade. This is because, the parameters used by the observer has been tuned, so that, the observer converges to the true road grade as quickly as possible. Moreover, it has been seen that the observer does not perform well for high variation of torque.

Usually, when designing an estimator, the performance of the estimator should be tested by applying a Root Mean Square Error (RMSE). However, due to the limited time, this method has not been applied. It is therefore left as a recommendation. Although, the controller performs well using estimated parameters, the estimation of the road grade could be improved, to achieve better performance regarding the controller.

Keywords: *Extended Kalman Filter, Load Estimation, Kalman Filter, Luenberger Observer, Mass Estimation, Vehicle Cruise Control, Vehicle Speed Control, Slope Estimation.*

Acknowledgment

This master's thesis was carried out at the Systems and Software Engineering department, at e-Traction in Apeldoorn, Netherlands.

First, I would like to express my sincere gratitude to my two supervisors, and the Systems and Software Engineering department team. I thank Geert KWINTENERG who gave me the opportunity to do my master's thesis at e-Traction and apply control engineering to automotive systems. I thank Rastislav TOMAN for his support, and interesting discussions we had over telephone. I thank the Systems and Software Engineering department team including Willem ROOVERS, Gijsbert BLOKHUIS, and Andrei UNGUREANU for their guidance, and help throughout this master's thesis.

Finally, I thank all employee of e-Traction I interacted with. A great company of dedicated and passionate people to whom I wish all the best.

Kevin MBENZA MBUAMBUA
Apeldoorn, Netherlands
30.08.2018

Table of contents

LIST OF ABBREVIATION **1**

LIST OF FIGURES **2**

LIST OF TABLES **4**

1 INTRODUCTION **5**

1.1 e-Traction 5

1.2 Cruise Control (CC) 8

1.3 Problem statement 8

1.4 Requirements 8

1.5 Cruise Control Architecture 10

1.6 Outline 11

2 CRUISE CONTROL DESIGN APPROACH **12**

2.1 Model Based Design Methodology 12

2.2 Cruise Control Approach 13

2.3 Chosen Design 14

3 VEHICLE MODEL AND STATE SPACE CONTROLLER DESIGN **17**

3.1 Vehicle Model 17

3.2 In wheel-motor based driveline 19

3.3 Cruise Control modeling 20

3.4 Summary 31

4 ACTUATOR SATURATION CONTROL 33

4.1 Effects of saturation 33

4.2 Anti – Windup control 34

4.3 Summary 35

5 ONLINE PARAMETERS ESTIMATION 36

5.1 Estimator model 36

5.2 Kalman Filter 37

5.3 Extended Kalman Filter 38

5.4 Luenberger observer 45

5.5 Summary 46

6 SOFTWARE VERIFICATION AND VALIDATION 47

6.1 CC system – Simulink Model 47

6.2 Unit Testing 49

6.3 Model – in – the – Loop (MIL) 55

6.4 Summary 59

7 CONCLUSION AND DISCUSSION 60

REFERENCES 62

List of abbreviation

ACC	Adaptive Cruise Control
ADAS	Advanced Driver Assistance Systems
AW	Anti-Windup
CC	Cruise Control
EKF	Extended Kalman Filter
DNR	Drive Neutral Reverse
MBD	Model Based Design
MIL	Model in the Loop
LC	Lower Level Controller
LO	Luenberger Observer
PID	Proportional Integral Derivative
PCM	Powertrain Control Module
RMSE	Root Mean Square Error
RLS	Recursive Least Square
SAS	Steering Angle Sensor
UC	Upper Level Controller
UT	Unit Testing
VEM	Vehicle Energy Management
ZEV	Zero Emission Vehicle

List of Figures

Figure 1-1. Global architecture of TheMotion 6
Figure 1-2. Block diagram of TheMotion 6
Figure 1-3. Functional design of the cruise control 8
Figure 1-4. Cruise Control overall architecture 10
Figure 2-1. Model Based Design - V Cycle 12
Figure 2-2. Cruise Control Two Level Structure 14
Figure 3-1. Longitudinal forces acting on a vehicle moving on an inclined road 17
Figure 3-2. TheWheel block diagram 19
Figure 3-3. Upper Level Plant 20
Figure 3-4. State feedback control 24
Figure 3-5. State feedback controller with integral control 27
Figure 3-6. Simulation result without integration control Step response of 11 m/s and initial of 10 m/s 27
Figure 3-7. Simulation result with integration control Step response of 11 m/s and initial of 10 m/s 28
Figure 3-8. State feedback controller with observer control 29
Figure 3-9. Simulation result of observer performance
Step response of 14 m/s and initial speed of 10 m/s 30
Figure 4-1. Simulation result of controller without saturation control
Step response of 14 m/s and initial speed of 10 m/s 34
Figure 4-2. Simulation result of controller with saturation control
Step response of 10 m/s and initial speed of 11 m/s 35
Figure 6-1. High level architecture of the CC system in Simulink 48
Figure 6-2. Designed cruise controller in Simulink 48
Figure 6-3. Control torque request subsystem in Simulink 49
Figure 6-4. UC controller subsystem 50
Figure 6-5. UC controller unit test results 51
Figure 6-6. a. EKF estimator modelled in Simulink 51
Figure 6-7. Mass Selection Logic subsystem 52
Figure 6-8. Mass Selection StateFlow principle 52
Figure 6-9. Mass Selection Logic Unit test result 53
Figure 6-10. CC system unit test result. Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9) 54
Figure 6-11. MIL working principle 55
Figure 6-12. a. MIL test results – Test case 1 Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9) 56
Figure 6-12. b. MIL test results – Test case 2 Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9) 56

Figure 6-12. c. MIL test results – Test case 3 Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9) 56
Figure 6-13. a. Simulation results for a vehicle mass of 14024 kg 58
Figure 6-13. b. Simulation results for a vehicle mass of 16000 kg 58

List of Tables

Table 1-1. Functional specifications of the cruise control 9
Table 1-2. Functional specification of e – Traction 10
Table 3-1. Cruise Control Requirements 21
Table 6-1. Cruise Control subsystems 47
Table 6-2. Cruise controller inputs and outputs 49
Table 6-3. UC controller inputs and outputs 50
Table 6-4. Mass Selection Logic inputs and outputs 53
Table 6-5. LO observer inputs and outputs 54
Table 6-6. Cruise Control MIL test cases 56

1 Introduction

The automotive industry has experienced worldwide success since the Roaring Twenties. Despite the various crises, such as the oil crisis of the 1970s, and the automotive industry crisis of 2008-2010, the use of automobiles has kept increasing. In 1970, 30 million cars were produced, and 246 million were registered worldwide. By 2020, approximately 105 million vehicles are expected to be produced, and more than 800 million vehicles could be registered.

Considering these facts, growth in automobile use leads to increase road safety, by establishing speed-policies to reduce the risk of road accidents. Furthermore, the environmental aspects have become an issue. Population growth with its energy demand, exhaustion of fossil energy resources, the greenhouse gases, and air pollution are some of the challenges that will be faced in the coming decades. Aware of these issues, more and more countries have started to set up policies aiming to promote the development of zero emission vehicles (ZEV), and the use of sustainable energy.

The presented challenges have motivated the need to develop vehicles that optimize fuel consumption, provide safe and smooth ride, as well as minimize the impact on the environment. To deal with these issues, ADAS systems have been developed, and introduced in automotive systems to improve road safety, and eco – driving.

1.1 e-Traction

e-Traction is an automotive supplier based in the Netherlands. This company, which employs about 50 people has invented and developed *TheWheel*, a unique in-wheel electric powertrain technology for heavy duty transportation.

By simplifying the powertrain architecture of a conventional vehicle, TheWheel gives a direct drive power and reaches efficiency up to 92 percent, which is more than two time higher than a conventional vehicle, which is approximately 40 percent.

The in-wheel motor based called *TheMotion* (Figure 1-1) consists of:

- TheWheel: Direct drive in-wheel motor system
- TheDrive: High voltage motor drive
- TheControl: Electronic control unit divided in two components:
  Vehicle Energy Management (VEM)
  Powertrain Control Module (PCM)
- TheConnect: Power distribution system

Figure 1-2 shows the domain in which TheMotion is being integrated. The domain illustrates all external components that are relevant for TheMotion system.

Figure 1-3 shows the system architecture of TheMotion system.

This master's thesis is focused on the development of a software component, an Advanced Cruise Control, which will be introduced in the PCM (Figure 1-4). The PCM is the ECU that is responsible to determine the torque command for each wheel hub motor in the system. It interfaces with driver inputs such as the accelerator pedal, Drive Neutral Reverse (DNR) selector, Steering Angle Sensor (SAS), brake pedal, and parking brake by means of CAN bus.

Figure 1-1. Visual appearance of TheMotion

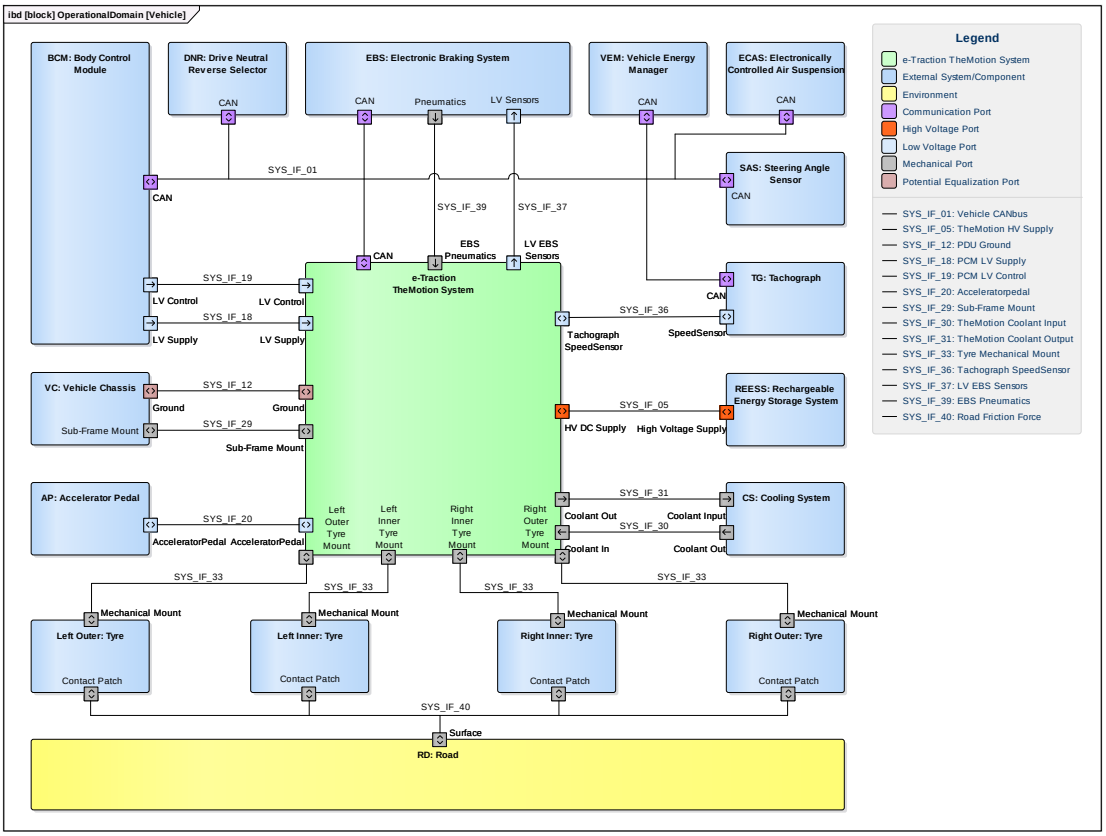

Figure 1-2. Operational domain of TheMotion system

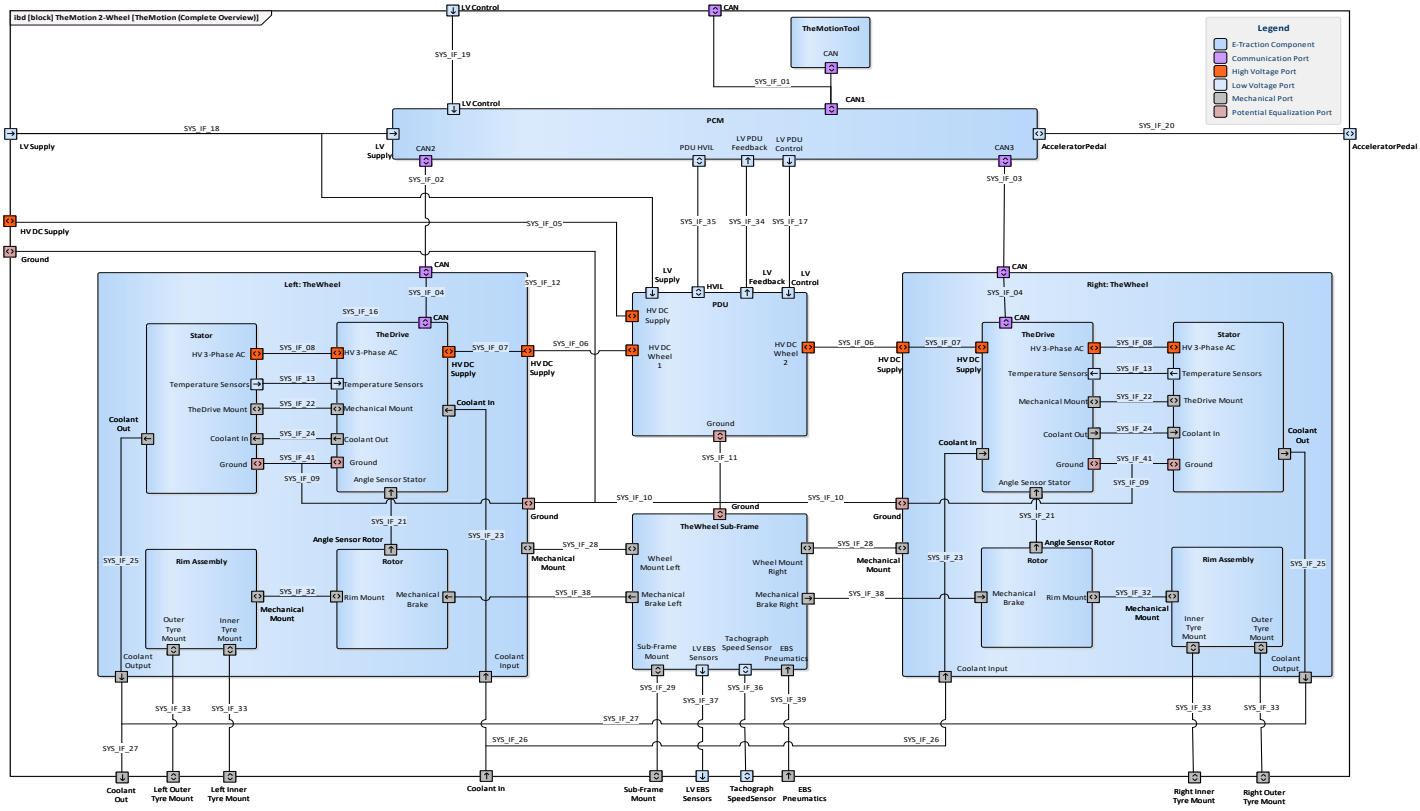

Figure 1-3. System architecture of TheMotion system

Figure 1-4. Visual appearance of the PCM

1.2 Cruise Control (CC)

A CC acts as an automatic speed control for vehicles, it maintains a desired speed throughout a drive cycle. The output of the system, which corresponds to the torque request, is controlled by a controller to maintain the desired speed. In a normal driving scenario, the driver must press the accelerator pedal on an ongoing basis to maintain the vehicle speed. The CC removes this task, which can be uncomfortable and exhausting.

1.3 Problem statement

e-Traction needs a software component that can maintain a set speed for both city buses, and delivery trucks for city distribution. By meeting predefined requirements, the software component shall be able to deliver a torque request to reach or maintain the set speed. Additionally, the software component shall adapt to any change either from the driver request or the surrounding environment.

The aim of this master's thesis is defined as:

Develop, model, and verify a cruise control software module, for an in wheel-motor based powertrain, which delivers a torque request to maintain a desired speed set by a driver

1.4 Requirements

1.4.1 Cruise Control requirements

The functional specifications of the CC in Table 1-1 describe the functional design of the software component (Figure 1-3).

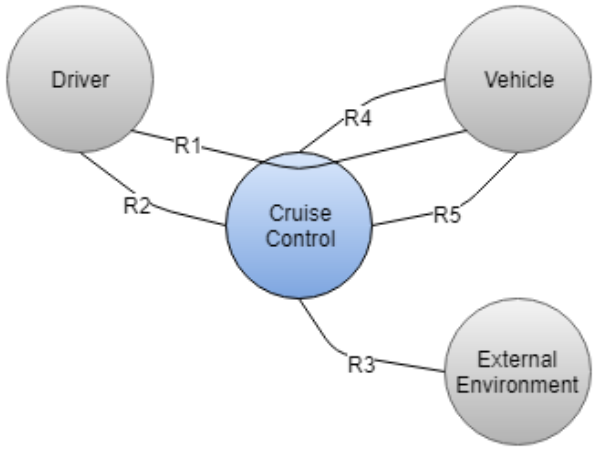

Figure 1-5. Functional design of the cruise control

Requirement ID	Description	Requirements	Requirements Level
R1	Maintain or get the desired speed set by the driver	The controller shall maintain the set speed with a minimum of speed over and undershoot	Percentage overshoot, PO: $PO < 10\ \%$
		The controller shall react at any change from the driver and the external disturbances with a minimum amount of time	Settling time, Ts:: $T_s \leq 5\ s$
R2	It shall be controlled by the driver	The component shall react in a correct way according to the input buttons	SET, RESET, RESUME, ACCELERATE, DECELERATE, PAUSE
R3	It shall be able to adapt to any dynamic and static variation	The controller shall be able to overcome the external disturbances	Cancel the effect of climbing, rolling, and drag resistance
R4	Driving comfort	The operation performed by the controller shall be smooth. Acceleration and deceleration are related to the passenger comfort	Longitudinal Acceleration, a_x: $a_x \in [-2.5\ ;1]$
R5	Regenerative braking	The controller shall use the regenerative braking as an option to maintain the desired speed	

Table 1-1. Functional specifications of the cruise control

The requirements R2 revolves around a logic controller, which determines the correct reaction of the CC according to the driver request. The Systems and Software Engineering department team has already made the controller behind it. Therefore, this part of the CC is not dealt in this master's thesis. However, the designed logic controller should be considered, while the performance of the CC developed in this master's thesis is based on the CC inputs.

1.4.2 e – Traction requirements

After developing and testing the CC in MATLAB/Simulink, the CC Simulink model is integrated into the PCM Simulink model. Then, the CC Simulink model is used to generate a C code from it, as well as tested in real time simulation to demonstrate that the implementation of the designed controller does suit for purpose of being integrated into the PCM. The requirements for this part are summarized in Table 1-2.

Requirement ID	Description
RE1	The cruise control shall conform with the existing software architecture
RE2	The cruise control shall be modeled according to e-Traction modelling guideline. It shall be modelled so that a code can be generated from it
RE3	The generated code shall be executable on the e – Traction PCM platform

Table 1-2. Functional specification of e – Traction

1.5 Cruise Control Architecture

The developed CC is based on the architecture illustrated in Figure 1-4. It consists of:

- A two-level structure represented by the Upper Level Controller (UC controller), and Lower Level Controller (LC controller)
- An online parameters estimation block to estimate the vehicle mass, and the road grade. The feedback of these estimated parameters is then used by the Lower Level Controller, to compute the desired torque.

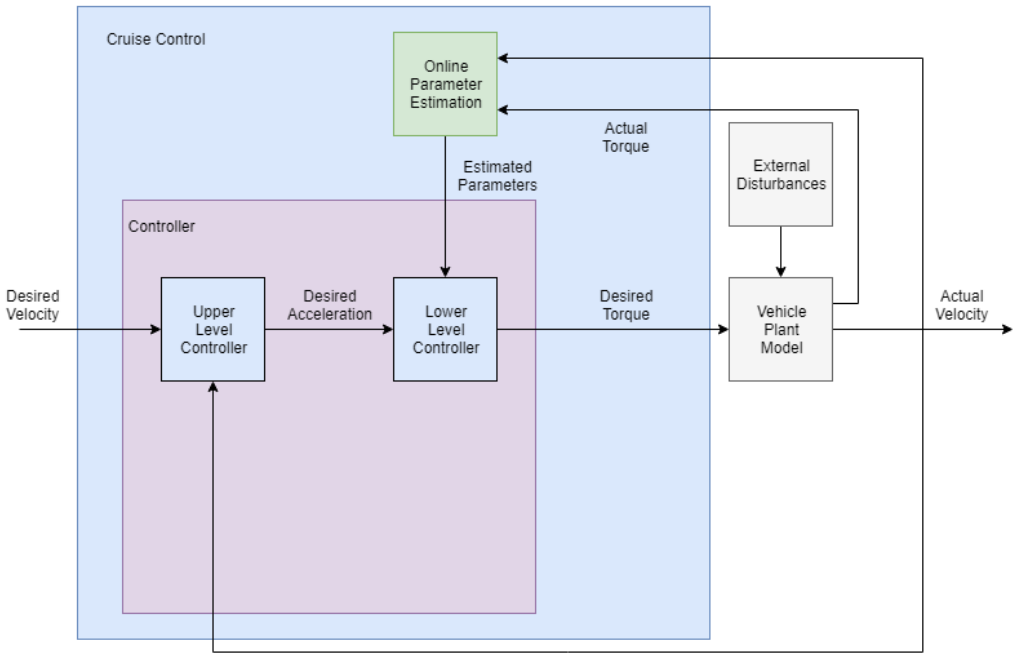

Figure 1-6. Cruise Control overall architecture

1.6 Outline

To respond to the requirements of the CC system, and e – Traction, the master's thesis is organized as follow:

Chapter 2: Cruise Control design approach

This chapter presents the approach used to design the CC system. It deals with the Model Based Design Method to develop a software component, a general structure of the CC system, and literature study related to methods used to compensate the effect of external disturbances.

Chapter 3: Vehicle dynamic model and state space controller

This chapter introduces the longitudinal vehicle dynamic model used for the simulation, and the CC design. The CC design is divided in two parts: the first part refers to the state space controller used to design the upper level controller, and the second part presents the design of the lower level controller.

Chapter 4: Actuator saturation control

The CC modelled in chapter 2 does not take the requirement R4 (Driving comfort) into account. This chapter deals with a saturation control to keep the longitudinal vehicle acceleration in the predefined boundary. Saturation could lead to non-linearity in the controller, and thus, lead to undesirable response from the system. Then, a saturation control strategy is introduced.

Chapter 5: Online parameters estimation

Two observers, Extended Kalman Filter (EKF), and Luenberger Observer (LO) are introduced. The first one is used to estimate the mass of the vehicle, and the second one the road grade. These estimated parameters are used to cancel the effects of the climbing, and rolling resistance.

Chapter 6: Software verification and validation

This chapter refers to the Unit Testing (UT), and Model in the Loop testing (MIL) applied to the designed CC. The UT testing is applied to the CC Simulink model using Simulink Test. This test verifies whether the controller behaves as expected. Then, the MIL testing is applied using Simulink. The PCM Simulink model runs in a Loop with other components to verify the accuracy of the designed CC. This final step of this master's thesis validates the implementation of the controller for real time application.

Chapter 7: Conclusion and discussion

This chapter summarizes all steps that have been performed during this master's thesis, and explores other approaches that could be done to improve the performance of the controller.

2 Cruise Control design approach

2.1 Model Based Design Methodology

Model Based Design (MBD) is a method to develop a control system using MATLAB/Simulink.

Thanks to MATLAB/Simulink, it is possible to model, analyze, and control physical systems governed by mathematical models. In addition, it is also possible to use MATLAB/Simulink to test the performance of each software component of a hardware separately in a simulated environment, and therefore, avoid the faults that can be found in a physical hardware.

The V diagram (Figure 2-1) illustrates the MBD approach. The V diagram is divided in two parts: the left side corresponds to the design of the system, and the right side refers to the validation and verification of the system. The right side corresponds to the testing of the generated code from the designed software. It indicates whether the design software is suitable for the integration into the targeted system.

The Systems and Software Engineering department uses this methodology for the PCM software development.

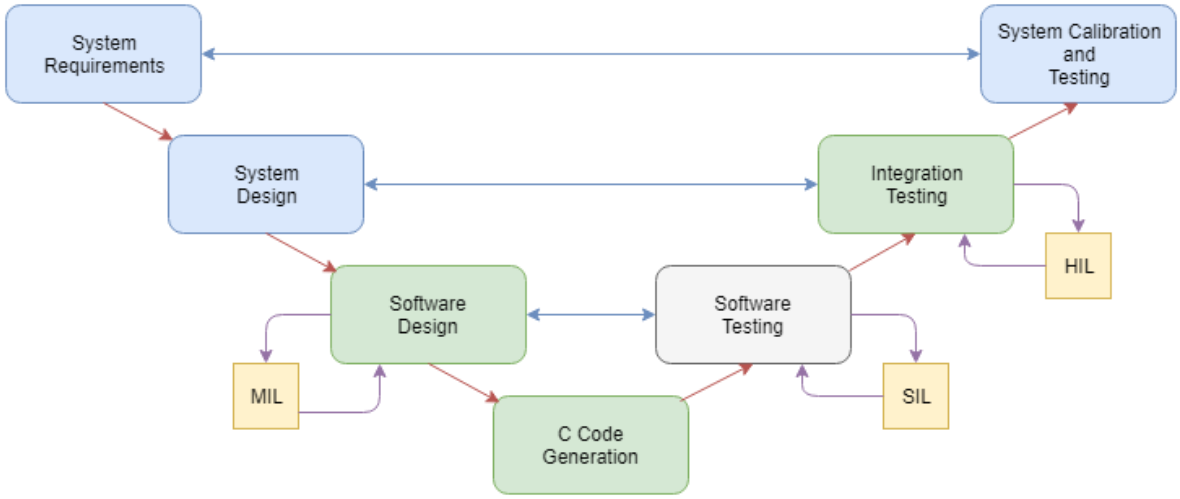

Figure 2-1. Model Based Design - V Cycle

In this master's thesis, the system refers to the PCM (see section 1.1), and the designed software corresponds to the cruise control. Therefore, this master's thesis is focused on the green blocks illustrated in Figure 2-1.

The design part (left side) consists of:	
System requirements	Defining the high-level requirements of the system
System Design	Designing a high-level architecture of the system. The inputs, outputs, and software components used by the system are defined.
Software Design	This part involves several steps: – Defining the requirements of the software component – Modeling of the software component in MATLAB/Simulink – Applying the UT testing (Simulink Test) to the software Simulink model, to verify the logic implemented in the software – Applying the MIL testing to the system Simulink model, to verify the accuracy of the software component
C Code Generation	One of the benefits of modelling a system in Simulink is that, it is possible to generate a code from the Simulink model, and therefore, avoid the handwritten code. The generated code is used throughout the right side of the V diagram.
The validation and verification part (right side) consists of:	
Software Testing	Checking the behavior of the generated code in a simulated environment without any other software (SIL test). This step is not performed for the PCM software development
Integration Testing	The software is integrated into the targeted hardware. To verify whether the software component works correctly depending on the other components of the hardware, the HIL test is applied.
System Testing	This part corresponds to the high-level test, where the hardware (or the physical system) interacts with external components. This final stage indicates whether the designed system work as expected according to the predefined requirements (System requirements)

Due to the time limit, the Integration Testing has not been performed.

2.2 Cruise Control Approach

In control theory, a plant is the system to be controlled. In this master's thesis, the plant refers to the vehicle.

When designing a controller, the controller shall work for linear as nonlinear model of the targeted plant. In [1], this approach is used to design a CC. For the linear model of the plant, the team uses three controllers: PID controller, State Space controller, and Fuzzy controller.

According to the results, these feedback controllers seem not to be sufficient when applied to the nonlinear model of the plant. Because, when we linearize a nonlinear system, we usually linearize around a certain operating point. Consider the example of the nonlinear function $f(x) = x^2$. We linearize this function around $x = 1$ using Taylor series, $f(x) \approx f(1) + f'(1)(x - 1)$. It gives us $f(x) \approx 2x - 1$. Around $x = 1$, $f(x) \approx 2x - 1$ is a good approximation of $f(x) = x^2$. However, far from $x = 1$, the approximation is not valid anymore. Therefore, if there is a dynamic change of the operating point, one cannot apply this approach to design a controller. Furthermore, according to [2], a simple feedback controller could not provide the appropriate performance, because such a controller is not able to handle the noise from the sensor.

Therefore, due to the non-linearity of the vehicle dynamics model, the driver request, and the variation of external disturbances, one cannot use a simple feedback control to design an effective CC system. Moreover, by considering the architecture of the UC controller from section 1.5, this one does not take the variation of the external disturbances into account. Hence, it could lead to undesirable response from the system.

One of the goals of this master's thesis is to design a third controller, which will be able to compensate the external disturbances, and thus, improve the performance of the designed controller. This method is called feedforward control.

2.3 Chosen Design

Research regarding the CC system have led to use a two-level structure model based on [3].

- The upper level controller (UC controller) computes the desired acceleration based on the desired and actual vehicle velocity.

- The lower level controller (LC controller) computes the desired torque to reach the desired acceleration from the UC Controller. It uses vehicle dynamics model, propulsion system maps, and nonlinear control to calculate in real-time the torque request to track the desired acceleration.

Figure 2-2 illustrates the two-level structure model.

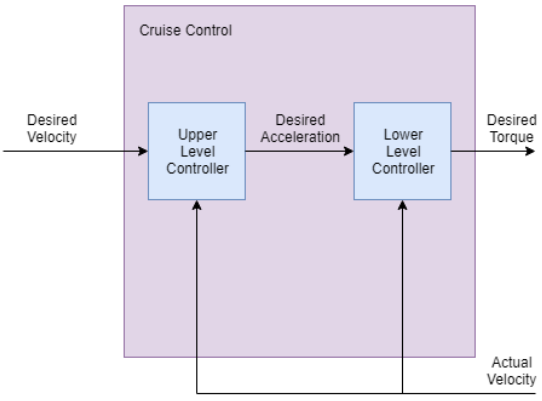

Figure 2-2. Cruise Control Two Level Structure

This hierarchical architecture of the CC is used to compensate the nonlinear vehicle dynamics model, and have more flexibility regarding the control of the vehicle acceleration.

As the LC Controller uses vehicle dynamics model, and propulsion system maps, the design of the CC system is focused on the development of the UC controller.

During this master's thesis, a PID controller was initially used to design the UC controller. But due to the tuning of the PID parameters, which depend on one another regarding the CC response, we choose a State Space controller to model the UC Controller. We assume a longitudinal vehicle model, and neglect lateral model.

The longitudinal vehicle dynamics model uses the Newton's Second Law, which describes the longitudinal dynamics of a body. Besides, the Newton's Second Law is also applicable for rotational bodies, such as the tires. By applying the Newton's Second Law to the vehicle, we know that the longitudinal vehicle dynamics are influenced by the traction force (or tire force), the climbing, the rolling, and the drag resistance. Amongst these forces, the traction force (or the torque request) is computed by the LC controller.

In general, the traction force depends on some variables, such as the slip ratio, the normal load, and the friction coefficient from the Coulomb's law of friction. However, when designing a cruise control, we assume a steady state velocity, and a longitudinal vehicle model. These assumptions imply a zero-slip between the tire and the road, which means that the vehicle body and the tires have the same longitudinal speed.

The resistance (or disturbance) forces, including the mass, the slope, and the drag should be compensated. However, such disturbances are not constant. The slope depends on the topology of the road, the vehicle mass depends on the load, and the drag depends on the vehicle speed. The controller could use these parameters to compute the desired torque to track the desired acceleration, and overcome these disturbances. However, devices to measure these parameters are expensive, and can increase the complexity of TheMotion system. It is then necessary to develop virtual sensors, which are able to estimate in real time these parameters.

There are methods to estimate the mass, and the road grade. Based on the actual velocity and actual torque of the vehicle, it is possible to estimate these parameters. In [4], a Recursive Least Squares method (RLS) is used to estimate simultaneously the mass, and the road grade. While in [5], the same tool is used to estimate the mass, the road grade, and the drag coefficient. Another approach uses an Extended Kalman Filter (EKF) [6] to estimate the mass and the road grade. This approach uses a state space representation of a dynamic system. Holm [7] tries to improve the performance of the EKF by adding an extra sensor, an accelerometer, which improves the estimation of the road grade. However, an accelerometer is not available in TheMotion system.

Promising results were obtained with the EKF, while the actual velocity and torque signal were noisy. In [7], the performance of the EKF have been tested with simulated and real data. The results have shown that the EKF converges to the true value of the mass with a minimum amount of time (approximately 50 seconds), and an acceptable accuracy (5 % error). Due to limited time, and better understanding of the EKF, which is based on state space representations, this method has been adopted in this master's thesis. Moreover, to improve the estimation of the road grade, a second estimator, a Luenberger Observer (LO) has been developed.

The LO observer is used for linear systems. Compared with the EKF, which uses probabilistic approach, the LO observer uses time constants to give the uncertainty of the estimated parameters. The estimated mass, and road grade are then used by the LC controller to compute the torque request.

The LC controller uses a plant model represented by a first order system in series with an integrator. The plant input are the desired torque and external disturbances, and the plant output is the actual vehicle speed. The UC controller uses the same plant as the LC controller, but differ from their inputs, the plant input of the UC controller is the desired acceleration. The UC controller uses a State Space controller to compute the desired acceleration.

Such a controller shall respect the requirements defined in section 1.4.1: maintain the desired speed with a minimum of speed over and undershoot, and react at any change either from the driver or the external environment with a minimum amount of time. To respect these requirements, the developed State Space controller uses a pole placement method [8] to calculate the state feedback gain.

3 Vehicle Model and State Space controller design

The first section of this chapter presents the longitudinal vehicle dynamics model. The second part introduces the in-wheel motor-based model of e-Traction. Finally, this chapter concludes with a description of the State Space controller used to model the UC controller.

3.1 Vehicle Model

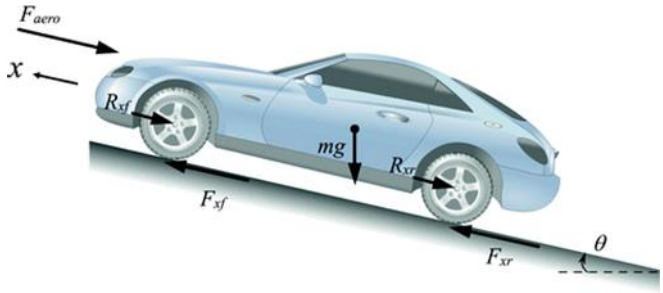

Figure 3-1. Longitudinal forces acting on a vehicle moving on an inclined road

The equation which describes the longitudinal motion of a vehicle is given by

$$m\dot{V}_x = F_{xf} + F_{xr} - F_{aero} - R_{xf} - R_{xr} - mg\sin(\theta) \tag{3.1}$$

where m is the vehicle mass, $\dot{V}_x$ is the longitudinal vehicle acceleration, F_{xf} and F_{xr} are the tire forces at front and rear tires, F_{aero} is the drag resistance, R_{xf} *and* R_{xr} are the rolling resistances at front and rear tires, g is the acceleration due to the gravity, and θ is the road slope. Now, $F_x = F_{xf} + F_{xr}$ and $R_x = R_{xf} + R_{xr}$.

3.1.1 Wheel dynamics

Tire force (Tractive force)

The tire force F_x represents the tractive force that the propulsion system provides, and the force transmitted by the road to the tires.

As stated in section 2.3, F_x depends on some factors, which implies the use of a tire model. Nonetheless, we considered for the design of the cruise control a zero slip between the road and the tire, while the operations performed by the cruise control are smooth. Moreover, TheMotion has a system that detects slip during driving and braking. In this case, the CC is disactivated automatically.

The following equation describes the dynamic of the wheel

$$I_w \dot{\omega}_w = T_w - R_o F_x \tag{3.2}$$

where I_w is the inertia of the wheel, $\dot{\omega}_w$ is the angular acceleration of the wheel, T_w is the torque transmitted to the wheel, and R_o is the radius of the wheel.

3.1.2 External Disturbances

Drag resistance

The drag resistance F_{aero} is given by

$$F_{aero} = \frac{1}{2} \rho_{air} C_d A_F V_x^2 \tag{3.3}$$

where ρ_{air} is the mass density of the air, C_d is the aerodynamic drag coefficient, A_F is the frontal area of the vehicle, and V_x is the longitudinal vehicle velocity.

Rolling resistance

The rolling resistance R_x is given by

$$R_x = f_r mg \cos(\theta) \tag{3.4}$$

where f_r is the friction coefficient, m is the vehicle mass, g is the acceleration due to the gravity, and θ is the road slope defined by

$$\theta = \arctan\left(\frac{road\ grade\ \%}{100}\right) \tag{3.5}$$

Climbing resistance

The climbing resistance represents the longitudinal component of the gravitational force $\overrightarrow{F_g}$ acting on the vehicle.

$$\overrightarrow{F_g}.\vec{x} = mg\sin(\theta) \tag{3.6}$$

3.2 In wheel-motor based driveline

TheWheel consists of TheDrive (Electric motor controller), the propulsion system (Electric motor), and the wheel. This structure of TheWheel gives a direct drive power by simplifying the powertrain system, and the driveline architecture of a conventional vehicle. Figure 3-2 illustrates the powertrain architecture of TheWheel.

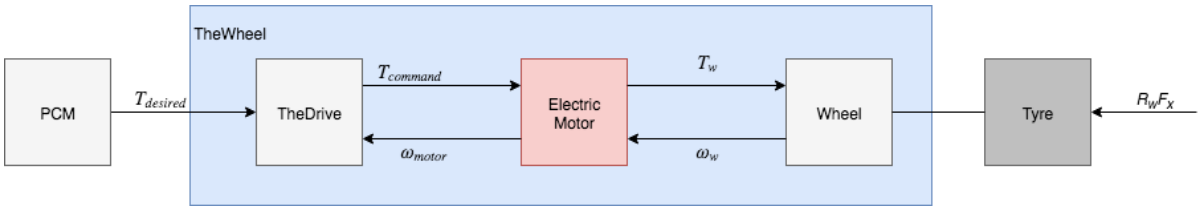

Figure 3-2. TheWheel block diagram

Since the Electric motor is directly link to the wheel, the electric motor speed ω_{motor} is the same as the wheel speed ω_w. The same applied to the torque ($T_{command}$ is equal to the torque transmitted to the wheel T_w).

Based on the current state of the electric motor (angular velocity) and the desired torque $T_{desired}$, TheDrive computes the torque command $T_{command}$ transmitted to the wheel.

Now, we assume that the group TheDrive and Electric Motor is represented by a first order system

$$\tau\frac{dT_w}{dt} + T_w = T_{desired} \tag{3.7}$$

where τ is the time constant of TheDrive – Electric Motor.

3.3 Cruise Control modeling

The purpose of a standard cruise control is to maintain or get the desired speed set by the driver.

3.3.1 Upper Level Controller (UC controller)

To design the UC controller, we model the vehicle plant as a first order system in series with an integrator. Here, the desired longitudinal acceleration is the plant input, and the actual velocity is the plant output. Figure 3-3 illustrates the UC controller and its vehicle plant model.

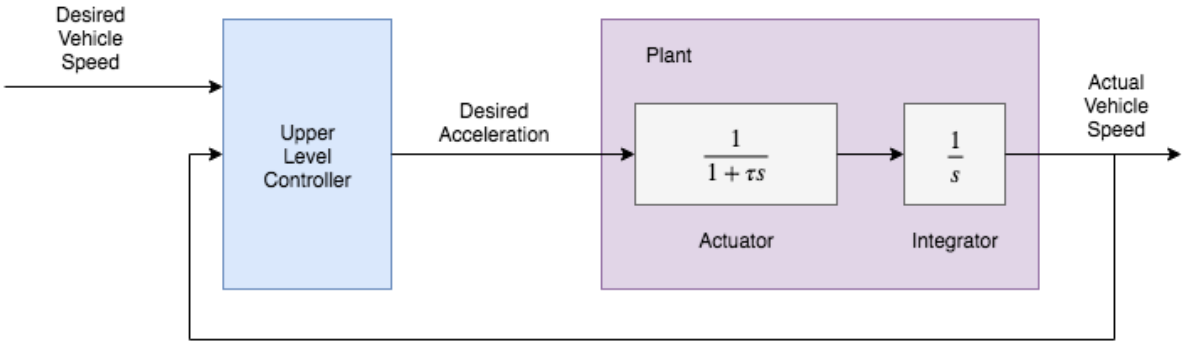

Figure 3-3. Upper Level Plant

In this model, the actuator (TheDrive – Electric Motor system) is governed by

$$\tau \frac{da_x}{dt} + a_x = a_{desired} \tag{3.8}$$

where a_x is the longitudinal vehicle acceleration, and $a_{desired}$ is the desired longitudinal acceleration computed by the UC controller.

Before designing the UC controller, we must consider the requirements defined in section 1.4.1. The requirements are summarized in the Table 3-1.

A State Space controller is proposed to design the UC controller.

Requirement ID	Category	Requirement	Requirement Level
R1	Maintain or get the desired speed set by the driver	The controller shall maintain the speed with a minimum of speed over and undershoot	Percentage overshoot, PO: $PO \leq 10\%$
		The controller shall react at any change either from the driver or the external environment with a minimum amount of time	Settling Time, Ts: $Ts \leq 5\ s$
R4	Driving comfort	The operation performed by the cruise control shall be smooth.	Longitudinal acceleration, a_x: $a_x \in [-2.5\ ;\ 1]$

Table 3-1. Cruise Control Requirements

3.3.1.1 State space controller

The first step to develop the State Space controller, is to find a state space representation of the vehicle.
A state space representation is a method which uses state space variables, to describe a linear (or nonlinear) dynamic system. The state space representation of a linear system is given by

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases} \tag{3.9}$$

where x is the state space vector which contains the state space variable, $\dot{x}$ is the derivative of x ($\dot{x} = \frac{dx}{dt}$), u is the input of the system, y is the output vector, A is the state matrix, B is the input (or control) matrix, C is the output matrix, and D is the feedthrough matrix.

For more information related to state space representations, please refer to [8]

Vehicle state space representation

The relation between the velocity and the acceleration is given by

$$\frac{dV_x}{dt} = a_x \tag{3.10}$$

Using the relation above (3.10), (3.8) becomes

$$\frac{d^2V_x}{dt^2} = -\frac{1}{\tau}\frac{dV_x}{dt} + \frac{a_{desired}}{\tau} \tag{3.11}$$

We come up with a system of linear equation of the longitudinal vehicle motion

$$\begin{cases} \dfrac{dV_x}{dt} = a_x \\ \dfrac{da_x}{dt} = -\dfrac{a_x}{\tau} + \dfrac{a_{desired}}{\tau} \end{cases} \tag{3.12}$$

(3.12) gives a state representation of the vehicle

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{3.13}$$

where $x = [V_x\ a_x]^T$, $u = a_{desired}$, $A = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{\tau} \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ \frac{1}{\tau} \end{bmatrix}$, $and\ C = [1 \quad 0]$.

Discretization

One of the requirements of e-Traction is to design a digital controller, which means that the controller shall run with discrete signals. To do this, we will work with discrete state representation using Euler discretization

$$\dot{x} = \frac{x(k+1) - x(k)}{dt} \tag{3.14}$$

where dt is the sampling time.

In discrete time (3.13) becomes

$$\begin{cases} x(k+1) = A(k)x(k) + B(k)u(k) \\ y(k) = C(k)x(k) \end{cases} \tag{3.15}$$

where $A(k) = I + Adt$, $B(k) = Bdt\ and\ C(k) = C$.

Now, any variable $a(k) = a_k$

Controllability

When designing a State Space controller, we must do a controllability analysis. Controllability analysis for linear system deals with the rank of the controllability matrix $c = [B\ AB\ A^2B \dots A^{n-1}B]$ (resp. $c_k = [B_k\ A_kB_k\ A_k^2B_k \dots A_k^{n-1}B_k]$) for continuous system (resp. discrete system). n is the dimension of the state vector x.

A system is controllable, if and only if, the rank of the controllability matrix c is n.

The discrete controllability matrix of the vehicle is given by

$$c_k = [B_k \;\; A_k B_k] \tag{3.16}$$

Using the expression of A_k and B_k, we obtain:

$$c_k = \begin{bmatrix} 0 & \frac{T_s^2}{\tau} \\ \frac{Ts}{\tau} & \frac{T_s}{\tau} * (1 - \frac{T_s}{\tau}) \end{bmatrix} \tag{3.17}$$

By calculating the determinant of the matrix c_k, we find that $rank(c_k) = n$. Therefore, the system is controllable.

Pole Placement Method

z–Transform

Since we work with discrete time system, the z–transform shall be introduced. The z–transform of a function $f(t)$ is defined by

$$Z\{f(k)\} = F(z) = \sum_{k=1}^{\infty} f(k)(z^{-1})^k \tag{3.18}$$

where $f(k)$ is the discrete version of $f(t)$, and $F(z)$ is the z–transform of $f(k)$.

The z–transform is similar as the Laplace transform for continuous system.

For more information about discrete time systems, please refer to [8].

Pole Placement

Now that we have proved that the system is controllable, we can use the pole placement method. In this section, we assumed that all state variables are measurable, it means that the controller knows all state variables (velocity and acceleration) at each time. However, in reality, TheMotion system does not have an accelerometer. Therefore, an observer will be developed later, to get an estimation of the longitudinal vehicle acceleration. Figure 3-4 gives a scheme of the system with state feedback control (Closed-Loop).

The desired acceleration $a_{desired}$ is given by:

$$a_{desired} = V_{desired} - K_k x_k \tag{3.19}$$

where K_k is the state feedback gain of the State Space controller, and $V_{desired}$ is the desired vehicle speed.

Using the discrete state representation of the vehicle (3.15), and the z – transform (3.18), we can find the characteristic polynomial of the discrete system in closed loop.

$$\frac{Y(z)}{V_{desired}(z)} = C_k(z \cdot I_2 - A_k + B_k K_k)^{-1} B_k \tag{3.20}$$

Where I_2 is the identity matrix.

A_k is a matrix of dimension 2, then, the system is a second order system, and it has two poles.

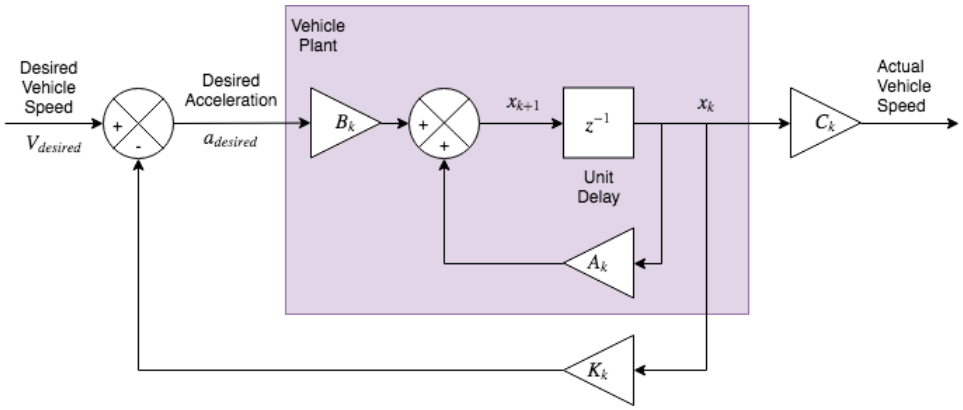

Figure 3-4. State feedback control

Controller Requirement

The transfer function of a second order system in continuous time is given by

$$Y(s) = \frac{k\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{3.21}$$

where k is a gain, ω_n is the natural frequency, ζ is the damping ratio, and s is the Laplace transform variable.

The transfer function can be rewritten as

$$Y(s) = \frac{k\omega_n^2}{(s - s_1)(s - s_2)} \tag{3.22}$$

Where s_1 and s_2 are the pole (or root) of the transfer function defined by

$$s_{1,2} = -\omega_n\zeta \pm j\omega_n\sqrt{1 - \zeta^2} \tag{3.23}$$

There is a relationship between the settling time T_s, the percentage overshoot PO, ω_n, and ζ. The equations that link these variables are given by

$$\omega_n \zeta \geq \frac{4.6}{T_s} \tag{3.24}$$

$$\zeta \geq -\frac{\ln(PO)}{\sqrt{\pi^2 + ln^2(PO)}} \tag{3.25}$$

Moreover, there is also a relationship between discrete poles z_k and continuous poles s_k.

$$z_{1,2} = e^{dt \cdot s_{1,2}} \tag{3.26}$$

where dt is the sampling time.

To meet the requirements R1 and R4, we choose $\zeta = 0.6$, and $\omega_n = 3.6\, rad.s^{-1}$.

Now that we have set the parameters to design the controller, we can calculate the state feedback gain.

If we considered $V_{desired}$ set to zero, the equation of the system in closed – loop (see Figure 3-4) is

$$zI_2 - A_k + B_k K_k = 0 \tag{3.27}$$

where $K_k = [K_1 \quad K_2\ ]$.

To find the state feedback gain, we use

$$det(zI_2 - A_k + B_k K_k) = 0 \tag{3.28}$$

Using the expression of A_k and B_k, (3.28) becomes

$$z^2 + z \cdot \left[\frac{Ts}{\tau}(1 + K_2) - 2\right] - \frac{dt}{\tau}(1 + K_2 - dt \cdot K_1) + 1 = 0 \tag{3.29}$$

We can rewrite (3.29) as

$$z^2 - z \cdot (z_1 + z_2) + z_1 z_2 = 0 \tag{3.30}$$

Finally, we obtain

$$\begin{cases} K_1 = \dfrac{\tau}{dt^2} \cdot [z_1 z_2 + [2 - (z_1 + z_2)] - 1] \\ K_2 = -1 + \dfrac{\tau}{dt}[2 - (z_1 + z_2)] \end{cases} \tag{3.31}$$

With this control, the requirement R1 (maintain or get the desired speed) is not met. There is a steady error due to the reference input (desired speed), because the reference input is not compared with the system output (actual velocity). Figure 3-6 shows the simulation results for a step response of 11 m/s, with an initial speed of 10 m/s.

Integral control

To remove the steady state error due to the step response, we set an integrator in series with the plant. Adding an integrator in series with the plant implies to add another state x_I to the state vector x [8].

x_I is defined as

$$\dot{x}_I = Cx - V_{desired} \leftrightarrow \dot{x}_I = V_{actual} - V_{desired} \tag{3.32}$$

where V_{actual} is the actual vehicle speed.

In discrete time, (3.32) becomes:

$$x_{I_{k+1}} = x_{I_k} + C_k x_k - dt * V_{desired} \tag{3.33}$$

Then we obtain a new state space representation defined as

$$\begin{cases} m_{k+1} = A_m m_k + B_m u_k + B_v V_{desired} \\ y = C_m m_k \end{cases} \tag{3.34}$$

where $m_k = [x_k \; x_{I_k}]^T$, $u_k = a_{desired}$, $A_m = \begin{bmatrix} A_k & 0 \\ dt \cdot C_k & 1 \end{bmatrix}$, $B_m = \begin{bmatrix} B_k \\ 0 \end{bmatrix}$, $B_v = -\begin{bmatrix} 0 \\ 0 \\ dt \end{bmatrix}$,

and $C_m = [C_k \quad 0]$.

Now, we obtain a new expression of the desired acceleration $a_{desired}$

$$a_{desired} = -K' x_k - K_I x_{I_k} \leftrightarrow a_{desired} = -[K'_k \; K_I] * m_k \tag{3.35}$$

Where K'_k is the new state feedback gain, and K_I is the integral gain.

Figure 3-5 gives a scheme of the controller with integral control.

A_m is a matrix of dimension 3, then, the system is a third order system, and it has three poles. For the first two poles, we take the poles used for the controller without integral control. For the third pole, we must choose one that will not influence the desired response. Since we have a third order system, the degree of its characteristic polynomial is 3. Moreover, we know that the requirements R1 should be met with the first two pole. Therefore, we choose $z_3 = \exp[Ts * (-20\omega_n \zeta)]$, which will be faster than the other poles, and thus, will not influence the dynamic response of the system.

To find the gains K_k' and K_I, we use

$$det(zI_3 - A_m + B_m \cdot [K'\ K_I]) = 0 \tag{3.36}$$

Using the same method applied for the controller without integral control, we can find K_k' and K_I.

Figure 3-7 shows the simulation results with integral control for a step response of 11 m/s, with an initial speed of 10 m/s.

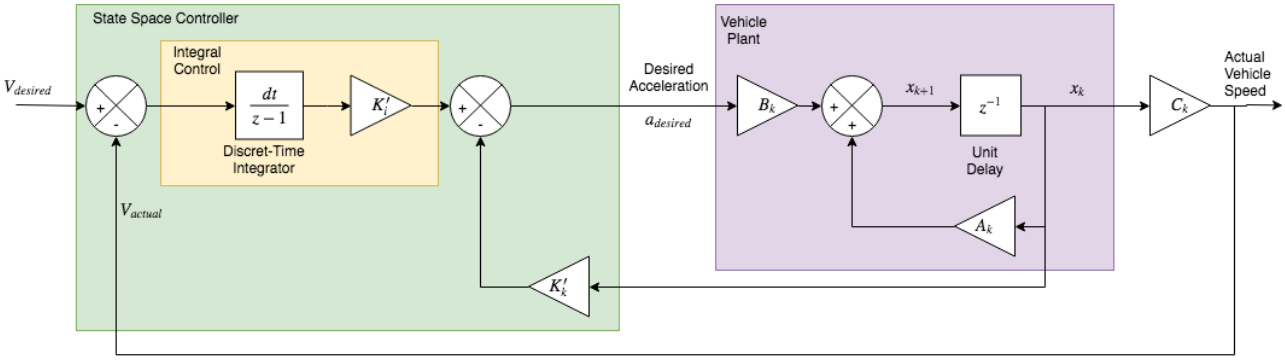

Figure 3-5. State feedback controller with integral control

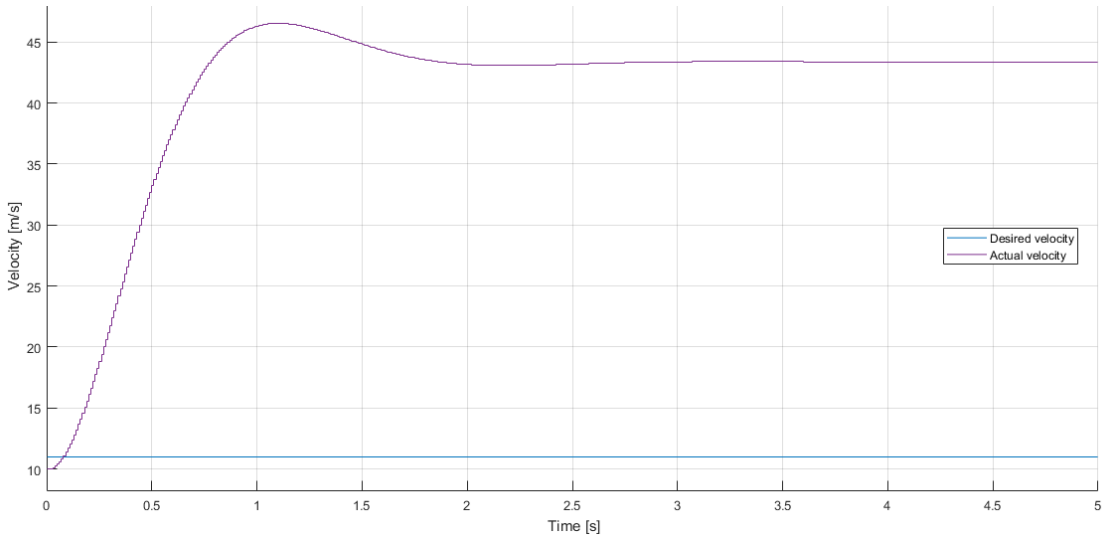

Figure 3-6. Simulation result without integration control
Step response of 11 m/s and initial of 10 m/s

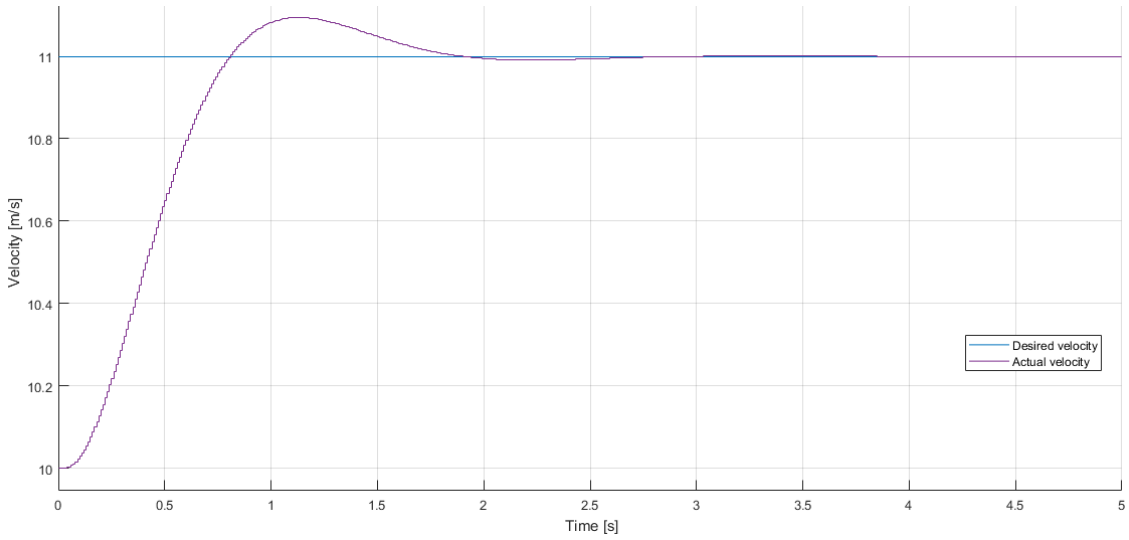

Figure 3-7. Simulation result with integration control
Step response of 11 m/s and initial of 10 m/s

Observer design

The controller designed in the previous section can be used if all state variables are known by the controller. Amongst the two state variables (velocity and acceleration), the velocity signal is available in TheMotion system.

Based on the output (velocity) and input (desired acceleration) of the plant, we can design an observer.
In general, the observer is a perfect copy of the plant without disturbance input. It has the same input and output as the plant. Figure 3-8 shows the overall structure of the system.

The equation of the observer is given by

$$\hat{x}_{k+1} = A_k\hat{x}_k + B_k u_k + L_k(y_k - C_k\hat{x}_k) \tag{3.37}$$

And the dynamic error of the observer is given by

$$e_{k+1} = (A_k - L_k C_k)e_k \tag{3.38}$$

Where $L_k = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix}$ is the observer gain, $\hat{x}_k$ is the estimated state vector, and $e_k = x_k - \hat{x}_k$.

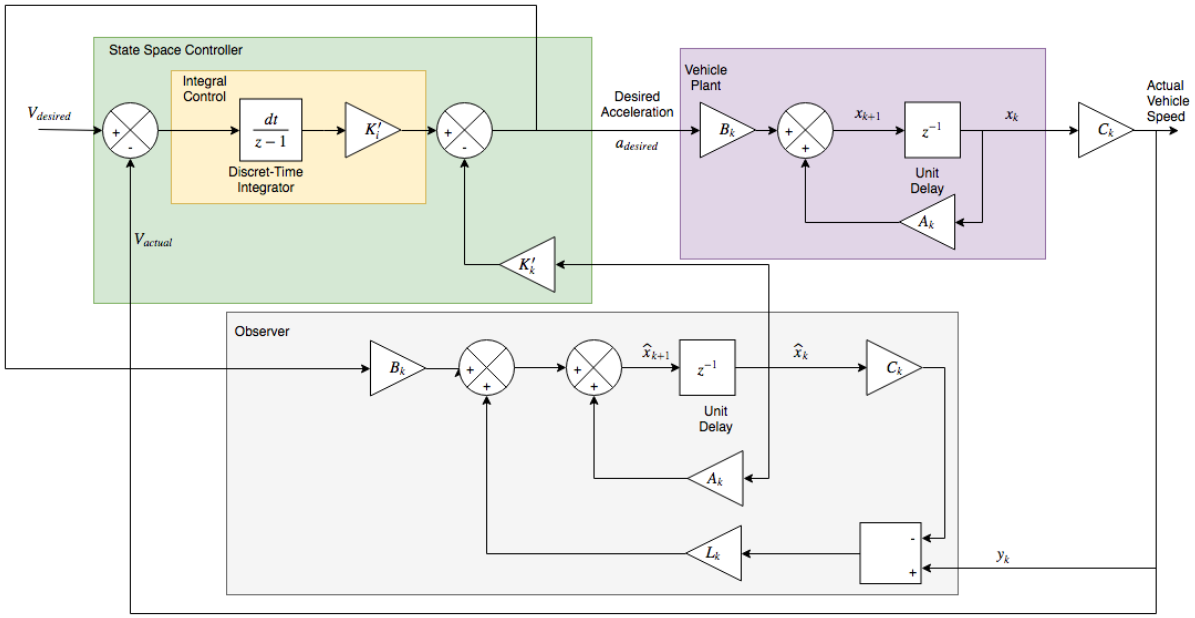

Figure 3-8. State feedback controller with observer control

Observability

As the controllability analysis, the observability analysis must be done to determine whether a system is observable.

Observability analysis for linear system deals with the rank of the observability matrix $o = [C\ CA\ CA^2 \dots CA^{n-1}]^T$ (resp. $o_k = \left[C_k\ C_kA_k\ C_kA_k^2 \dots C_kA_k^{n-1}\right]^T$) for continuous time system (resp. for discrete time system) . n represents the dimension of the state variable x.

A system is observable, if and only if, the rank of the observability matrix o is n.

The discrete observability matrix of the system is given by

$$o_k = \begin{bmatrix} C_k \\ C_kA_k \end{bmatrix} \tag{3.39}$$

Using the expression of $A_k\ and\ C_k$, we obtain

$$o_k = \begin{bmatrix} 1 & 0 \\ 1 & dt \end{bmatrix} \tag{3.40}$$

From (3.40), we deduce that $rank(o_k) = n$. Thus, the system is observable.

Observer gain

To calculate the observer gain L_k, we use

$$det(zI - A_k + L_kC_k) = 0 \tag{3.41}$$

The observer estimates the state variables which keep changing. Therefore, the dynamic of the observer shall be faster than the dynamic of the closed-loop system without the observer.

We therefore choose $L_1 = \exp(-dt * 100)$ and $L_2 = \exp(-dt * 101)$.

By applying the method used to find the gains of the controller (K_k' and K_I), we deduce the observer gain L_k

$$\begin{cases} L_1 = 2 - \left(l_1 + l_2 + \dfrac{dt}{\tau}\right) \\ L_2 = \dfrac{-1 + l_1(1 - l_2) + l_2 - \dfrac{dt}{\tau}\left(l_1 + l_2 - 2 + \dfrac{dt}{\tau}\right)}{dt} \end{cases} \tag{3.42}$$

The observer designed in this section is called a Luenberger Observer.

Figure 3-9 shows the performance of the observer using noisy velocity signal. The observer is able to reach and track the true state, even if the actual velocity signal is noisy.

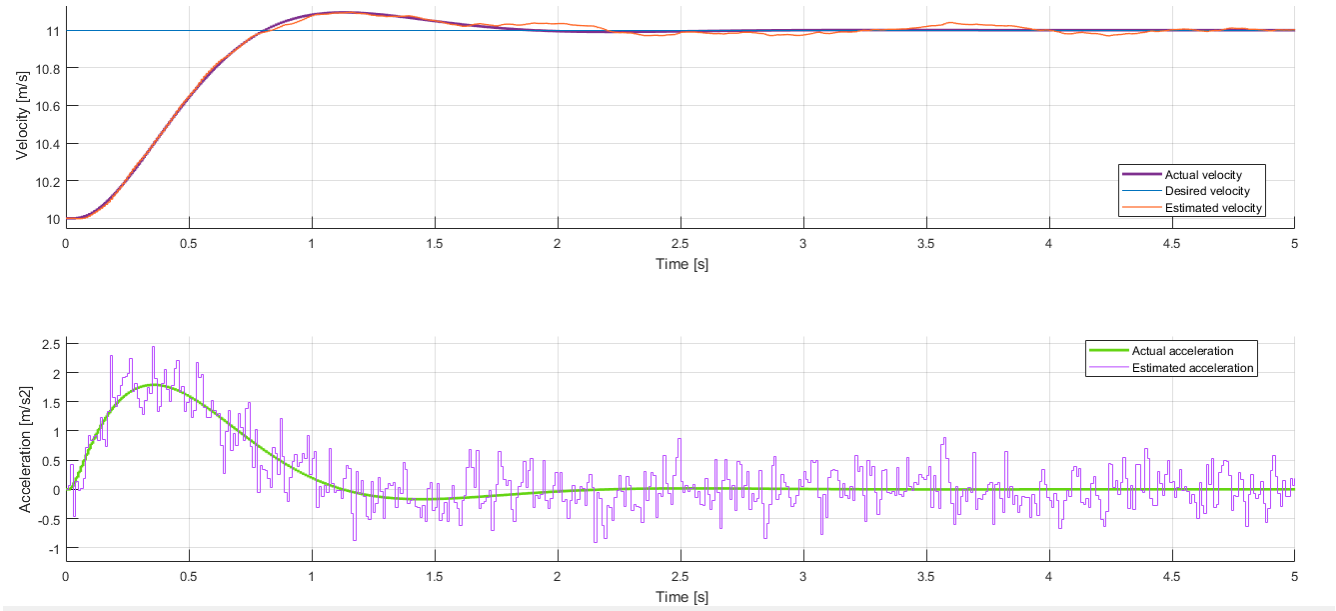

Figure 3-9. Simulation result of observer performance
Step response of 10 m/s and initial speed of 11 m/s

3.3.1.2 Lower Level Controller

After designing the UC controller to get the desired acceleration, we now design the lower level controller (LC controller) to finally compute the desired torque.

The LC controller uses the longitudinal vehicle dynamics model (3.1), and wheel dynamic model (3.2). In the case of a cruise control, we assume that there is zero-slip between the road, and the tires. It implies

$$V_x = R_0 * \omega_w \tag{3.43}$$

where R_0 is the wheel radius.

Using (3.1) and (3.2), we finally deduce the desired torque defined as

$$T_{desired} = R_o * \left[(m + m_r) * a_{desired} + \frac{1}{2}\rho_{air} C_d A_F V_x^2 + f_r mg\cos(\theta) + mg\sin(\theta)\right] \tag{3.44}$$

where $m_r = (I_w + I_m)/R_0^2$, and I_m is the inertia of the motor.

In this formula, two terms should be considered separately, the torque applied to reach the desired acceleration

$$T_a = R_o(m + m_r) * a_{desired} \tag{3.45}$$

and the torque applied to overcome all the disturbances

$$T_d = R_o * \left[\frac{1}{2}\rho_{air} C_d A_F V_x^2 + f_r mg\cos(\theta) + mg\sin(\theta)\right] \tag{3.46}$$

Therefore, the response of the system mainly depends on the torque applied to overcome all the disturbances.

3.4 Summary

In this chapter, a simplified vehicle dynamics model has been introduced to design the cruise control. External disturbances were also studied. Regarding the powertrain system, a simple model was used to represent the behavior of the in – wheel motor based.

A state space controller with integral control has been used to design the UC controller, along with an observer to estimate the state of the vehicle, which are the velocity and the acceleration.

Finally, the LC controller using the desired acceleration from the UC controller, vehicle dynamics model, and the powertrain system features has been designed.

It has led to an empirical formula for the desired torque. It has been seen that this formula depends on the climbing, the rolling, and the drag resistance. In this chapter, it is assumed that these variables are constant. However, this assumption can lead to undesirable response, when these variables change. Thus, later in this master's thesis, a method to compensate these disturbances will be introduced.

4 Actuator Saturation Control

The cruise control developed in chapter 1 does not meet the requirement R4 (Driving comfort). This requirement is related to the desired acceleration, which must be in a predefined boundary. It implies that the desired acceleration should be saturated. When using a saturation function, two cases must be considered: when the desired acceleration is in the predefined boundary, and when the desired acceleration is out of the boundary.

Because, the desired acceleration must be saturated, and no the desired torque, this chapter is focused on the design of a saturation control for the UC controller.

In this chapter, the effect of a saturation function is analyzed. Then, a solution based on a logic control approach is used to overcome the impact of the saturation function on the system response. According to driver's driving experience, an acceleration between $-2{,}5\ m/s^2$ and $1\ m/s^2$ does not affect drivers and passengers comfort [2].

4.1 Effects of saturation

The saturation function for the UC controller is defined as

$$a_{sat} = \begin{cases} a_{max}, & a_{desired} > a_{max} \\ a_{desired}, & a_{min} \leq a_{desired} \leq a_{max} \\ a_{min}, & a_{min} \leq a_{desired} \end{cases} \tag{4.1}$$

When the desired acceleration is in the predefined boundary $[a_{min}; a_{max}]$, the requirement R4 is met. However, when, the step input (desired velocity) increases, the time of response increases as well, because, the desired acceleration is saturated. For a given initial speed, there is a limit value of the step input when the system response becomes unstable.

Out of the boundary, the system works as an open loop system until the desired acceleration (input of the system) returns within the boundary. In this case, because, the time of response increases, the integral control used by the UC controller keeps integrating a large error, and thus, it outputs a large input for the system, which leads to a large desired acceleration. When the error starts to decrease, the output from the integral control is still too large, and then leads to a delay which generates overshoots, and instabilities. This phenomenon is called windup [9]. Figure 4-1 illustrates the windup issue.

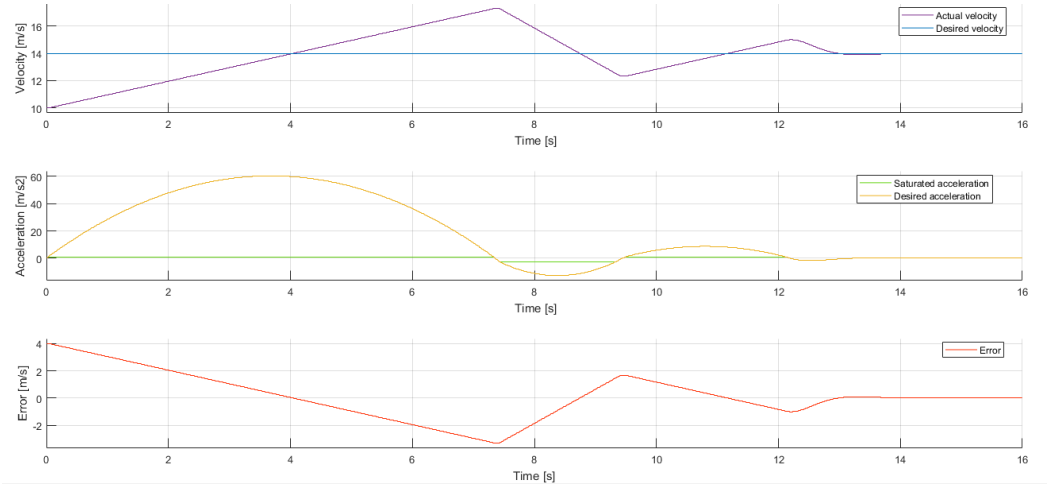

Figure 4-1. Simulation result of the controller without saturation control
Step response of 14 m/s and initial speed of 10 m/s

4.2 Anti – Windup control

As said in section 4.1, when the input saturates, the error keeps increasing. According to this fact, we could use the logic control defined as

$$a_{lim} = \begin{cases} -K_k x_k - K_I x_I, & a_{min} \le a_{desired} \le a_{max} \\ 0, & a_{min} \le a_{desired} \; or \; a_{desired} \le a_{max} \end{cases} \tag{4.2}$$

However, this logic control is not enough to solve the windup issue. This anti-windup method does not take the reset of the integral control into account. In the previous section, we have seen that when the system reaches the desired velocity, the output of the integral is still too large. Therefore, we choose another control logic which deals with the reset of the integral control block

$$a_{lim} = \begin{cases} -K_k x_k - K_I x_I, & a_{min} \le a_{desired} \le a_{max} \\ -K_k x_k, & a_{desired} \ge a_{max} \; and \; x_I \ge 0 \\ -K_k x_k - K_I x_I, & a_{desired} \ge a_{max} \; and \; x_I \le 0 \\ -K_k x_k, & a_{desired} \le a_{min} \; and \; x_I \le 0 \\ -K_k x_k - K_I x_I, & a_{desired} \le a_{min} \; and \; x_I \ge 0 \end{cases} \tag{4.3}$$

The logic control used, allows the desired acceleration to be close to the saturate input. As presented, when the desired acceleration is in the predefined boundary, the system behaves as expected (the requirement R4 is met). Or else, depending on the sign of the error and the desired acceleration, the logic control is applied. Figure 4-2 shows the simulation results using the designed logic control.

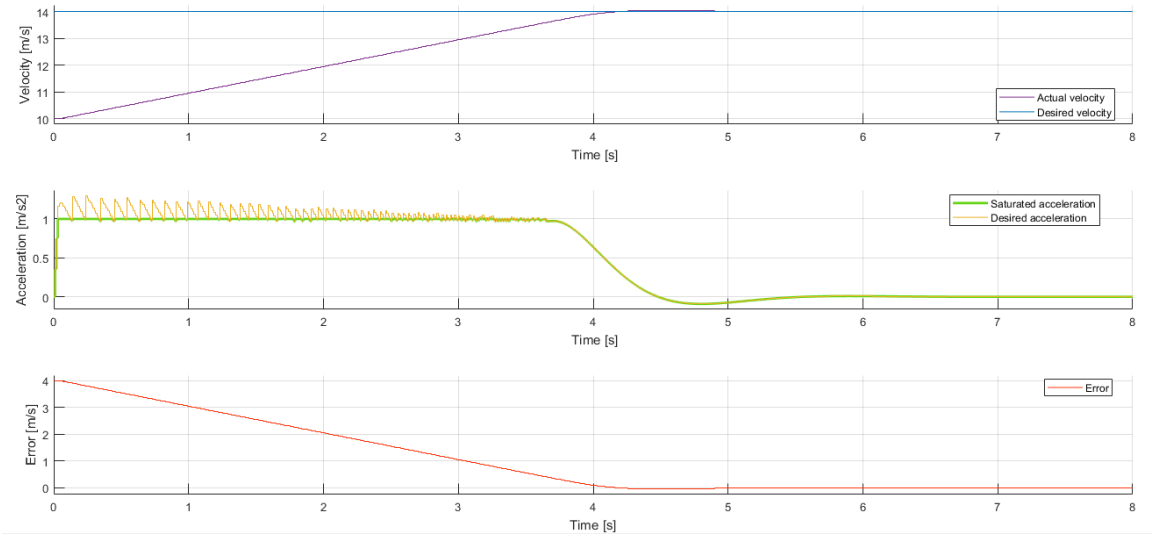

Figure 4-2. Simulation result of controller with saturation control
Step response of 14 m/s and initial speed of 10 m/s

4.3 Summary

In this chapter, a logic control based on the error between the desired and actual velocity has been developed. To implement this control in the UC controller, the input of the observer designed in section 3.3.1.1 must be replaced by the saturated acceleration, because, the observer is a copy of the plant. Despite the saturation input, the observer state (velocity and acceleration) still converges and tracks the true state of the plant.

This chapter completes the design of the state space controller. The requirements R1 (maintain or get the desired set speed), and R4 (driving comfort) are met.

5 Online parameters estimation

The desired torque formula (3.44) established in chapter 3 (see section 3.3.1.2) consists of two terms: the torque applied to reach the desired acceleration from the UC controller, and the torque applied to overcome all disturbances.

The control law used for the desired torque basically depends on the external disturbances. We assume that the vehicle is driven on a flat road, which implies a road grade of zero percent. Moreover, we assume a constant vehicle mass, while the mass of both city buses and delivery trucks might change when the vehicles are brought to a standstill. With these assumptions, when the road grade, and the vehicle mass vary, the behavior of the controller changes: either it takes a long time to reach the desired velocity or it diverges to negative values. Moreover, the slow reaction of the controller causes undershoots.

As said above, the second term of the desired torque is applied to overcome all disturbances. It means that when the climbing, the rolling, and the drag resistance change, this second term of the desired torque shall provide enough torque to overcome these disturbances.

This chapter focuses on the study of a state observer to estimate the vehicle mass, and the road grade. These estimated values are used to compensate the external disturbances. We assume that the drag coefficient is constant.

5.1 Estimator model

The model used to estimate the vehicle mass, and the road grade is based on the longitudinal vehicle dynamics model defined in chapter 3 (see section 3.1).

By combining the expression (3.1) of the longitudinal vehicle dynamics, (3.2) of the wheel dynamic, and the expression of all external disturbances (3.3), (3.4) and (3.5), we obtain

$$(m + m_r)\dot{V}_x = \frac{T_w}{R_w} - \left[\frac{1}{2}\rho_{air} C_d A_F V_x^2 + f_r mg \sin(\theta) + mg \sin(\theta)\right] \tag{5.1}$$

Amongst these variables, the torque transmitted to the wheel T_w, and the longitudinal vehicle velocity V_x are known at each time.

5.2 Kalman Filter

Rudolf E. Kalman introduced a linear estimator known as Kalman filter between 1958 and 1961. Based on Bayesian theory, the Kalman filter is used to estimate the state of a dynamic system. In the Kalman filter, a discrete time dynamic system is represented by

$$\begin{cases} x_{k+1} = F_k x_k + B_k u_k + w_k \\ z_k = H_k x_k + e_k \end{cases} \tag{5.2}$$

$$\begin{cases} w_k \sim N(0, Q_k) \\ e_k \sim N(0, R_k) \end{cases} \tag{5.3}$$

where x_k is the state space vector which contains the state variables, F_k is the state matrix, B_k is the input matrix, u_k is the input of the system, w_k is the process noise with zero mean, and with covariance Q_k corresponding to the system error, z_k is the output vector, H_k is the observation matrix, and e_k is the measurement noise with zero mean, and with covariance R_k corresponding to the measurement error.

In the Kalman filter, it is assumed that the state vector is a Gaussian variable, which has for mean value $\bar{x}$, and covariance matrix P_k.

$$x_k \sim N(\bar{x}, P_k) \tag{5.4}$$

The algorithm used by the Kalman filter is defined as

Initialization

1. Initialize the state of the filter. $x_0 = x(0)$
2. Initialize the covariance matrix of the state, which corresponds on how accurate the initial value of the state vector is. $P_0 = P(0)$

Predict

1. Use the first equation of the expression (5.2) to predict the state at $k + 1$
2. Adjust the accuracy of the state vector, which corresponds to the covariance matrix P_{k+1} for the uncertainty in the prediction.

Update

1. Get a measurement which corresponds to the output vector z_k, and associate the accuracy related to this measurement, R_k.
2. Compute the difference between the measurement and the predicted state.
3. Compute the Kalman gain, which describes the weight of the accuracy between the measurement and the predicted state.
4. Based on the computed Kalman gain, set the new state between the predicted state and the measurement.
5. Based on the accuracy of the measurement, update the accuracy of the state vector

The algorithm is summarized by the Kalman filter equations

Initialization

$$\hat{x}_{1|0} = [\hat{x}_{1|0_1} \quad \hat{x}_{1|0_2} \quad \hat{x}_{1|0_3} \quad \cdots \quad \hat{x}_{1|0_{n-1}} \quad \hat{x}_{1|0_n}]^T$$
$$P_{1|0} = var(\hat{x}_{1|0})$$

Predict

$$\begin{aligned} \hat{x}_{k+1|k} &= F_k \hat{x}_{k|k} + B_k u_k \\ P_{k+1|k} &= F_k P_{k|k} F_k^T + Q_k \end{aligned} \tag{5.5}$$

Update

$$\begin{aligned} \tilde{y}_k &= y_k - H_k \hat{x}_{k|k-1} \\ K_k &= P_{k|k-1} H_k^T \left(H_k P_{k-1|k} H_k^T + R_k\right)^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k \tilde{y}_k \\ P_{k|k} &= (I - K_k C_k) P_{k|k-1} \end{aligned} \tag{5.6}$$

where var denotes the variance, $\hat{x}_{k|k}$ is the estimated state of the state vector x_k at step k, $\hat{x}_{k+1|k}$ is the pre-estimation of the state vector x_k at step k based on step $k+1$, $P_{k|k}$ is the covariance matrix related to the state vector $\hat{x}_{k|k}$ at step k, and $\tilde{y}_k$ represents the residual.

For more information related to the Kalman filter and its method, please refer to [10].

5.3 Extended Kalman Filter

The Kalman filter works for linear systems. There is a variant of the Kalman Filter used for nonlinear systems, the Extended Kalman Filter (EKF). The algorithm presented above is unchanged for the EKF.

The general discrete representation of a dynamic system is given by

$$\begin{cases} x_{k+1} = f_k(x_k, u_k, w_k) \\ z_k = h_k(x_k) + e_k \end{cases} \tag{5.7}$$

where f_k and h_k are two nonlinear functions.

The EKF compared with the Kalman filter linearizes the nonlinear function f_k, and h_k around the current estimated state at each step. To predict the state at $k+1$, the first equation of (5.7) is used. It means the use of the nonlinear function f_k.

The equations used by the EKF are

Initialization

$$\hat{x}_{1|0} = [\hat{x}_{1|0_1} \quad \hat{x}_{1|0_2} \quad \hat{x}_{1|0_3} \quad \cdots \quad \hat{x}_{1|0_{n-1}} \quad \hat{x}_{1|0_n}]^T$$
$$P_{1|0} = var(\hat{x}_{1|0})$$

Predict

$$\begin{aligned} F_k &= \frac{\partial f_k}{\partial x_k}(x_k, u_k)_{|\hat{x}_{k|k}, u_k} \\ W_k &= \frac{\partial f_k}{\partial w_k}(x_k, u_k)_{|\hat{x}_{k|k}, u_k} \\ x_{k+1} &= f_k(x_k, u_k) \\ P_{k+1|k} &= F_k P_{k|k} F_k^T + W_k Q_{k-1} W_k \end{aligned} \tag{5.8}$$

Update

$$\begin{aligned} H_k &= \frac{\partial h_k}{\partial x_k}(x_k)_{|\hat{x}_{k|k}} \\ \tilde{y}_k &= y_k - h_k(\hat{x}_{k|k-1}) \\ K_k &= P_{k|k-1} H_k \left(H_k P_{k-1|k} H_k^T + R_k\right)^{-1} \\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k \tilde{y}_k \\ P_{k|k} &= (I - K_k C_k) P_{k|k-1} \end{aligned} \tag{5.9}$$

where W_k is a matrix corresponding to the uncertainty in each variable of the system: inputs and state variables.

5.3.1 Mass and road grade estimation

Using the method in [6] to reduce computational complexity, we can rewrite the expression (5.1)

$$\frac{\sin(\theta + \varphi)}{\cos(\varphi)} = \sin(\theta) + \cos(\theta) * \tan(\varphi) \tag{5.10}$$

Thanks to (5.10), (5.1) becomes

$$\dot{V}_x = \frac{\left(\frac{T_w}{R_w} - C_a V_x^2\right) - m \cdot G \sin(\theta + \varphi)}{(m + m_r)} \tag{5.11}$$

where $C_a = \frac{1}{2}\rho_{air} C_d A_F$, $\varphi = \cos\left(artan(fr)\right)$, and $G = \frac{g}{\varphi}$.

The state vector chosen to estimate the vehicle mass, and the road grade is defined as

$$x = \begin{bmatrix} V_x \\ \frac{1}{m} \\ \sin(\theta + \varphi) \end{bmatrix} \leftrightarrow x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{5.12}$$

The process noise vector is represented by

$$w = [w_1 \quad w_2 \quad w_3]^T \tag{5.13}$$

where:

- w_1 is the process noise describing the uncertainty in the torque signal. As stated in [6], because the torque is not directly measured but calculated, we may have losses in the signal

- w_2 is the process noise of x_2. We assume a constant vehicle mass. However, for uncertainty in the mass value, the derivative is modelled as a noise with small variance.

- w_3 is the process noise of x_3. We assume that the variation of the road slope is low. However, the road slope variation in mathematical sense is unknown, then, the derivative of the slope is assumed to be a noise with a certain variance.

From (5.11), (5.12) and (5.13), the state representation of the system becomes

$$\dot{x} = \begin{cases} \dfrac{\left[\dfrac{T_w(1+w_1)}{R_w} - C_a x_1^2\right] x_2 - G x_3}{1 + x_2 m_r} \\ w_2 \\ w_3 \end{cases} \tag{5.14}$$

Using Euler discretization, (5.14) yields to

$$x_{k+1} = \begin{cases} V_{x_k} + dt \dfrac{\left[\dfrac{(T_w + w_{k_1})}{R_w} - C_a {x_{k_1}}^2\right] x_{k_2} - G x_{k_3}}{1 + x_{2_k} m_r} \\ x_{k_2} + dt \cdot w_{k_2} \\ x_{k_3} + dt \cdot w_{k_3} \end{cases} \tag{5.15}$$

where dt is the sampling time.

Using (5.7), we finally obtain

$$f_k(x_k, T_w, w_k) = \begin{cases} V_{x_k} + dt\dfrac{\left(\dfrac{(T_w + w_{k_1})}{R_w} - C_a {x_{k_1}}^2\right) x_{k_2} - G x_{k_3}}{1 + x_{2_k} m_r} \\ x_{k_2} + dt \cdot w_{k_2} \\ x_{k_3} + dt \cdot w_{k_3} \end{cases} \tag{5.16}$$

$$z_k = H_k x_k + e_k$$

where $H_k = [1 \quad 0 \quad 0]$.

To apply the EKF, the state matrix F_k, and the matrix W_k should be calculated at each step based on the current estimated state. Using the expression from (5.8), we find

$$F_k = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_{k_1}} & \dfrac{\partial f_1}{\partial x_{k_2}} & \dfrac{\partial f_1}{\partial x_{k_3}} \\ \dfrac{\partial f_2}{\partial x_{k_1}} & \dfrac{\partial f_2}{\partial x_{k_2}} & \dfrac{\partial f_2}{\partial x_{k_3}} \\ \dfrac{\partial f_3}{\partial x_{k_1}} & \dfrac{\partial f_3}{\partial x_{k_2}} & \dfrac{\partial f_3}{\partial x_{k_3}} \end{bmatrix}_{|\hat{x}_{k|k}, u_k} \tag{5.17}$$

$$F_k = \begin{bmatrix} 1 - 2 \cdot \dfrac{\hat{x}_{k_2} C_a x_{k_1}}{1 + \hat{x}_{k_2} m_r} dt & \dfrac{\frac{T_w}{R_w} - C_a {\hat{x}_{k_1}}^2 + G\hat{x}_{k_3}}{(1 + \hat{x}_{k_2} m_r)^2} dt & -\dfrac{Gdt}{1 + \hat{x}_{k_2} m_r} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}_{|\hat{x}_{k|k}, u_k}$$

$$W_k = \begin{bmatrix} \dfrac{\partial f_1}{\partial w_1} & \dfrac{\partial f_1}{\partial w_2} & \dfrac{\partial f_1}{\partial w_3} \\ \dfrac{\partial f_2}{\partial w_1} & \dfrac{\partial f_2}{\partial w_2} & \dfrac{\partial f_2}{\partial w_3} \\ \dfrac{\partial f_3}{\partial w_1} & \dfrac{\partial f_3}{\partial w_2} & \dfrac{\partial f_3}{\partial w_3} \end{bmatrix}_{|\hat{x}_{k|k}, u_k} \tag{5.18}$$

$$W_k = \begin{bmatrix} \dfrac{T_w}{R_w} \cdot \dfrac{\hat{x}_{2_k}}{1 + \hat{x}_{2_k} m_r} dt & 0 & 0 \\ 0 & dt & 0 \\ 0 & 0 & dt \end{bmatrix}_{|\hat{x}_{k|k}, u_k}$$

As done for the design of the observer in chapter 3 (section 3.3.3.1), we must do an observability analysis for the system.

For nonlinear systems, the observability analysis is different. Here, we must do a local observability analysis. Sometimes, equations governed by nonlinear system are too long, and thus, can lead to complex calculation if the observability analysis for linear system is applied.

The observability analysis uses the continuous time representation of the system (5.14). The approach used to determine the observability of the system, follows the method presented in [11].

Without taking the noise into account, the system can be represented in continuous time as

$$\begin{cases} \dot{x} = f(x,u) \\ z = h(x) \end{cases} \tag{5.19}$$

where x is the state variable (5.12), f is the nonlinear function defined as

$$f(x,T_w) = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} \frac{\left(\frac{T_w}{R_w} - C_a \cdot x_1^2\right) - m \cdot G \cdot x_3}{(x_2 + m_r)} \\ 0 \\ 0 \end{bmatrix} \tag{5.20}$$

and h the function given by

$$h(x) = [1 \quad 0 \quad 0] \cdot x \tag{5.21}$$

For nonlinear systems, a system is observable if and only if the rank of the observability matrix O is equal to the dimension of the state vector x. The observability matrix O is defined as

$$O = \begin{bmatrix} L_f^0 \\ L_f^1 \\ L_f^2 \end{bmatrix} \cdot \frac{\partial h}{\partial x}(x) \tag{5.22}$$

where L_f represents a differential operator commonly named Lie derivative of h according to the nonlinear function f.

L_f is represented by

$$L_f \cdot h(x) = [L_f \cdot h_1(x) \;\; L_f \cdot h_2(x) \ldots L_f \cdot h_r(x)]^T \tag{5.23}$$

where r is the dimension of the vector h. Each component of L_f is defined as

$$\forall\, i \in \{1,2,\ldots,r\}, \qquad L_f \cdot h_i(x) = \frac{\partial h_i}{\partial x} \cdot f(x,T_w) \tag{5.24}$$

The following results present the features of the Lie derivative:

Initial condition: $$L_f^0 \cdot h(x) = h(x) \tag{5.25}$$

Recursive calculation: $$L_f^k \cdot h(x) = L_f \left(L_f^{k-1} \cdot h(x) \right) \tag{5.26}$$

To get the observability matrix, we must determine the first three Lie derivatives

Row 1: $L_f^0 \cdot \frac{\partial h}{\partial x}(x)$

$$L_f^0 \cdot \frac{\partial h}{\partial x}(x) = \frac{\partial h}{\partial x}(x) = [1 \quad 0 \quad 0] \tag{5.27}$$

Row 2: $L_f^1 \cdot \frac{\partial h}{\partial x}(x)$

$$\begin{gathered} L_f^1 \cdot \frac{\partial h}{\partial x}(x) = L_f \left(L_f^0 \cdot \frac{\partial h}{\partial x}(x) \right) = L_f \cdot \frac{\partial h}{\partial x}(x) \leftrightarrow \frac{\partial}{\partial x}\left(L_f \cdot h(x) \right) \\ \frac{\partial}{\partial x}\left(L_f \cdot h(x) \right) = \frac{\partial}{\partial x}\left(\frac{\partial h}{\partial x}(x) \cdot f(x, T_w) \right) = \frac{\partial}{\partial x}\left([1 \quad 0 \quad 0] \cdot \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \right) = \frac{\partial f_1}{\partial x}(x, T_w) \\ L_f^1 \cdot \frac{\partial h}{\partial x}(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \end{bmatrix} \end{gathered} \tag{5.28}$$

Row 3: $L_f^2 \cdot \frac{\partial h}{\partial x}(x)$

$$\begin{gathered} L_f^2 \cdot \frac{\partial h}{\partial x}(x) = L_f \left(\frac{\partial f_1}{\partial x}(x, T_w) \right) = \frac{\partial}{\partial x}\left(L_f \cdot f_1(x, T_w) \right) \\ \frac{\partial}{\partial x}\left(L_f \cdot h(x) \right) = \frac{\partial}{\partial x}\left(\frac{\partial f_1}{\partial x}(x, T_w) \cdot f(x, T_w) \right) = \frac{\partial}{\partial x}\left(\frac{\partial f_1}{\partial x}(x, T_w) \cdot \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \right) \\ L_f^2 \cdot \frac{\partial h}{\partial x}(x) = \frac{\partial}{\partial x}\left[\frac{\partial f_1}{\partial x_1} \cdot f_1 + \frac{\partial f_1}{\partial x_2} \cdot f_2 + \frac{\partial f_1}{\partial x_3} \cdot f_3 \right] = \frac{\partial A}{\partial x} \end{gathered} \tag{5.29}$$

Finally, the observability matrix O becomes

$$O = \begin{bmatrix} 1 & 0 & 0 \\ \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \\ \frac{\partial A}{\partial x_1} & \frac{\partial A}{\partial x_2} & \frac{\partial A}{\partial x_3} \end{bmatrix} \tag{5.30}$$

The system is observable if O has maximum rank. This equivalent to say

$$\det\begin{bmatrix}\frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3}\\ \frac{\partial A}{\partial x_2} & \frac{\partial A}{\partial x_3}\end{bmatrix} \neq 0 \tag{5.31}$$

Because $f_2 = 0$, and $f_3 = 0$ (see 5.20), A can be simplified as: $A = \frac{\partial f_1}{\partial x_1} \cdot f_1$

(5.31) yields to

$$f_1 \cdot \left[\frac{\partial f_1}{\partial x_2} \cdot \frac{\partial^2 f_1}{\partial x_3 \partial x_1} - \frac{\partial f_1}{\partial x_3} \cdot \frac{\partial^2 f_1}{\partial x_2 \partial x_1}\right] \neq 0 \tag{5.32}$$

Using (5.20) and (5.32), we can deduce

$$-\frac{2V_x C_a G \dot{V}_x}{1 + x_{2_k} m_r} \neq 0 \tag{5.33}$$

On/Off logic

As demonstrated above, the estimator works under some conditions, which satisfy the relationship (5.33). According to (5.33), the estimator needs a dynamic situation to run. Moreover, the mechanical brake force has not been modelled in the longitudinal vehicle dynamics model. The results from [6] shows that the EKF performed well in a certain range of torque, and from a certain velocity. Thus, the accuracy of the designed EKF will depend on preset conditions defined as

Longitudinal Acceleration
The absolute value of the longitudinal acceleration shall be strictly higher than 0.1 m/s^2

Threshold Velocity
The estimator starts to run when the vehicle has reached at least a velocity of 10 m/s

In – wheel – motor-based Torque
The absolute value of the in – wheel – motor-based torque shall be between 2000 $N.m$ and 10 000 $N.m$

Braking
When the brake pedal is pressed, the estimator does not run

When these conditions are not valid, the estimation of the state variable is stopped until these conditions are satisfied. However, after some time, the accuracy of the velocity can be trustworthy. Therefore, the estimator keeps estimating the actual velocity, if the velocity is above the threshold value.

The EKF will be implemented in city buses, and delivery trucks. For a city bus, the mass does not change when the vehicle is in motion. However, when the bus stops, and the bus doors are open, people may get on or off the bus. Therefore, when the bus is stopped, and the bus doors are open, the EKF must be reinitialized at the current estimated mass. A logic has been developed to stop the estimation of the vehicle mass when this one is trustworthy, and reinitialize the EKF at the current estimated mass when the bus doors are open.

5.4 Luenberger observer

The need to develop a second observer for the road slope, results in the fact that the acceleration shall be non-zero. And by considering the design of a cruise control, this condition cannot be applied. Therefore, in this part, a Luenberger observer (LO) has been developed to enhance the road grade estimation.

The LO observer has already been introduced in chapter 3 (see section 3.1.1.1) to estimate the vehicle acceleration. As said in chapter 3, the observer is a simple copy of the vehicle plant model. The vehicle plant model is based on the relationship (5.1).

We can rewrite (5.1) as

$$M \cdot \dot{V}_x = F - \hat{m} \cdot G \cdot \alpha \tag{5.34}$$

where $M = \hat{m} + m_r$, $F = \frac{T_w}{R_w} - \frac{1}{2}\rho_{air} C_d A_F V_x^2$, $\hat{m}$ is the estimated mass from the EKF, and $\alpha = \sin(\theta + \varphi)$.

5.4.1 Road grade estimation

The discrete time representation of the system for the road grade estimation is given by

$$\begin{bmatrix} V_{x_{k+1}} \\ \alpha_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & -dt\frac{\hat{m}}{M} \cdot G \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_{x_k} \\ \alpha_k \end{bmatrix} + \begin{bmatrix} \frac{dt}{M} \\ 0 \end{bmatrix} \cdot F_k$$
$$V_{x_k} = [1\ 0] \cdot \begin{bmatrix} V_{x_k} \\ \alpha_k \end{bmatrix} \tag{5.35}$$

Here, we apply the same method presented in chapter 3 (see section 3.3.1.1) to find the observer gain

$$\det\left(\begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix} - \begin{bmatrix} 1 & -dt \cdot \frac{\hat{m}}{M} \cdot G \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \cdot [1 \quad 0] \right) = 0 \tag{5.36}$$

where $a = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ is the observer gain.

We choose $z_1 = \exp(-4 \cdot dt)$ and $z_2 = \exp(-5 \cdot dt)$ as root of the system

Using the method in chapter 3 (see section 3.3.1.1), we finally obtain

$$(5.20) \leftrightarrow \begin{cases} a_1 = 2 - (z_1 + z_2) \\ a_2 = \dfrac{1}{dt \cdot GM}[1 + z_1 + z_2 - z_1 z_2] \end{cases} \tag{5.37}$$

On/Off logic

By doing an observability analysis for this system, we should find that the system is always observable. However, we should consider that the mechanical brake is not modeled in the longitudinal vehicle dynamics model. Therefore, when the brake pedal is applied, the LO observer shall stop the estimation of the road grade but keep estimating the velocity. It has been set that the LO observer runs when the vehicle speed is above 0.1 m/s.

5.5 Summary

In this chapter, methods to estimate the mass and the road grade has been introduced. The approach uses the Extended Kalman filter (EKF), and Luenberger Observer (LO observer). The EKF estimates simultaneously the vehicle mass and the road grade. To design this observer an observability analysis has been done. This observability analysis has led to use the estimator under some conditions. Therefore, a control logic has been developed to stop the estimation of the state variables when the conditions are not satisfied.

One of the functions of the cruise control is to maintain a desired speed, which implies zero acceleration. Therefore, a Luenberger observer (LO observer) has been developed to take over the estimation of the road grade when the EKF does not run. Even if the observability analysis of the LO observer has shown that the system could run in any condition, a logic has been implemented to stop the LO observer either when the brake pedal is applied, or the velocity is below 0.1 m/s.

6 Software verification and validation

In the previous chapters, a cruise controller using state space representations has been designed. In addition, a model observer using the Extended Kalman Filter (EKF), and the Luenberger Observer (LO observer) has been developed to estimate the vehicle mass, and the road grade.

By looking at the V – diagram introduced in chapter 2 (see section 2.1), we are still in the Software design stage. To complete this stage, we must apply the Unit Testing (UT) to the overall cruise control system (CC system), and the Model in the Loop (MIL) to the Powertrain Control Module (PCM). The UT testing is applied using Simulink Test, and the MIL testing using MATLAB/Simulink.

This chapter aims to verify and validate the CC system for real time implementation. Real time implementation implies Hardware in the Loop (HIL), and vehicle test. These parts will not be discussed in this master's thesis.

6.1 CC system – Simulink Model

As said in chapter 1, the software team has developed a logic controller to respond to the requirement R2 (The CC system shall be controlled by the driver). The cruise controller designed in this master's thesis, computes the desired torque to meet the requirements R1 (Maintain or get the desired speed set by the driver), R3 (It shall be able to adapt in dynamic and static variation), and R4 (Driving comfort). Both logic and cruise controller form the complete CC system. Therefore, the UT testing will be applied to the CC system. Figure 6-1 illustrates the high-level architecture of the CC system, and Figure 6-2 illustrates the designed cruise controller in Simulink.

The CC system is divided in three subsystems described in Table 6-1.

Subsystem	Description
Handle input	Checks if all the CC system inputs are valid to use them.
CC logic	Represented by a StateFlow to model a state machine, it uses Handle subsystem outputs to return the CC status (OFF, ON, SET, HOLD, RESUME, ACCELERATE, and DECELERATE), and the set speed. These outputs will then be used by the Control torque request subsystem.
Control torque request	Computes the torque request and returns the set speed and the CC status. This subsystem includes the designed cruise controller, and other components designed by the software team. Figure 6-3 shows the Control torque request subsystem in Simulink.

Table 6-1. Cruise Control subsystems

The software team has developed the Handle input, and CC logic subsystems.

As seen in chapter 5, the estimator needs the actual torque, the actual speed, the door signal, and the brake pedal signal to run. The reset input is used by the integral control of the UC controller. It resets the initial state of the integral block when the SET or RESUME button is pressed. This avoids undesirables over or undershoots response at the beginning of the CC activation. The cruise controller inputs and outputs are summarized in Table 6-2.

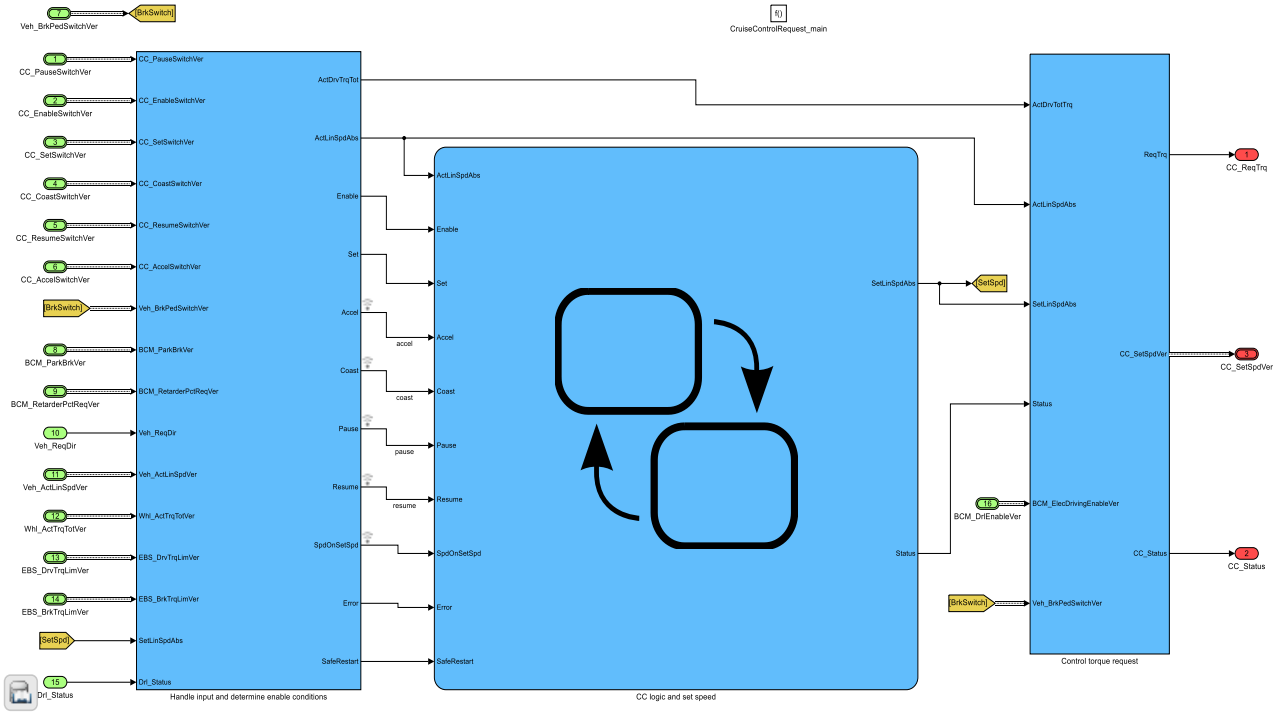

Figure 6-1. High level architecture of the CC system in Simulink

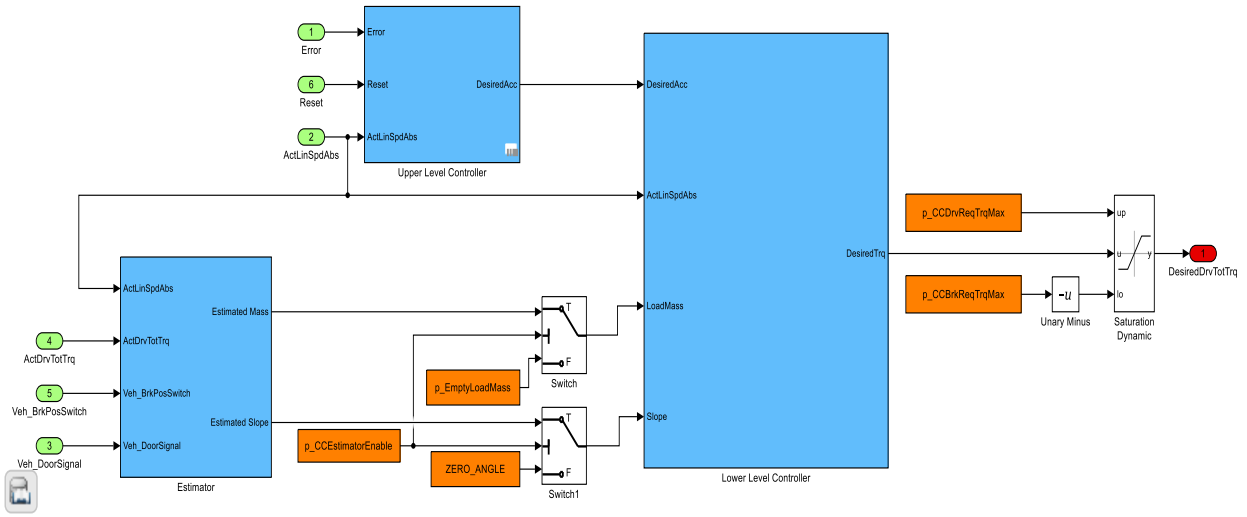

Figure 6-2. Designed cruise controller in Simulink

	Variable	Description
Input	Error	Difference between the set and actual vehicle speed.
	ActLinSpdAbs	Actual vehicle speed.
	Veh_DoorSignal	Boolean which returns true when the vehicle doors are open.
	ActDvTotTrq	Actual torque.
	Veh_BrkPosSwitch	Boolean, which returns true when the brake pedal is pressed.
	Reset	Boolean variable, which returns true when the RESUME or SET button is pressed. It is used to reset the initial condition of the integral control at the actual speed.
Output	DesiredDrvTotTrq	Desired torque request

Table 6-2. Cruise controller inputs and outputs

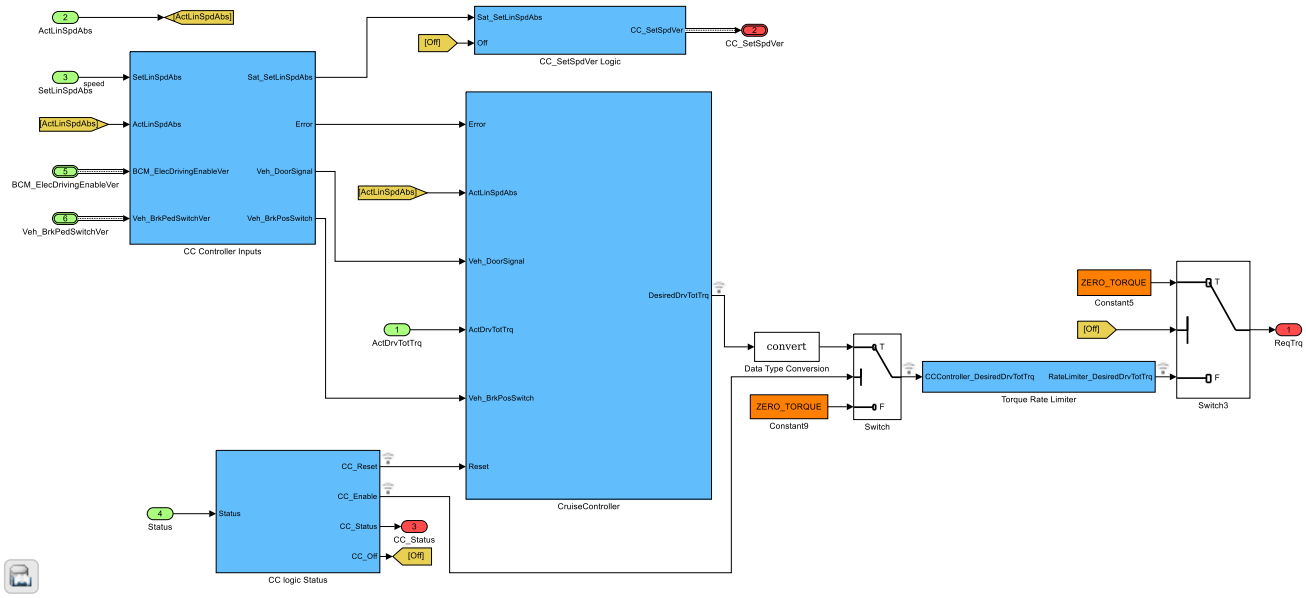

Figure 6-3. Control torque request subsystem in Simulink

6.2 Unit Testing

When designing a software component which uses logic controllers, the UT testing must be applied. UT testing consists on running individually in open-loop, software components of a system. The test cases used for the unit test are made for purpose of testing all possibilities of logic controllers. In this section, we unit test the CC system using Simulink Test.

All parts of the CC system shall respond to some requirements. These requirements are met using logic controllers. The software team has already verified the functionality of the Handle input, and CC logic subsystems. Now, we must verify the functionality of the cruise controller.

In this section, we first unit test each subsystem of the cruise controller, and then, the CC system. Since the LC controller is based on mathematical operations, the UT testing of this controller is not applied.

6.1.1 Cruise Controller Unit Testing

First, we unit test the UC controller. The UC controller uses a logic controller represented by a StateFlow to model the Anti-Windup control. Figure 6-4 shows the UC controller subsystem in Simulink, and Table 6-3 describes its inputs and outputs.

The goal of the UC controller is to respond to the requirement R4. In figure 6-5, we can see that the Anti-Windup control (AW) behaves as expected. When the computed acceleration is out of the predefined boundary, the AW reset the error to zero, to avoid an open loop system.

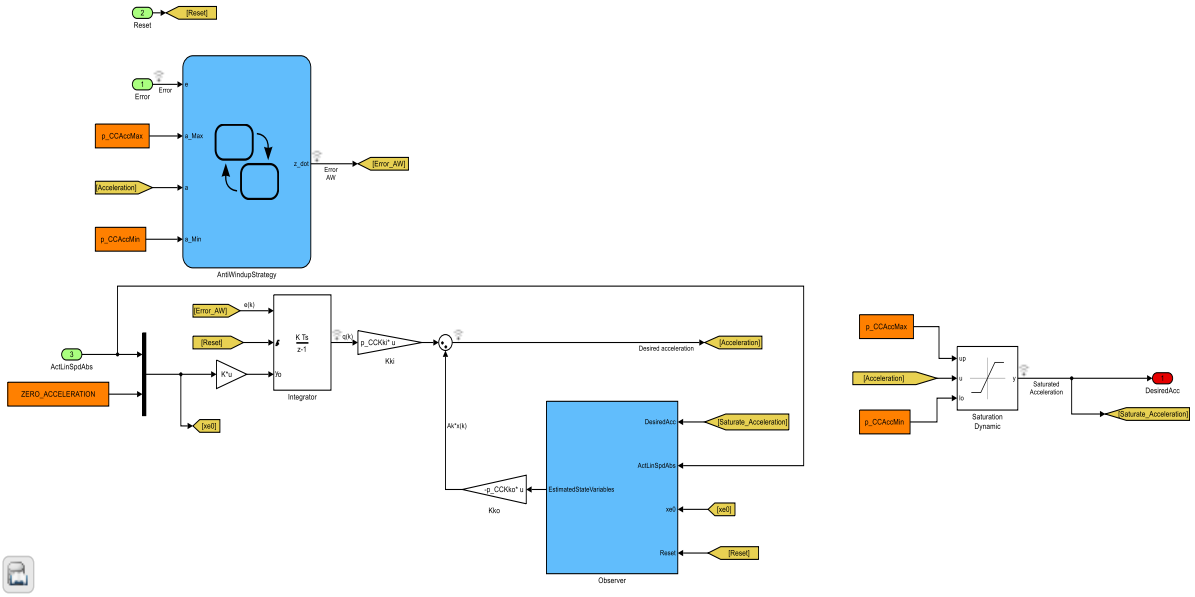

Figure 6-4. UC controller subsystem

	Variable	Description
Input	Error	Difference between the set and actual vehicle speed
	ActLinSpdAbs	Actual vehicle speed
	Reset	Boolean which returns true when the RESUME or SET button is active. It is used to reset the initial condition of the integral control at the actual speed.
Output	DesiredAcc	Desired acceleration

Table 6-3. UC controller inputs and outputs

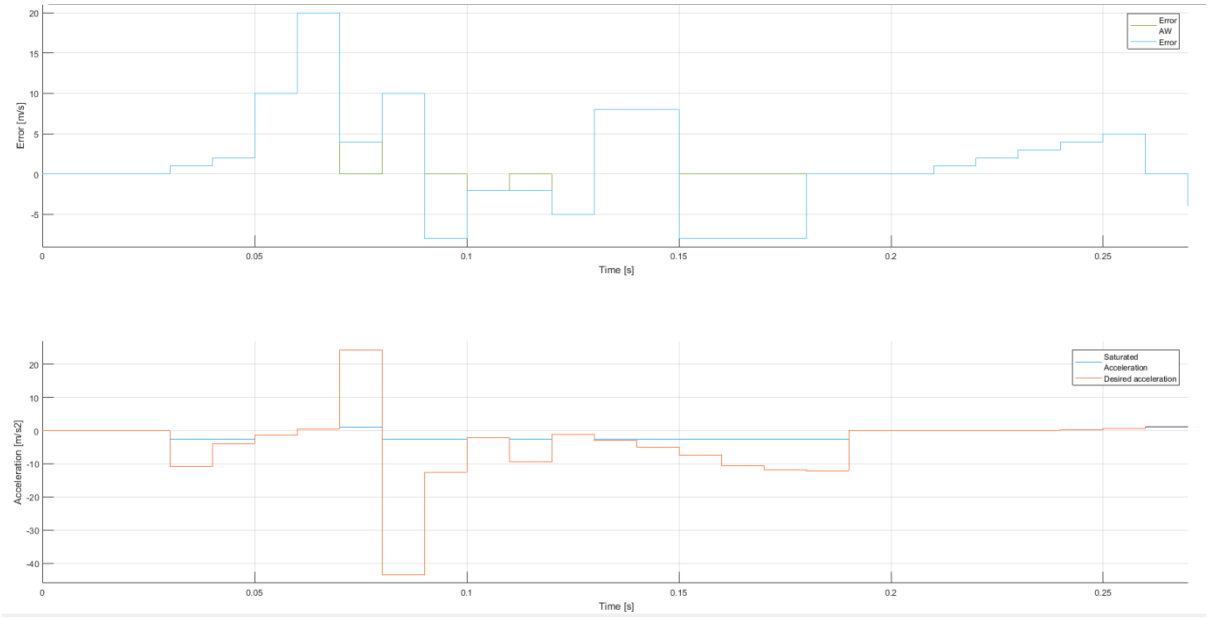

Figure 6-5. UC controller unit test results

Model observer

The model observer consists of two estimators: the EKF and the LO observer. Both estimators use logic controllers to make them run when the On/Off conditions defined in chapter 5 are satisfied. However, because the calculations of the EKF are too long, we decide to unit test one part of the EKF. Figure 6-6 shows both estimators modelled in Simulink.

The unit testing is applied to each estimator separately.

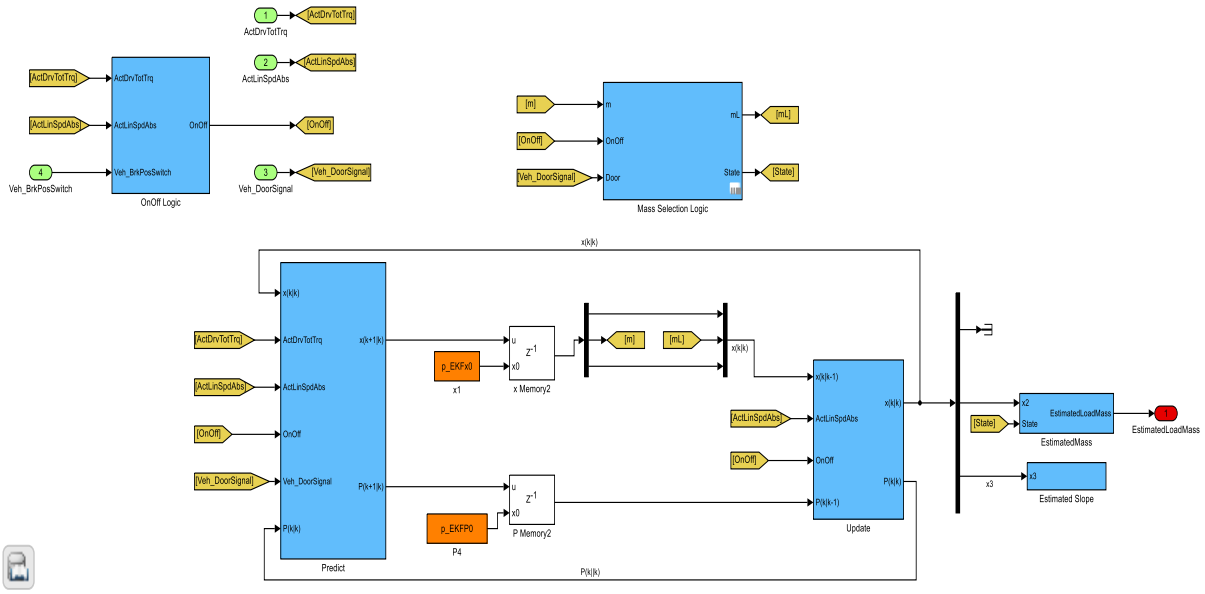

Figure 6-6. a. EKF estimator modelled in Simulink

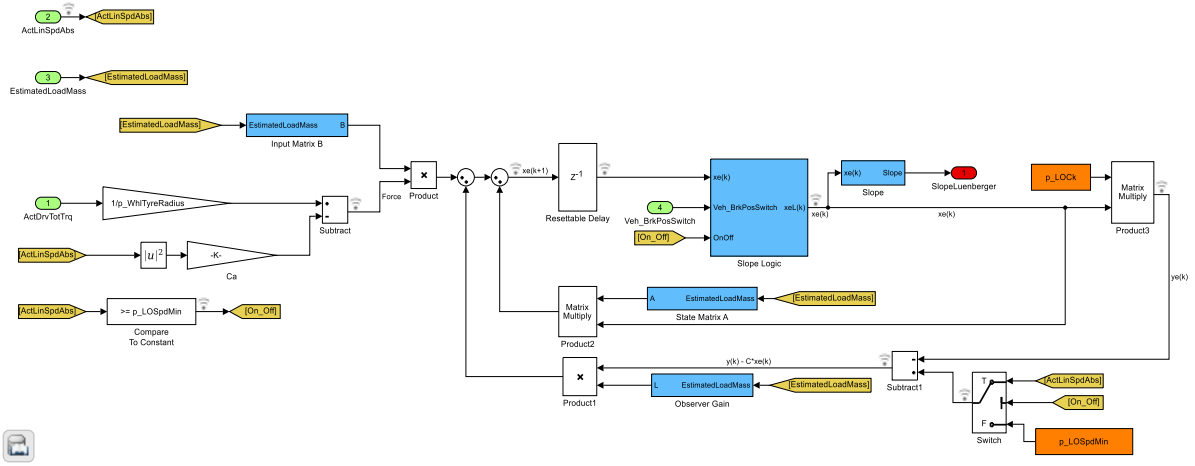

Figure 6-6. b. LO observer modelled in Simulink

EKF estimator

In the EKF, the Mass Selection Logic subsystem is tested. Table 6-4 describes the Mass Selection Logic inputs and outputs. The goal of this subsystem is to stop the estimation of the mass when this one is trustworthy. This subsystem uses logic blocks.

For the mass selection, a StateFlow is used. The StateFlow works as follow: if the estimator is running, and it has not been paused for 50 seconds, then, the estimation of the mass is stopped. Figure 6-7 shows the Mass Selection Logic subsystem, and Figure 6-8 shows the functionality of the StateFlow.

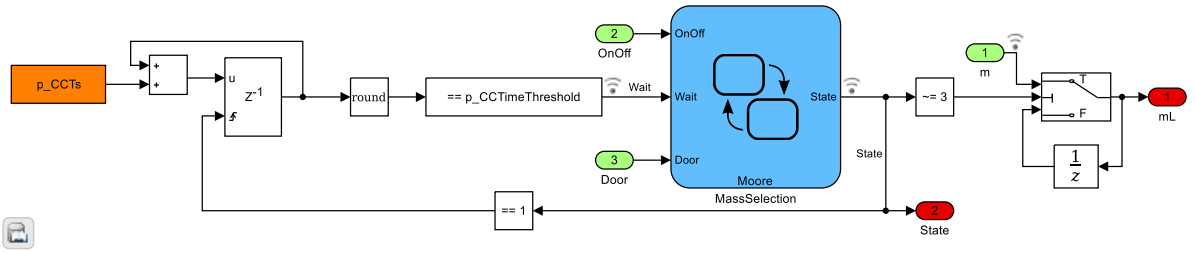

Figure 6-7. Mass Selection Logic subsystem

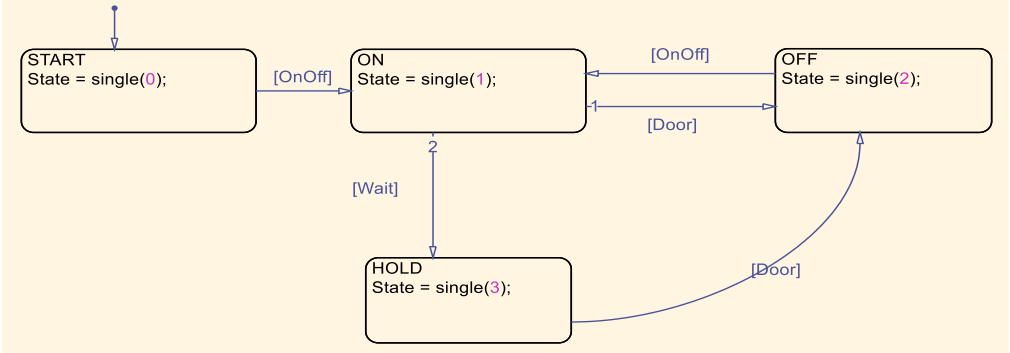

Figure 6-8. Mass Selection StateFlow principle

When unit testing, it is possible to override constant parameters such as the threshold time parameter (50 seconds). Due to the computational time, we set this parameter to 0.08 seconds.

When the StateFlow is in ON state, the StateFlow returns 1, and then, the estimation of the mass continues to run. When the StateFlow is in OFF state, the StateFlow returns 2. At this state, logic blocks present in the predict and update subsystem make sure that the estimator does not run, and thus the estimated mass remains constant at the output of the update block. When the StateFlow is in HOLD state, the StateFlow returns 3, and the outputs of the Mass Selection Logic of the estimated mass remains constant. Figure 6-9 shows the results of the Mass Selection Logic unit test.

Mass Selection Logic block

	Variable	Description
Input	m	Second component of the estimated state vector defined in chapter 5 (see section 5.3.1).
	OnOff	Boolean which refers to the On/Off Logic stated in chapter 5 (see section 5.3.1).
	Door	Boolean which returns true if the vehicle doors are open.
Output	mL	Second component of the estimated state vector coming from the StateFlow block
	State	Variable corresponding to the state number of the StateFlow

Table 6-4. Mass Selection Logic inputs and outputs

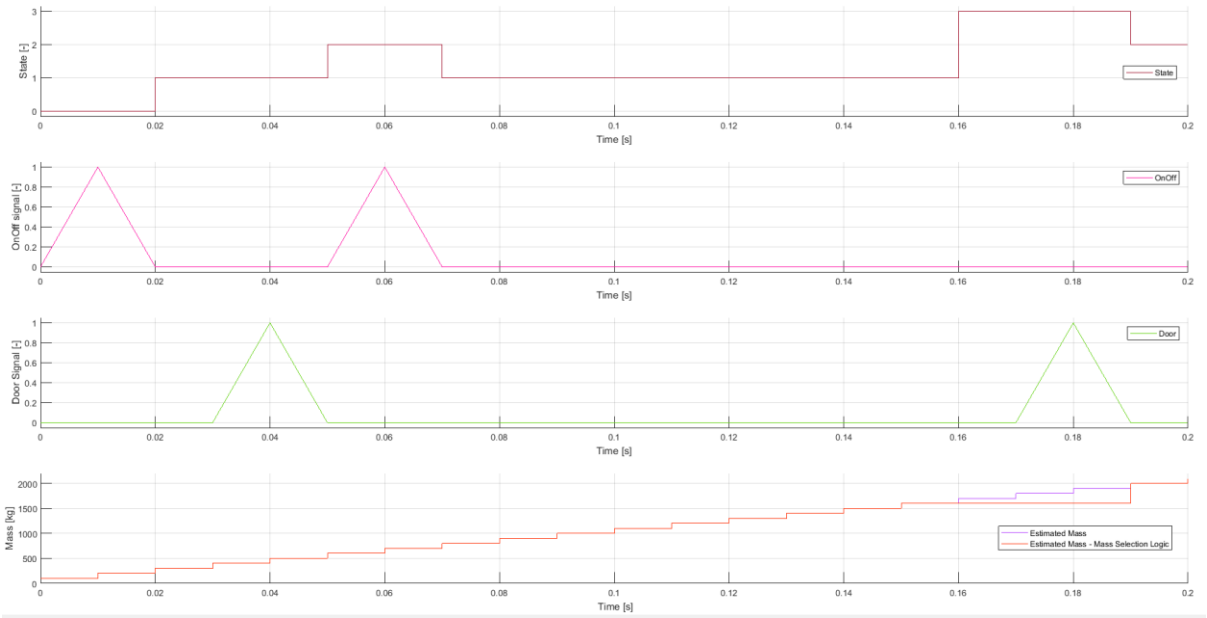

Figure 6-9. Mass Selection Logic Unit test result

LO observer

As the EKF, the LO observer runs under the conditions defined in chapter 5. Table 6-5 describes the LO observer inputs and outputs. The LO observer starts to run when the velocity is above 0.1 m/s. When, the velocity is below 0.1 m/s, the estimated velocity is kept at 0.1 m/s. For the estimated slope, either when the brake pedal is applied, or the velocity is below 0.1 m/s the estimated slope remains constant.

Due to the time limit, the unit test for the LO observer has not been done.

	Variable	Description
Input	ActDrvTotTrq	Actual torque
	ActLinSpdAbs	Actual vehicle speed
	Estimated Load Mass	Estimated mass from the EKF estimator
	Veh_BrkPosSwitch	Boolean which returns true when the brake pedal is applied
Output	SlopeLuenberger	Estimated slope

Table 6-5. LO observer inputs and outputs

6.1.2 CC System Unit Testing

After testing the functionality of each unit of the CC system, we now test the functionality of the CC system at high level. Here, we verify the logic controllers implemented in the Handle inputs, and CC logic subsystem. In this test, the actual torque is set to 0. Therefore, the cruise controller provides the torque request. Based on the driver request (CC status), the cruise controller determine the torque request.

Since, we are interested in the status of the CC system, which allows the cruise controller to calculate the torque request, the CC system inputs and outputs are not described. Figure 6-10 shows the unit test of the CC system. Regarding the CC Status signal, we can see that the RESUME status is hold for a long time. In this case, it does not mean that the driver presses the RESUME button for a long period, but, when the RESUME button is pressed, the CC status is kept at the RESUME status until the vehicle reaches the set speed.

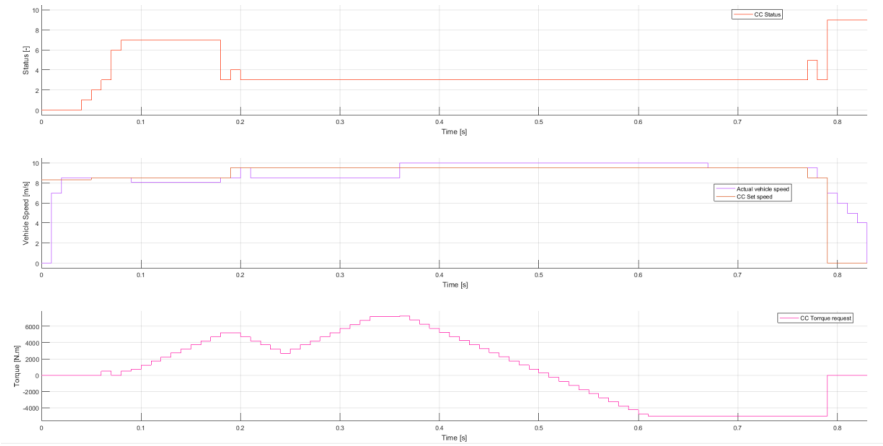

Figure 6-10. CC system unit test result.
Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9)

6.3 Model – in – the – Loop (MIL)

In the previous section, we verified the functionality of the overall CC system. All the requirements have been met.

To finally complete the Software design step, the MIL testing must be applied. In the MIL test, the CC system runs in a Loop with other components of the PCM, and other components that interact with the PCM. This test aims to summarize the behavior of the PCM (including the CC system) with other components. Figure 6-11 illustrates the principle of the MIL testing.

This section is divided in two parts. In the first part, we apply the MIL testing to the system to verify the performance of the cruise control. And in the second part, we verify the performance of the model observer.

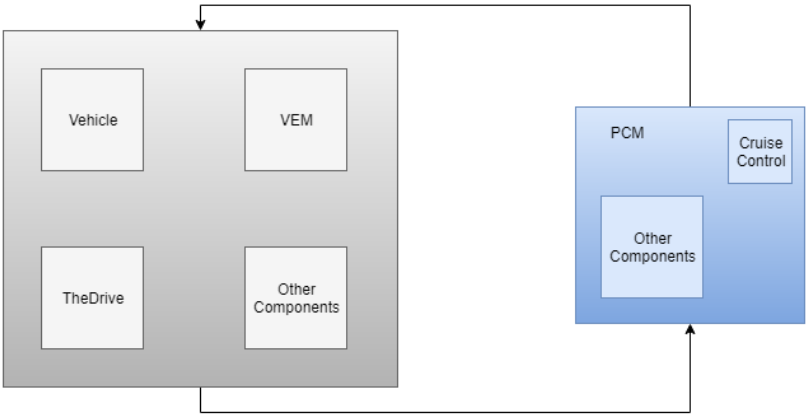

Figure 6-11. MIL testing working principle

6.3.1 CC system – MIL

To apply the MIL to the PCM, and verify the functionality of the CC system, we should consider test cases that involve the requirements defined for the CC system.

In this section, the PCM performs three test cases described in Table 6-5. The test cases TC1 and TC2 are related to the requirements R1, R2, and R3. The test case TC3 is related to the requirement R4.

In Figure 6-12, the cruise control behaves as expected. The CC system reacts to the driver's behavior, and the variation of external disturbances. For each test, the time of response of the CC system is below 5 seconds, and the steady state value of the vehicle speed is within a band of +/- 2% of the set speed. Regarding the disturbance rejection of the CC, the CC maintains the set speed using the estimated road slope from the LO observer.

Test cases ID	Test cases description	Initial conditions	Expected result
TC1	Accelerator pedal overrides the cruise control.	The cruise control is ON The driver set the cruise control at 50 km/h	The accelerator pedal is pressed, thus, the actual speed shall be greater than 50 km/h. When the Accelerator is depressed, the actual speed shall return to 50 km/h and remain at this speed until the Cruise Control is in OFF or PAUSE status.
TC2	Accelerate, Decelerate, Pause, and Resume functionality of the cruise control.	The cruise control is ON The driver set the cruise control at 40 km/h.	ACCELERATE (resp. DECELERATE) button pressed, the cruise control shall increase (resp. decreases) the vehicle speed. Brake pedal is pressed, the cruise control stops maintaining the speed and save the value of the set speed for next use. RESUME button is pressed, the cruise control is reset at the previous set speed. PAUSE button is pressed, the cruise control stops maintaining the speed and save the value of the set speed for next use.
TC3	Disturbances rejection	The cruise control is ON The driver set the cruise control at 50 km/h.	The road grade varies. The cruise control should maintain the set speed even if there is a variation of external disturbances.

Table 6-6. Cruise Control MIL test cases

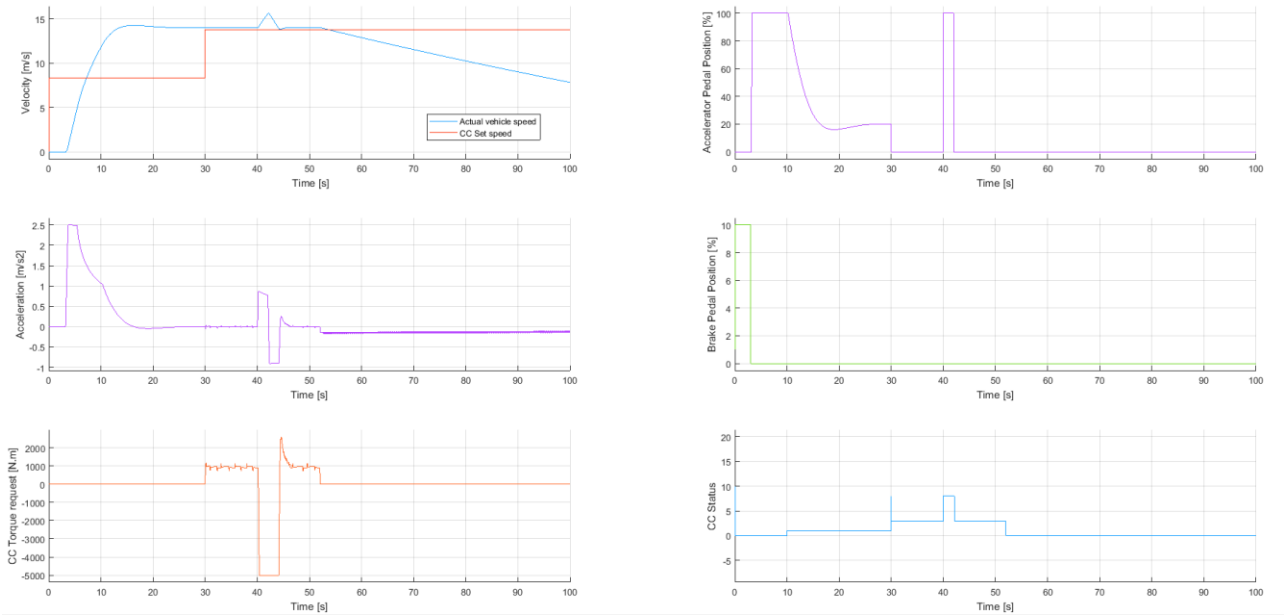

Figure 6-12. a. MIL test results – Test case 1
Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9)

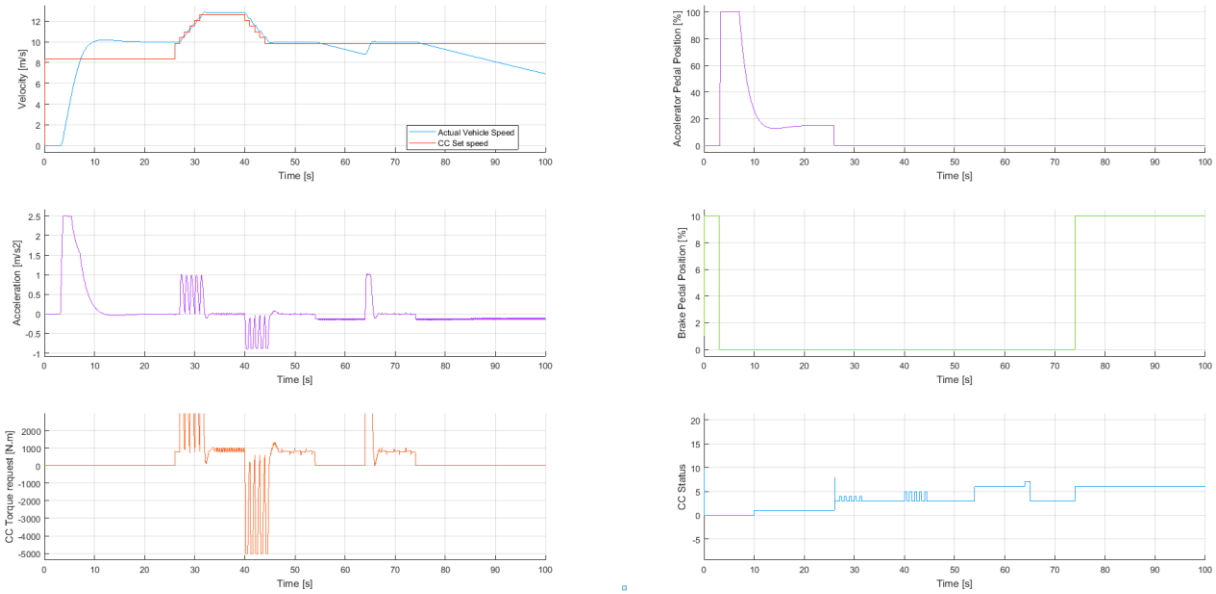

Figure 6-12. b. MIL test results – Test case 2
Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9)

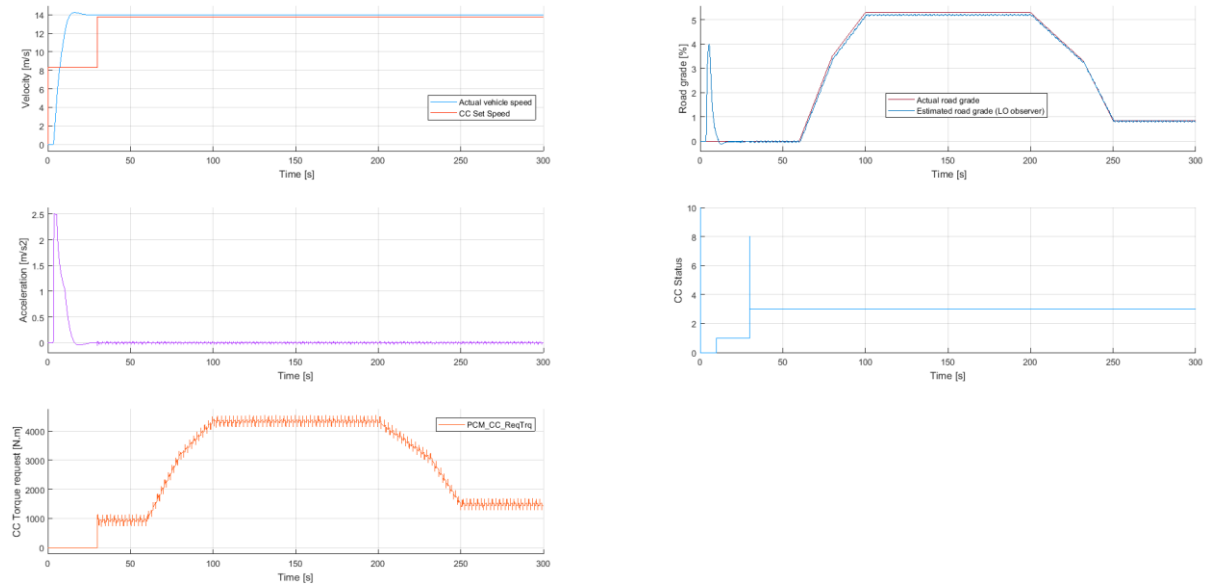

Figure 12. c. MIL test results – Test case 3
Status: OFF (0), ON (1), SET (2), HOLD (3), ACCELERATE (4), DECELERATE (5), PAUSE (6), RESUME (7), ACCELERATOR PEDAL OVERRIDE (8), and ERROR (9)

6.3.1 Estimator MIL

The test cases performed in the previous section do not simulate a dynamic situation where the EKF can be applied.

In this section, both estimators are tested with simulated data. The vehicle plant model follows a real driving cycle. To achieve this, a controller provides the required torque and mechanical brake to follow this cycle. These simulated signals are used by the estimator.

As said in chapter 5, a logic controller has been used to stop the estimation of the mass when this one is trustworthy. In figure 13, we can see that the EKF estimates the mass with a minimum amount of time, and with a certain accuracy. When the EKF stops estimating the mass, it continues to estimate the road slope. The estimated value of the mass is then used by the LO observer to estimate and enhance the slope estimation.

We can see that the estimated mass is within a band of +/-5% of the true mass. Regarding the slope, the EKF takes a certain time before reaching the true value of the road slope. Because, we assume that the variation of the road slope is low. And when we look at the road slope variation, there are some parts where the variation is high.

The LO observer reaches and tracks the road grade with a certain accuracy. We can see some overshoots of the estimated slope for the LO observer. Because, when it restarts to estimate the state, it tries to reach as quickly as possible the true slope, and thus, it leads to a non-smooth signal. This is related to the tuning of the LO observer gain.

During the MIL test for the LO observer, it has been seen that the observer seems to estimate the slope badly when the variation of the actual torque is high.

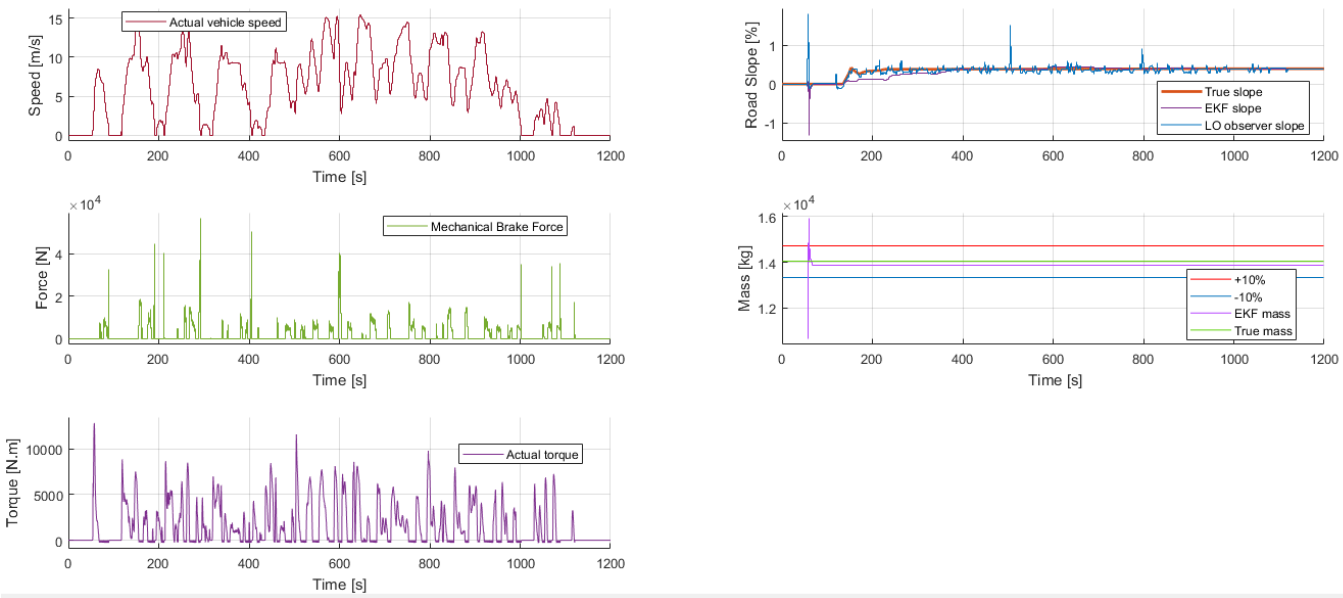

Figure 6-13. a. Simulation results for a vehicle mass of 14024 kg

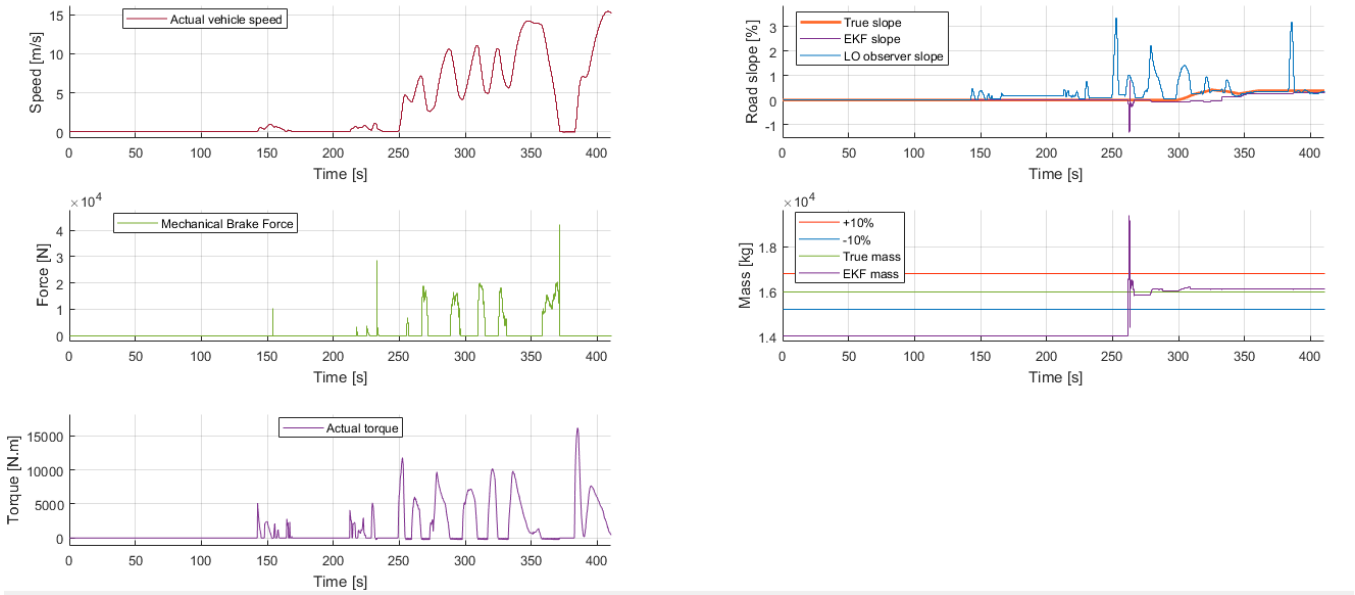

Figure 6-13. b. Simulation results for a vehicle mass of 16000 kg

6.4 Summary

In this chapter, the overall cruise control system including the designed cruise controller and logic controller related to the requirement R2 have been tested and validated. The unit testing has been applied to most of the unit of the CC system, and the CC system itself. The unit testing has been used to verify the functionalities of the CC system, and the model observer which revolve around logic controllers. To validate the designed CC system, the Model in the Loop (MIL) has been applied to test the interaction between the CC system, and other components. The results have shown that the cruise control react as expected to the driver's command, and the variation of external disturbances. In the same step, the performance of the estimator has been tested using simulated data. The Extended Kalman Filter (EKF) estimates the vehicle mass within a band of 5 to 10 percent of the true mass. Due to the assumption of a low variation of the slope, the EKF takes a certain time before reaching and tracking the true slope. However, thanks to the Luenberger observer (LO observer), it is possible to enhance the estimation of the road slope, because, the LO observer considers a high variation of the road slope. Still, the tuning of the LO observer should be improved.

7 Conclusion and discussion

The aim of this master's thesis was to develop, model, and verify a cruise control software module for an in wheel-motor based powertrain, which delivers a torque request to maintain the desired speed set by a driver. The software was designed using the MBD methodology, and MATLB/Simulink software. The designed software consists of two controllers, and a model observer. One of the controllers called upper level controller, uses a state space controller to determine the desired acceleration to maintain the set speed, the other one called lower level controller, uses the desired acceleration, longitudinal vehicle dynamics model, and powertrain features to determine the torque request, which tracks the desired acceleration. Since lower level controller uses vehicle longitudinal dynamic model, an online parameter estimation method has been developed to estimate the vehicle mass, and the road grade. The online parameter estimation involves the Extended Kalman Filter (EKF), which estimates simultaneously the vehicle mass and the road grade, and the Luenberger Observer (LO observer), which enhances the estimation of the road grade.

7.1 Conclusion

The designed software responds to all predefined requirements. It has been tested by unit testing, and validated by applying the MIL testing. The MIL shows that the designed cruise control gives good results. The cruise control reacts at any changes either from the driver's command or external disturbances with a minimum amount of time, and tracks the set speed with an acceptable accuracy. Regarding the online parameter estimation, both the EKF and LO observer give acceptable results for the vehicle mass, and road grade estimation. The EKF estimates the vehicle mass with a minimum amount of time, within a range from 5 to 10 percent of the true mass. However, the time of convergence of the road grade can be sometime quit long if the variation of slope is high. The LO observer which enhances the road grade estimation, converges and tracks the road grade with a minimum amount of time and a certain accuracy.

However, for the LO observer, when it starts or restarts to estimate from the previous estimate, the estimation starts with an overshoot. Despite the logic controller implemented for the LO observer, the estimated road grade is not smooth, but can be used by the cruise control to compute the torque request.

7.2 Discussion

The designed cruise control has met the requirements, and can be implemented in real time application. However, it can be improved.

- The parameters used by the state space controller can be improved to avoid in certain cases “hard responses”.

- A saturation control has been designed for the UC controller. However, the same control should be applied for the LC controller due to the features of the electric motor (minimum and maximum torque allowed).

- Apply the Root Mean Square Error (RMSE) to evaluate the performance of both estimators.

- Unit test and improve the control logic used for the LO observer to get a smooth road grade estimation.

- Perform test cases when the vehicle mass changes, to verify the functionality of the EKF. Due to the computational time that the MIL testing requires, these tests has not been performed.

References

[1] K. Osman, M. Rahmat and M. Ahmad, "Modelling and Controller Design for a Cruise Control System," *5th International Colloquium on Signal Processing & Its Applications (CSPA),* 2009.

[2] K. Yi, Y. Cho, S. Lee, J. Lee and N. Ryoo, "A Throttle/Brake Control Law for Vehicle Intelligent Cruise Control," *Seoul 2000 FISITA World Automotive Congress,* 2000.

[3] R. Rajamani, Vehicle Dynamics and Control, Springer, 2011.

[4] A. Vahidi, A. Stefanopoulou and H. Peng, "Recursive least squares with forgetting for online estimation of vehicle mass and road grade: theory and experiments," *Vehicle System Dynamics,* vol. 43, no. 1, pp. 31-55, 2005.

[5] S. Kim, "Design of the Adaptive Cruise Control Systems: An Optimal Control Approach," Spring, 2012.

[6] V. Winstead and I. Kolmanovsky, "Estimation of road grade and vehicle mass via model predictive control," *Proceedings of 2005 IEEE Conference on Control Applications,* 2005.

[7] E. Holm, "Vehicle Mass and Road Grade Estimation using Kalman Filter," MA thesis, Linköping, 2011.

[8] G. F. Franklin, J. David Powell and A. Emami-Naeini, Feedback Control of Dynamic Systems, Pearson, 2009.

[9] A. Packard, "Saturation and Antiwindup Strategies," 2005. [Online]. Available: https://jagger.berkeley.edu/~pack/me132/Section15.pdf.

[10] R. Labbe, "Kalman and Bayesian Filters in Python," 2018. [Online]. Available: https://drive.google.com/file/d/0By_SW19c1BfhSVFzNHc0SjduNzg/view. [Accessed February 2018].

[11] U. Kiencke and L. Nielsen, Automotive Control Systems. For Engine, Driveline, and Vehicle, Springer, 2005.