Institute of Architecture of Application Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master Thesis

# Software in the Loop Simulation Environment for eBike ABS

Hithesh Chandra Chandrashekaraiah

| | |
|---|---|
| **Course of Study:** | Information Technology |
| **Examiner:** | Prof. Dr. Ilche Georgievski |
| **Supervisor:** | Dipl.-Ing. Johann Skatulla,<br>Dr.-Ing. Oliver Maier |
| **Commenced:** | December 8, 2022 |
| **Completed:** | June 9, 2023 |

## Abstract

Simulating a system that changes over time is crucial to tuning the performance, checking the system constraints, and assessing it at the prototype stage. Software in the Loop (SiL) simulation is a test setup where the software is running in a virtual environment rather than on the real target hardware. In this case, the system under test is the software, and the virtual environment could be a host computer, a virtual computer, a server, or the cloud.

Generally, in the automotive industry, testing the software on real hardware (an electrified bike (eBike) in our case) is time- and cost-intensive. By using SiL, one can already test the software even before the complete hardware is available. This helps reduce overall software development costs and time. Next to that, it helps to improve the algorithm, and the simulation results can be used as a benchmark to further compare the actual vehicle behavior. One more important aspect to consider here is the overall safety of the software or system. SiL will help verify the safety of the software by simulating scenarios that could potentially be dangerous in vehicle testing. SiL also enables rapid prototyping of the control logic and provides a platform for early validation of the logic in the development lifecycle.

The primary goal of this thesis is to develop a SiL simulation environment for eBike Anti-lock Braking System (ABS) functionality so that it can be used in the development and release phases of the software development lifecycle. To determine the necessary simulation maneuvers for the SiL model, we considered the actual Vehicle Test Catalogues (VTCs) defined for eBike ABS. The work includes modifying the available SiL model to function in a variety of riding situations, such as on surfaces with high friction, low friction, and mue-jump scenarios. Additionally, we derive the Key Performance Indicators (KPIs) and perform a sensitivity analysis by varying certain bike and environment parameters and observing the impact on the derived KPIs. Finally, measurements derived from actual bike testing are used to confirm the effectiveness of the SiL model. Furthermore, the SiL model should include an Inertial Measurement Unit (IMU) sensor model to evaluate ABS performance during IMU sensor errors. By doing so, it is possible to identify the IMU errors that have the greatest impact on performance and the values for every error that are optimal for ABS operation. This enables us to characterize the errors and error values that affect the ABS functionality.

The results demonstrate that the developed SiL model can be used during the development phase for rapid prototyping and provides an option for parameter application of the control software. Additionally, automated execution of maneuvers is developed, through which multiple maneuvers can be executed together, reducing the overall execution time. Furthermore, the IMU errors were identified, and the impact on the ABS function was assessed for a few of the errors related to the accelerometer and gyroscope. We focused on the IMU errors resulting in the incorrect estimation of the pitch angle, which could potentially lead to incorrect ABS activation and compromise vehicle safety.

# Contents

# List of Figures

# List of Tables

# List of Listings

# Acronyms

|  |  |
|---|---|
| **ABS** | Anti-lock Braking System. 3 |
| **ASCET** | Advanced Simulation and Control Engineering Tool. 28 |
| **ASW** | Application Software. 16 |
| **AUTOSAR** | Automotive Open System Architecture. 28 |
| **AV** | Outlet Valve. 22 |
| **COG** | Center of Gravity. 44 |
| **CSSim** | Chassis Systems Simulations. 7 |
| **CSW** | Complete Software. 24 |
| **eBike** | electrified bike. 3 |
| **ECU** | Electronic Control Unit. 19 |
| **EV** | Inlet Valve. 22 |
| **HIL** | Hardware in the Loop. 7 |
| **HSW** | Hardware related Software. 24 |
| **HU** | Hydraulic Unit. 16 |
| **IMU** | Inertial Measurement Unit. 3 |
| **JixEditor** | JAXFront integrated XML editor. 37 |
| **KPI** | Key Performance Indicator. 3 |
| **OOL** | Offline Open Loop. 28 |
| **PSL** | PC Simulation Library. 33 |
| **PSW** | Platform Software. 24 |
| **PWM** | Pulse Width Modulation. 23 |
| **SiL** | Software in the Loop. 3 |
| **VTC** | Vehicle Test Catalogue. 3 |
| **WSS** | WheelSpeed Sensor. 16 |
| **XML** | Extensible Markup Language. 36 |

# 1 Introduction

In this chapter, we outline the motivation for our research and give a brief overview of SiL simulation for eBike ABS function. A set of research objectives for this thesis is also defined with a brief description.

## 1.1 Motivation

Bicycles have grown in prominence over the past decades both as a means of commuting for everyday travel and as a recreational activity. Because of their compact design, they are swift and agile even in congested city traffic. Because of these characteristics, the bicycle market has grown rapidly . Robert Bosch GmbH launched the eBikes in 2011 with an aim to give a rider experience that includes strong and dynamic assistance as well as a high degree of comfort and convenience [Gmb22]. The improvement in comfort over traditional bicycles makes the eBikes appealing, which is reflected in their sales numbers. The number of eBikes sold in Germany increased by over 600% from 0.33 million in 2011 to 2.2 million in 2022 [Zwe23].

To distinguish between pedelecs and speed pedelecs, both are bicycles with an electric motor that assists the rider in addition to the riders input, but the difference here is the maximum supported speed. Pedelecs assist the rider up to a speed of 25 km/h, whereas in case of speed pedelecs the assistance is provided up to a speed of 45 km/h. Although the pedelec is considered a bicycle, the speed pedelec is already classified as a motor vehicle and is thus not allowed on bike lanes in Germany. At these high speeds, the safety of the rider is of utmost importance. As per the accident statistics, 17,000 individuals were injured in eBike accidents in 2021 and this high occurrence of accidents must be reduced to make riding eBikes safe and comfortable. Therefore, it is necessary to have eBikes equipped with ABS, which makes riding safer and allows more people to use this resource-saving form of mobility.

In 2022, Robert Bosch GmbH launched the second generation of ABS for eBikes. ABS is a braking assist system that aims to limit the brake force in an emergency brake situation to a value that results in the shortest possible brake distances while avoiding rollovers and locked wheels, which would result in uncontrollable riding conditions. By reducing the front brake pressure, ABS minimizes front wheel locking on low-friction surfaces and rear wheel lift-up on high-friction surfaces. The advantages of ABS include reduced stopping distance, steerability while braking, maintaining stability while braking on inhomogeneous surfaces, and preventing rear wheel lift-up during hard braking. By equipping all eBikes with an ABS device, up to 29% of all eBike accidents might be prevented [Gmb23a]. Therefore, ABS systems for eBikes are a promising technology with a possibly expanding market.

## 1.2 Research Objectives

The Application Software (ASW) contains the core logic of the eBike ABS function. At the moment, the validation of the ABS software is done through rigorous bike tests in order to tune the parameters and verify the performance in various riding scenarios. However, this takes a significant application effort as there are multiple eBike variants and varying parameters for each eBike type. In addition, thorough bike testing has to be performed in order to validate the ASW during each software delivery, and this is both cost- and resource-intensive. Therefore, to reduce the burden of bike tests, a software-in-the-loop simulation approach is best suited, as one can test at least the basic performance through simulation. Additionally, through SiL, one can perform tests that are dangerous in real bike testing and could be potentially hazardous. SiL provides the flexibility to define numerous maneuvers and execute them automatically, which reduces the need for bike tests for every maneuver. However, reliable simulation models for the hydraulic unit, sensors, and bike model need to be available to enable an efficient SiL environment.

The simulation models for the eBike Hydraulic Unit (HU), bike model, sensor models, and ABS controller software were available prior to the commencement of this thesis. The model for hydraulic unit is taken over from the work of [Bar21] and the bike model is taken over from [Man22]. However, the models need to be coupled together to work in a SiL environment for efficient maneuver creation and automated test execution. Additionally, there is also a simulation framework called CSSim [Gmb11] within Bosch to enable closed loop simulations based on passenger car vehicle model and offers features such as easier maneuver creation, generating results in a readable visual format. However, the CSSim does not include a simulation model of an eBike and thus it is not possible to use CSSim as a SiL environment for eBike directly.

The primary objective of this thesis is to provide a SiL simulation environment so that it is easier to run versions of the control software using the simulation models of HU and bike model. The benefits of such a system includes early detection of bugs and design flaws, especially in the control software and dimensioning, application, and optimization of control software parameters. The goal is to develop and parameterize a simulation model that accurately represents the braking behavior of an actual eBike. The SiL model should simulate longitudinal braking on a flat, straight road with changing down slopes. Furthermore, the SiL model should enable the engineer to define different riding maneuvers (high friction, low friction, mue jumps, and slope), execute these maneuvers automatically, and deliver outputs in a readable data format.

The future generations of Bosch eBike ABS use data from the IMU and WheelSpeed Sensor (WSS) to measure the pitch angle, roll angle, and sideslip angle to improve efficiency during braking in a curve and to prevent pitch-over during braking on high-friction surfaces. The ABS control software, consisting of the state estimation algorithm and the sensor fusion of IMU and WSS, is already available. Furthermore, an IMU sensor model has to be developed, and the accuracy of the already available state estimator logic of the ABS controller has to be assessed during IMU errors. Through the SiL, it is desired to induce the IMU errors into the simulation and then verify the ABS performance when the state variables are estimated incorrectly due to the injected IMU errors. This helps us identify the most relevant IMU errors and their impact on the functioning of the ABS core logic.

From the above-discussed challenges and research objectives related to SiL for eBike ABS, four specific research questions are formulated that need to be explored and analyzed in this thesis, and they are as follows:

[Q1.] How can the simulation maneuvers be defined in SiL and executed in an automated manner?

[Q2.] How reliable are SiL simulation results in comparison with bike measurements?

[Q3.] How do IMU offset errors influence ABS performance?

[Q4.] Which metrics are appropriate for assessing ABS performance in SiL?

## 1.3 Document Structure

The remainder of the document is divided into several chapters, as follows:

Chapter 2 provides the background information required to understand the concepts discussed in this thesis. Chapter 3 discusses the state of the art on topics that are related to or directly addressed in this thesis. The design and implementation of the proposed SiL simulation environment are discussed in Chapter 4. Chapter 5 presents an evaluation of the SiL environment as well as a sensitivity analysis by identifying the KPIs and the impact on these KPIs by varying certain bike and environment-related parameters. Finally, Chapter 6 concludes this thesis and provides an outlook on possible future work.

# 2 Background Information

This chapter presents an overview of several topics that will aid in understanding the technical aspects of this thesis. The typical components of the eBike ABS system are shown first, followed by the mue-slip curve linked to the ABS algorithm. The next parts outline the various concepts related to the hydraulic layout and IMU sensor that are required to comprehend this thesis work.

## 2.1 eBike ABS Components

The base brake in an eBike is identical to any other conventional hydraulic brake systems. The pressure is applied through the brake lever located on the handle bar which is then transmitted to the brake caliper on the wheels. As a result of the applied pressure, a frictional force is generated at the wheel contact point which decelerates the wheels and the eBike. The relevant components for ABS on an eBike are shown in Figure 2.1 which depicts the 1st generation of eBike ABS from Bosch.

The ABS Electronic Control Unit (ECU) consists of the HU which is responsible for brake pressure control. The current generation of eBike ABS is a 1-channel system, so that only the brake pressure at the front wheel can be controlled. Moreover, an ABS ECU is integrated with a pressure sensor which measures the pressure at the front wheel. The inlet and outlet valves of the HU are responsible for pressure build up, pressure holding and pressure reduction. ABS uses the information of brake pressure and wheels speeds to switch the inlet and outlet valves in order to prevent the wheel lock up on the front wheel and lift-up on the rear wheel. One high resolution WSS each on the front and rear wheels measure the respective wheel speeds. Subsequently, the vehicle speed is calculated front the wheel speeds by the ABS controller. A tonewheel is placed on the brake disc of both front and rear wheels and the WSS uses the tonewheel to calculate the wheel rotations.

## 2.2 ABS Braking Process

In the following, straight ahead riding and braking only on the front wheel is considered. The braking process begins with the actuation of the brake lever. Pressure is built up in the master cylinder, which generates the frictional force $F_x$ at the tire contact area. Between the tire and the road surface, the coefficient of friction $\mu$ and the wheel slip $\lambda$ are generated. According to [MW14], the coefficient of friction is calculated with the contact force of the wheel $F_z$ and is given by (2.1).

$$(2.1) \quad \mu = \frac{F_x}{F_z}$$

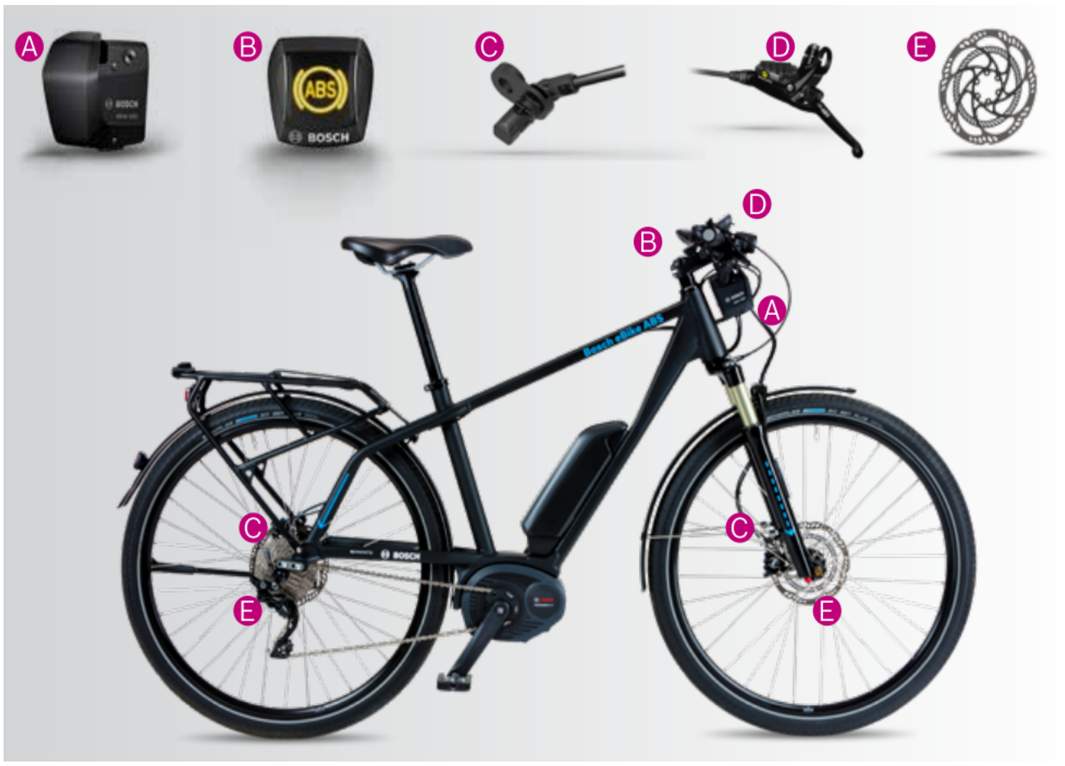


**Figure 2.1:** eBike ABS Components
A- ABS ECU, B - Display Unit consisting of the ABS Lamp, C-ABS hydraulic disc brakes , D-Brake Lever , E- High resolution front and rear wheel speed sensors. Source [Gmb23b]

The slip for braking the front wheel with the rear wheel not braked is calculated from the front ($V_f$) and rear ($V_r$) wheel speed and is given by (2.2).

$$\lambda = \frac{V_r - V_f}{V_r} \tag{2.2}$$

There is a defined relationship between slip and the friction coefficient. This is primarily dependent on the tire characteristics and the road surface. For a dry asphalt road surface and trekking tires, this relationship, which is referred to below as the "Mue-slip curve", is shown in Figure 2.2.

When riding without braking, the front and rear wheels have the same velocity and are moving at the same speed as the bicycle, and the slip is 0. This point lies at the origin of the mue-slip curve. As soon as the brakes are applied, the slip increases. The coefficient of friction increases until it reaches its maximum $\mu_{max}$. Up to the maximum friction value, braking is stable. A larger slip also leads to a larger coefficient of friction value, which in turn means that the vehicle can be braked more strongly. From the critical slip $\lambda_{critical}$, however, the braking becomes unstable. The applied braking torque is greater than the transmittable frictional force. The extreme case quickly occurs that $V_f = 0$ and the slip $\lambda$= 1. The wheel locks and slides. However, the coefficient of friction value depends not only on the slip, but also on the road surface and allows the curve to assume different $\mu_{max}$ values, as well as different flattening. Figure 2.3 shows the mue-slip curve

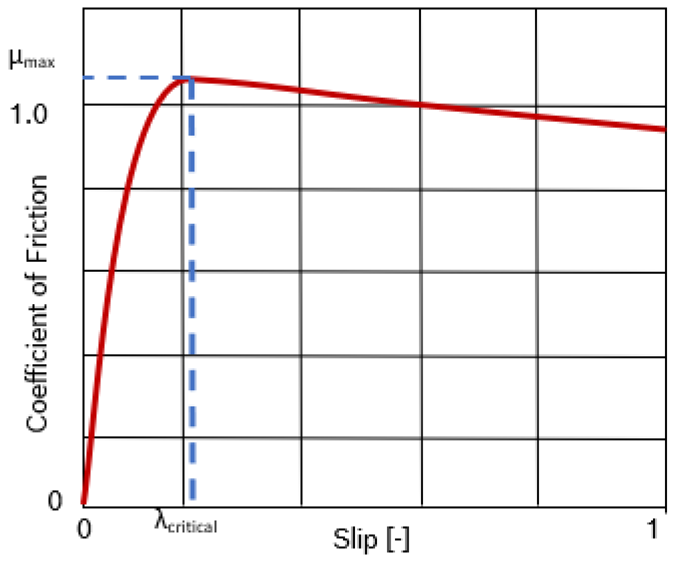


**Figure 2.2:** Mue Slip curve for a dry asphalt surface and trekking tires

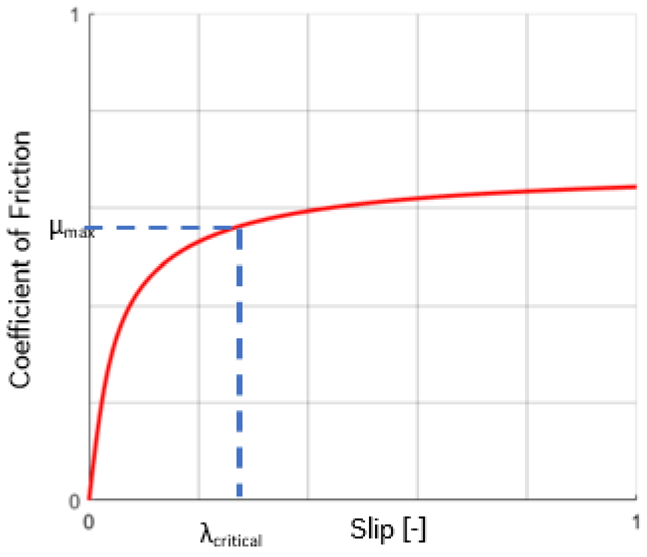


**Figure 2.3:** Mue Slip curve for a gravel surface and trekking tires

for a gravel surface and trekking tires. The aim of ABS-assisted braking is to set a slip so that the maximum coefficient of friction $\mu_{max}$ is reached in order to achieve maximum and at the same time stable braking without locking the wheels.

## 2.3 Hydraulic Layout of eBike ABS

Figure 2.4 depicts the hydraulic layout of ABS . The ABS ECU consisting of the hydraulic unit is mounted along the brake line between the brake lever and the brake caliper. The HU consists of an Inlet Valve (EV), an Outlet Valve (AV) and an accumulator. There is also a pressure sensor within the HU.

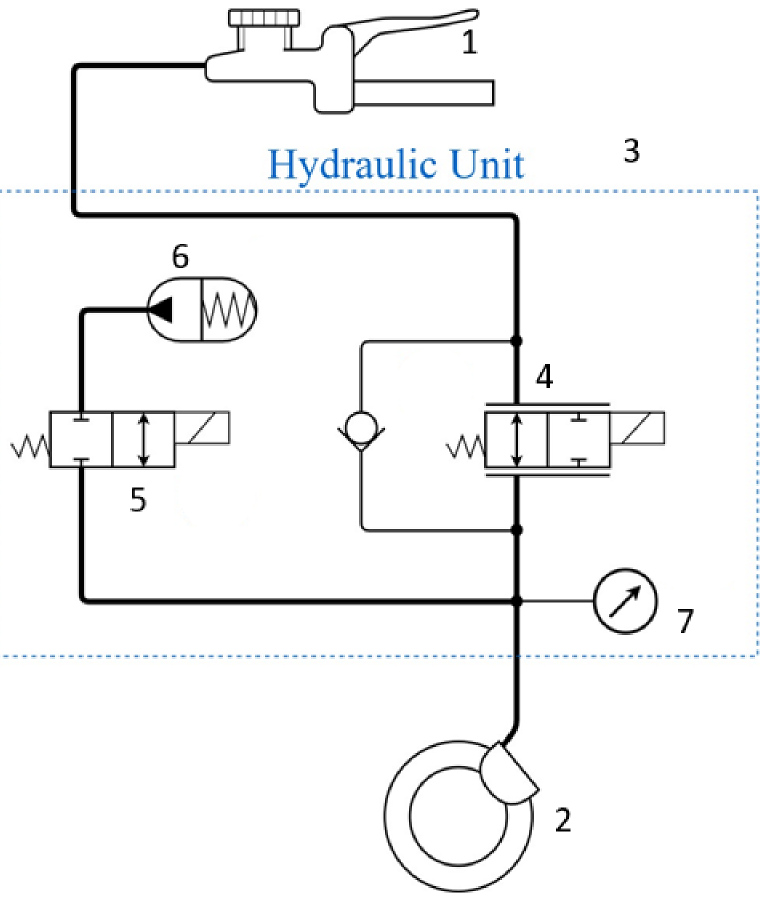


**Figure 2.4:** Hydraulic Layout of eBike ABS
Hydraulic Layout of the eBike HU (3), consisting of the inlet valve (4), the outlet valve (5), the accumulator (6) and a pressure sensor (7). The brake lever with master cylinder(1), as well as the brake caliper with brake disc (2) can also be seen.

When the wheel slip becomes too large, the EV is closed, and no further pressure buildup is possible inside the HU. If the actuation pressure is reduced below the pressure level in the brake caliper, the check valve will open and allow brake fluid from the HU to flow back into the master cylinder. Thus, the pressure at the brake caliper is never greater than the actuation pressure. Pressure hold is illustrated in Figure 2.5 (a). While both the EV and the AV are closed and the pressure is maintained, the braking force acting on the front wheel is constant. If the slip exceeds the critical slip value of

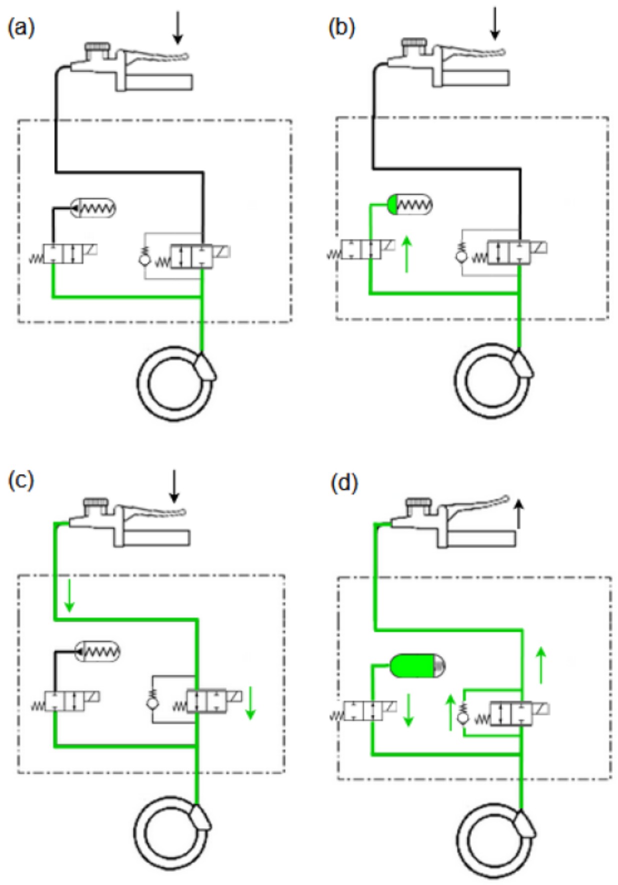


**Figure 2.5:** Phases of ABS control cycle
Phases of the ABS control cycle, including pressure hold (a), pressure reduction (b), pressure buildup (c), and emptying the accumulator (d)

$\lambda_{critical}$ according to Figure 2.2 and is to be reduced, the AV is opened. The pressure between the EV and the accumulator is increased. The pressure between EV and AV decreases, as does the braking torque applied to the front wheel. This pressure reduction can be seen in Figure 2.5 (b). It should be noted that the accumulator can only hold a limited volume of brake fluid. Thus, the number of control cycles that can be carried out is limited. As soon as the volume of the accumulator is full, it is no longer possible to reduce the brake pressure.

Starting from the pressure reduction phase, the AV can be closed again as soon as the critical slip value is below the threshold, and it is possible to build up the pressure again by opening the EV, which starts a new control cycle. A Pulse Width Modulation (PWM) control triggers the EV. This pulsed opening of the EV allows brake fluid to flow from the master cylinder and increases the pressure at the brake calipers. The pressure buildup can be seen in Figure 2.5 (c). After the end of the ABS braking cycle, the accumulator must be emptied. The draining is done via the non-return valve of the EV. For this, both the AV and the non-return valve must be open. If the rider applies the brake while the accumulator is being emptied, a new braking cycle begins with a non-empty accumulator and thus with a reduced number of possible control cycles. Emptying the accumulator is illustrated in Figure 2.5 (d).

## 2.4 Vehicle Test Catalogues

The Vehicle Test Catalogue (VTC) is a list of test cases with riding maneuvers for testing the functionality and performance of the ABS software within the complete vehicle system (SW-Verification and SW-Validation). The goal of VTC is to verify the functional behavior on a test bike and further tune the parameters accordingly if needed for further performance enhancement. A VTC contains the description of steps for the rider to reproduce the scenario, expected observation for that test-case and additionally it may contain few internal signals that needs to be considered for evaluation by the concerned function developer. Furthermore, the VTC also contains the environment conditions like the surface used for testing and the slope of the road surface. Table 2.1 shows as example of an eBike ABS VTC description. Here test case description consists of the steps to perform the test and expected results and measurement keys are used to evaluate the result.

| **Test Name** | **Topic** | **Test Case Description** | **Expected Result** | **Measurement Key** |
|---|---|---|---|---|
| ABS Test | ABS start timing<br>on high friction surfaces<br>with slow gradient brake input | Speed at start of braking : 30kph<br>Surface: Dry asphalt ($\mu = 0.8 - 1.0$)<br><br>Brake apply:<br>Slow brake input to activate ABS | 1) Not start ABS control early, ex. Early case :<br>- Before enough slip occurred<br>- Before rear lift up occurred<br>2) ABS control down to 7kph<br>3) Check the deceleration level just before ABS started. | Signal to evaluate:<br>1. SystemRelease_MC<br>2. TargetStateABS<br>4.Current_ABS_Mode<br>5.ABSOFF_Active |

**Table 2.1:** A sample VTC description

## 2.5 Software Architecture

It is important to understand the software architecture generally used in the automotive domain and is depicted in Figure 2.6. The Complete Software (CSW) comprises of the ASW and Platform Software (PSW). The PSW has the Hardware related Software (HSW) and Base Software BSW. The HSW consists of the drivers and software related to hardware abstraction whereas the BSW deals with memory management, network communications and diagnosis software. The ASW has the controller code for the ABS controller in our case. In this thesis we focus on the SiL simulation where the software is only the ASW and not the CSW as our primary focus is to enable SiL to test the core ABS controller logic and not the complete CSW.

## 2.6 Inertial Measurement Unit

IMUs are electronic devices that measure angular velocity and linear accelerations, and they are used in navigation systems, robotics, and automotive applications. They typically include gyroscopes and accelerometers to provide accurate measurements. In an eBike, the state estimator logic of the ABS controller uses the information from the IMU and WSS to compute pitch angle and lean angle using a sensor fusion algorithm. The Figure 2.7 shows the angular velocities and linear accelerations measured from an IMU sensor. The angular velocities are measured from the gyroscope and linear accelerations from the accelerometer. Angular velocity are given by roll Rate ($\Omega_X$), pitch rate ($\Omega_Y$), and yaw rate ($\Omega_Z$), while the linear accelerations are given by longitudinal acceleration ($a_X$ ), lateral acceleration ($a_Y$), and vertical acceleration ($a_Z$). The pitch angle is the angle around the Y axis, the roll angle is the angle around the X axis and yaw is the angle around Z axis.

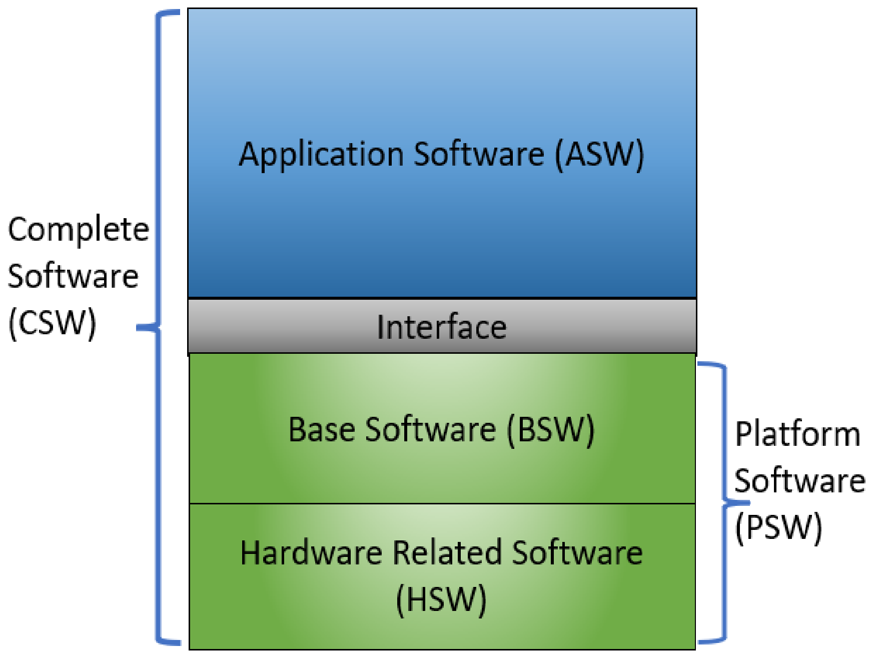


**Figure 2.6:** Software Architecture to differentiate ASW and CSW

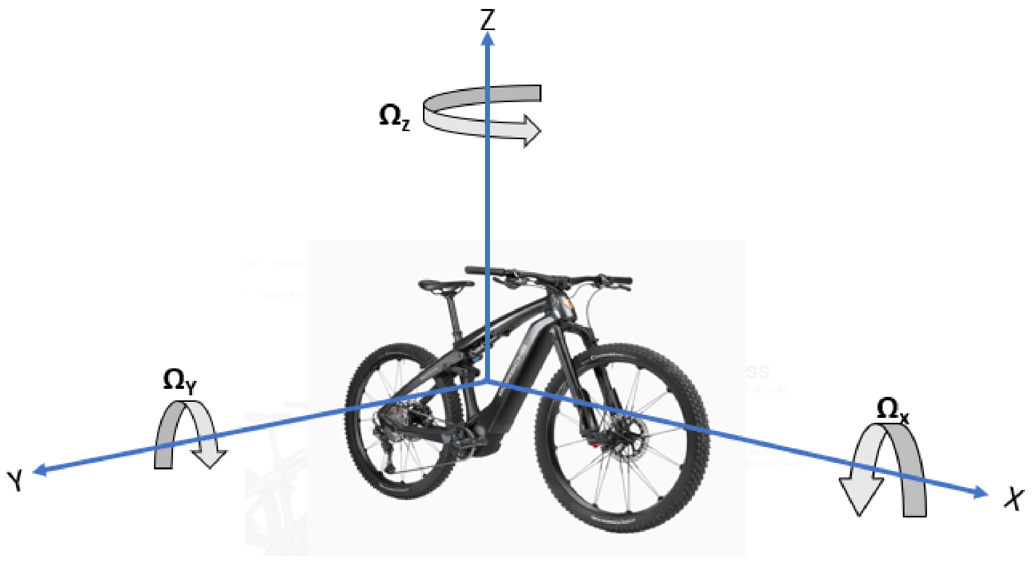


**Figure 2.7:** Angular velocities and linear accelerations of an IMU

# 3 State of the Art

This chapter gives an overview of the various testing methodologies used in the automotive industry. We talk about what these testing methodologies are for and how they differ from our use case. In addition, the idea of SiL simulation is discussed, as are the SiL tools utilized within Bosch.

## 3.1 Automotive Software Testing Methods

Numerous testing methodologies exist in the market to test the automotive software. Recent survey shows that the methods that rely on executable including simulation, actual vehicle, and hardware-in-the-Loop predominate in the automotive industry [AWS14].

Testing the software on the vehicle is essential to validate and check the performance as per the customer requirements. However, the testing on vehicle is done at a later stage in the development life-cycle and is resource intensive in terms of both cost and time. Moreover, the software should be reliable, because a failure during vehicle testing could be catastrophic and hence vehicle testing is not best suited for rapid prototyping where new functionalities are developed and reliability of software cannot be fully ensured. Furthermore, in the case of eBikes, there are a lot of eBike variants with different varying components like brake types, sensor types etc. and this causes a tremendous effort to flash and test the software and hence the vehicle tests are suitable during software release phase rather than during rapid prototyping phase.

As a result of these limitations of vehicle testing, HIL testing is being utilized to test automotive software components rather than actual vehicles. In an HIL test, ECU under test is given simulated inputs from a vehicle, sesnors and environment to make it appear as though it is reacting to real-world driving conditions. The HIL testing platform encompasses all the essential components of the vehicle. The simulator transmits signals to actual ECU, which subsequently transmit signals back to the system being tested, in order to assess its ability to respond suitably to the given inputs [Apt]. ETAS LABCAR [ETAb] and dSpace [dSp] are few examples for HIL systems in automotive domain. Figure 3.1 shows the top level architecture of a HIL system.

The important aspect to consider is that the HIL testing is typically a CSW integration testing in which the software under test is CSW (ASW+PSW) in addition to the real hardware. Although HIL is used to test the software, the primary objective of HIL is to verify the hardware unit or ECU. But, what we want to explore in this thesis is testing only the controller software (ASW) to enable a rapid prototyping of new functionalities. Additionally, HIL test setups are expensive and tests are conducted during the release phase of the software lifecycle. To perform HIL tests, the ECU hardware must be available along with the sensors and actuators needed for the project. On the contrary, hardware comes later in the project development process and hence HIL is not best suited

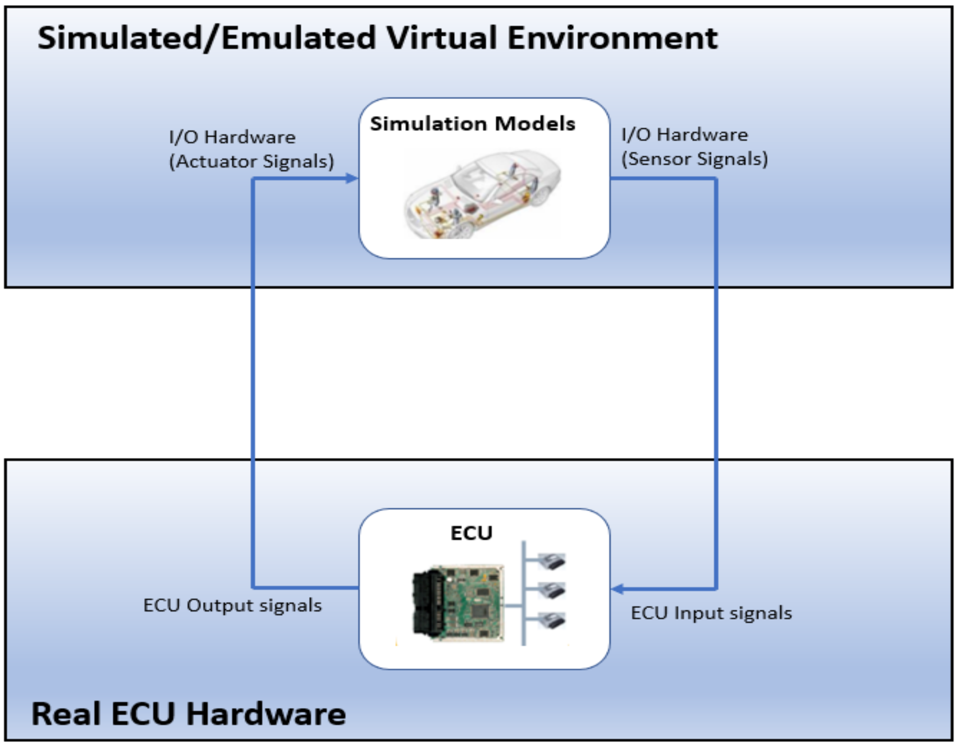

Simulated/Emulated Virtual Environment
I/O Hardware (Actuator Signals)
Simulation Models
I/O Hardware (Sensor Signals)
ECU
ECU Output signals
ECU Input signals
Real ECU Hardware

**Figure 3.1:** Generic HIL concept

to test the controller software in the early stages of development. Therefore it is necessary to find testing techniques that can be used during early development stages to perform rapid prototyping of the control software.

SiL testing is an efficient way to verify the control software during development or design phase. The generic SiL concept is shown in Figure 3.2. The software application which is the system under test is in a closed loop feedback path with the other simulation models such as the hardware unit, vehicle unit and sensors/actuators. The entire system is run on a virtual environment which could be a host computer, server or cloud. In the automotive Industry, several SiL based methods are used to enable rapid prototyping. In one such work [JKL16] authors propose a SiL simulation environment to test the communication between Automotive Open System Architecture (AUTOSAR) [AUT] components. A VFB (virtual function Bus) layer of AUTOSAR is simulated to test the source code and this simulated VFB layer is used to demonstrate the behaviour and interaction of different software components without hardware such as ECU. However, the proposal drifts from our focus which is to use SiL testing to validate the ASW and not the communication between AUTOSAR components.

## 3.2 SiL tools used in Bosch

There are already a few tools used in Bosch that offers SiL capabilities to some extent. One such tool is Advanced Simulation and Control Engineering Tool (ASCET) which offers Offline Open Loop (OOL) simulation environment. ASCET is a diagram and model based development tool from

ETAS [ETAa] and it used to model the control software functionalities. A generic OOL simulation setup is described in Figure 3.3 and the basic idea of OOL is to perform a logical verification of the ASW function by providing the inputs via vehicle measurements or artificial signal data. The offline experiment tool allows to test changes within the algorithm's parameters, logic or even with whole new classes or modules. Although ASCET OOL helps in rapid prototyping, it does not involve simulation models for sensors, actuators, and vehicle and this limits the usage only to a logical verification of the ASW. Moreover, OOL is an open loop simulation where there is no feedback path. The lack of feedback can make the simulation less accurate or realistic, but it can also make it simpler and easier to use. On the other hand, in a closed loop simulation, there is a feedback path that allows for adjustments and corrections. In an open loop system, the output is determined solely by the input, while in a closed loop system, the output is influenced by both the input and feedback. Figure 3.4 shows the general concept of open and closed loop systems.

**Chassis Systems Simulation**

CSSim [Gmb11] is a Simulink [Mat] based framework that enables SiL simulation. It simulates the dynamic behaviour of vehicles with focus on brake system development with embedded control algorithms. Figure 3.5 shows the top level architecture of CSSim. Models of brake system, vehicle, power train and electronics are included. Controller variable measurement and controller parameter modification is supported. On the other hand, CSSim doesn't offer the models with respect to the eBikes currently. The vehicle model, hydraulics and sensors are related to the passenger car

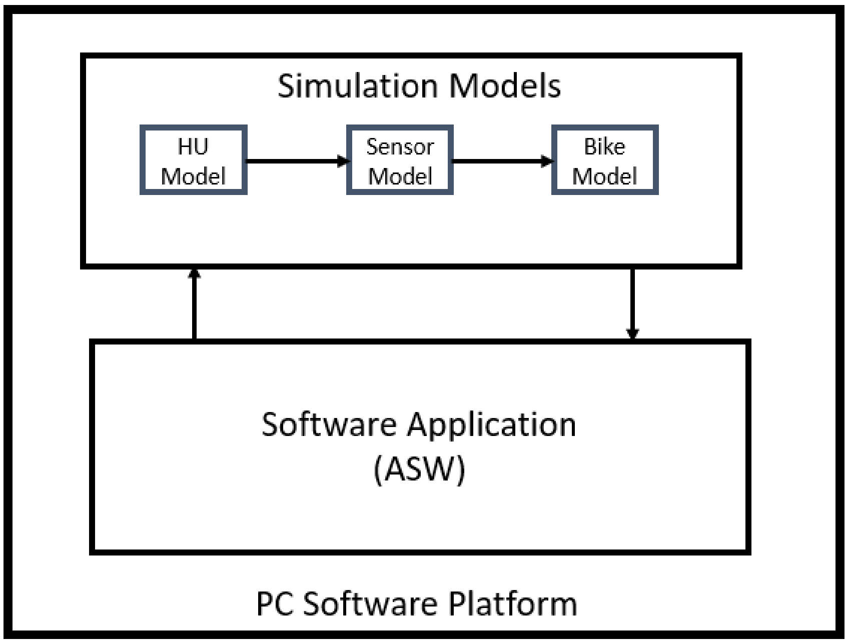


**Figure 3.2:** Generic SiL Concept

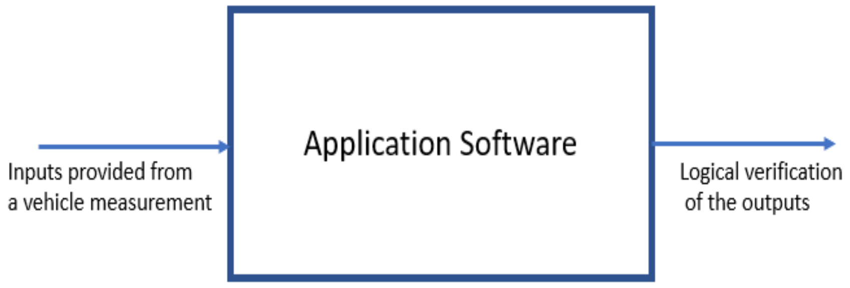

Application Software
Inputs provided from a vehicle measurement
Logical verification of the outputs

**Figure 3.3:** Generic OOL concept

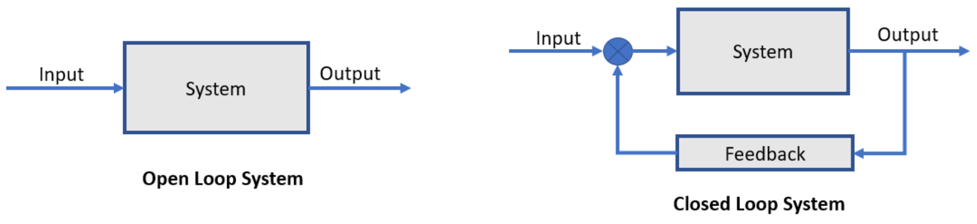


**Figure 3.4:** Concept of open loop and closed loop systems

and cannot to be used for our use-case. But, the framework of CSSim in general can be used by replacing the simulation models with eBike components and adapting accordingly to enable a SiL environment to test eBike ABS functionalities.

In [FGW18], the authors designed a simulation model to verify the effectiveness of different sensor fusion algorithms. The work included simulating the sensor errors, propogating these errors to the bicycle rider assistance system, and analyzing the impact on the riding dynamics. However, in our work, we already have the fixed sensor fusion algorithm for eBike ABS, and the aim is to identify the IMU related errors and their impact on the ABS braking performance. While performing real bike tests to evaluate the impact of IMU errors could be potentially dangerous, a SiL approach is best suited for this purpose. This approach allows for the simulation of various scenarios and the testing of the algorithm under different conditions without putting anyone at risk. By using this approach, developers can fine-tune the algorithm and ensure that it performs optimally in all situations, providing riders with a safe and reliable braking system.

To summarize, the available test strategies and SiL environments do not fully cover our use case, which is to have a SiL environment to use both in the design and release phases of the control software development. Therefore, here we focus on developing a SiL simulation environment using the framework of CSSim that enables ASW validation in both design and release phases

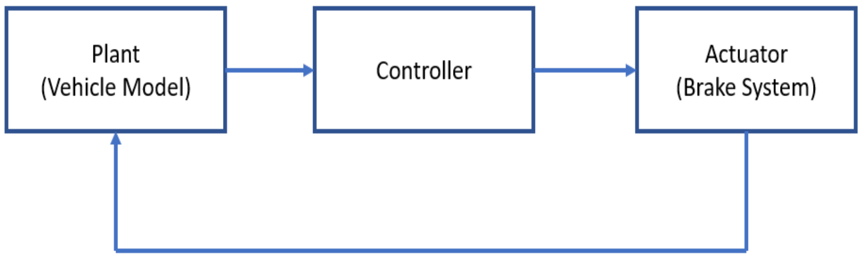


**Figure 3.5:** CSSim Top Level Architecture

by providing the option to define riding scenarios for eBike, executing the maneuvers, and finally having the end result in a readable format. Additionally, the SiL environment will have a feature to execute the maneuvers automatically and generate the results in a readable format. Furthermore, SiL can be used to evaluate the ABS state estimator logic and the braking performance by inducing IMU errors through simulation.

# 4 Design and Implementation of SiL environment

In this chapter, we discuss the design and implementation of the SiL simulation environments. Basically, we design two simulation environments: one using the framework of CSSim to create riding maneuvers and execute them automatically, and another SiL environment that incorporates the IMU sensor model to assess the ABS performance during IMU offset bias errors. The SiL with CSSim framework can be utilized to run the VTCs, while the SiL model with IMU can be utilized to test future ABS developments by inducing IMU errors.

## 4.1 Design of SiL environment with CSSim framework

### 4.1.1 Architecture

Figure 4.1 depicts the top-level architecture of the proposed SiL model. It provides the flexibility to define several maneuvers. The maneuvers are defined by using the actual VTC definition for eBike ABS functionality. Furthermore, the maneuvers serve as the foundation for automated test execution, in which the maneuvers are executed in batches in an automated manner. Furthermore, the SiL model is composed of various subsystems for auxilarries, the ABS controller, the hydraulic unit of ABS, and the eBike bike model. The SiL model's output will be in D97 format, which is frequently used by the engineers at Bosch to analyze signals.

The auxiliaries and scopes subsystem includes predefined scopes for online evaluation of simulation results as well as a merge process for merging environment signals into the final result file. The ABS controller is exported as an ASCET PC Simulation Library (PSL) export in the controller subsystem. The PSL is an ASCET library that allows one to use ASCET code in Matlab as a sFunction type. Before generating the PSL export, the relevant inputs and outputs for the controller should be configured in the ASCET tool. In addition, ASCET PSL should be provided with a suitable parameter file containing the ABS controller parameters for use within Matlab. The HU model of the eBike ABS system makes up the hydraulic unit subsystem. The HU model is based on the work of [Bar21], in which the author designed and parameterized the eBike HU model. The components of the HU are explained in Section 2.3. In addition, the HU subsystem is divided into lower-level subsystems such as sensor models, power electronics models, and the HU model. The eBike bike model is part of the bike model subsystem. The bike model is based on the work of [Man22], where the author developed and parametrized the bike model to act as an eBike. Furthermore, the bike model is further adapted in our work to simulate different maneuvers, including mue-jump scenarios. The parameters for the bike model and HU are loaded into the workspace by means of a post-load function callback once the simulation model has been successfully loaded and is error-free.

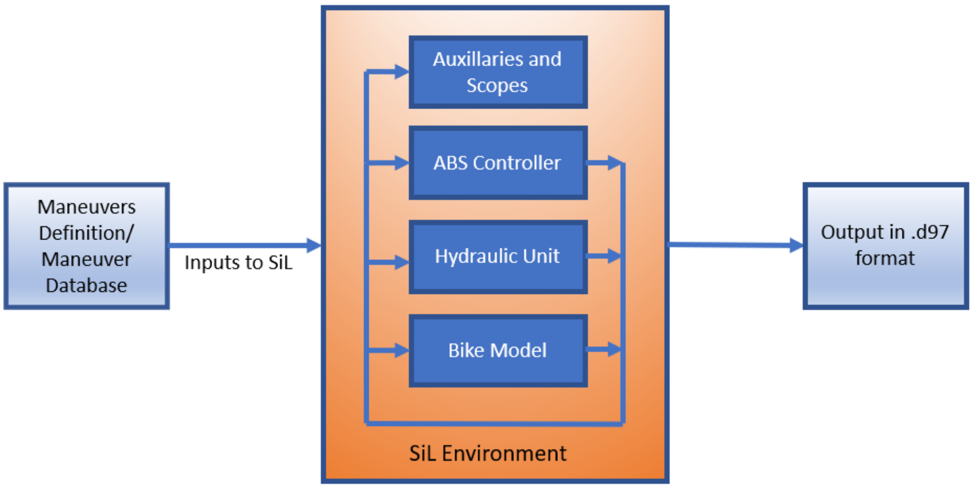


**Figure 4.1:** Architecture of the proposed SiL environment

### 4.1.2 Maneuver Database

Maneuver creation is essentially providing the inputs for the simulation environment and includes the definition of inputs that change with maneuvers. To define the simulation maneuvers, we referred to the standard VTCs defined for the eBike ABS. Based on the simulation model's capabilities, only the maneuvers for the dry asphalt (Figure 4.2b) and gravel (Figure 4.2a) surfaces are considered, as tire characteristics for other types of surfaces (e.g., roughroad, bump, sand, grassland, etc.) are not currently available. Furthermore, VTCs involving riding downhill and four variations of brake inputs, namely spike, jerk, medium gradient, and slow gradient, are included in our maneuver definitions. The Table 4.1 lists the maneuvers that were taken into account during the simulation and the respective names of the simulation maneuvers.

**(a)** Gravel Surface

**(b)** Asphalt Surface

**Figure 4.2:** Gravel and Dry asphalt surface

The primary inputs relevant for the simulation environment are as follows.

1. Pressure input from the rider. (Front and rear brake pressure)
2. Down Slope Percentage
3. Type of the surface (high friction, low friction, high-to-low transition, low-to-high transition)
4. Simulation duration
5. Initial speed.

| Test Name | Test case description | Expected result | Simulation maneuvers defined in SiL |
|---|---|---|---|
| ABS start timing on<br>high friction surfaces<br>with slow gradient<br>brake input | Speed at start of braking : 30kph<br>Surface: Dry asphalt (μ = 0.8 - 1.0)<br><br>Brake apply:<br>Slow brake input to activate ABS<br>a) Front only<br>b) Both (Rear = Rider's best) | 1) Not start ABS control early, ex. Early case :<br>- Before enough slip occurred<br>- Before rear lift up occurred<br>2) ABS control down to 7kph<br>3) Check the deceleration level just before ABS started. | ABS_HighMue_FrontOnly_SlowGradient_ZeroSlope<br>ABS_HighMue_FrontandRear_SlowGradient_ZeroSlope |
| ABS start timing<br>on high friction surfaces<br>with jerkbrake input | Speed at start of braking : 30kph<br>Surface: Dry asphalt (μ = 0.8 - 1.0)<br><br>Brake apply:<br>Jerk brake input to activate ABS<br>a) Front only<br>b) Both (Rear = Rider's best) | 1) Not start ABS control early, ex. Early case :<br>- Before enough slip occurred<br>- Before rear lift up occurred<br>2) No sudden or big lift rear wheel<br>3) ABS control down to 7kph | ABS_HighMue_FrontOnly_Jerk_ZeroSlope<br>ABS_HighMue_FrontandRear_Jerk_ZeroSlope |
| Medium gradientbrake input<br>on gravelsurfaces | Speed at start of braking : 30kph<br>Surface: Dry gravel (μ = 0.3 - 0.5)<br><br>Brake apply:<br>Medium brake input to activate ABS<br>a) Front only<br>b) Front and Rear | 1) No deep slip or high lift-up<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) Vehicle is stable with enough deceleration.<br>5) ABS control down to 7kph | ABS_Gravel_FrontOnly_MediumGradient_ZeroSlope<br>ABS_Gravel_FrontandRear_MediumGradient_ZeroSlope |
| Spikebrake (high gradient brake input)<br>on high friction surfaces | Initial Speed : 50kph<br>Speed at start of braking : 48kph<br>Surface: Dry asphalt (μ = 0.8 - 1.0)<br><br>Brake apply:<br>Spike brake input to activate ABS<br>a) Front only<br>b) Front and Rear | 1) No deep slip<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) ABS control down to 7kph<br>5) Check the deceleration level on test case e). | ABS_HighMue_FrontOnly_Spike_ZeroSlope<br>ABS_HighMue_FrontandRear_Spike_ZeroSlope |
| Spikebrake (high gradient brake input)<br>on gravel road surfaces | Speed at start of braking : 30kph<br>Surface: Dry gravel (μ = 0.3 - 0.5)<br><br>Brake apply:<br>Spike brake input to activate ABS<br>a) Front only<br>b) Front after Rear | 1) No deep slip or high lift-up<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) Vehicle is stable with enough deceleration.<br>5) ABS control down to 7kph | ABS_Gravel_FrontOnly_Spike_ZeroSlope<br>ABS_Gravel_FrontandRear_Spike_ZeroSlope |
| Medium gradientbrake input<br>on high friction surfaces<br>with high speed | Speed at start of braking : 70kph<br>Surface: Dry asphalt (μ = 0.8 - 1.0)<br><br>Brake apply:<br>Medium brake input to activate ABS<br>a) Front only<br>b) Front after Rear | 1) No deep slip or high lift-up<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) Vehicle is stable with enough deceleration.<br>5) ABS control down to 7kph | ABS_HighMue_FrontOnly_MediumGradient_ZeroSlope<br>ABS_HighMue_FrontandRear_MediumGradient_ZeroSlope |
| Medium gradientbrake inputon<br>high frictionsurfaces down slope | Initial Speed: 45kph<br>Speed at start of braking : 40.5kph<br>Surface: Dry asphalt (μ = 0.8 - 1.0)<br>Slop : Down -20%<br><br>Brake apply:<br>Medium brake input to activate ABS<br>a) Front only<br>b) Front after Rear | 1) No deep slip or high lift-up<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) Vehicle is stable with enough deceleration.<br>5) ABS control down to 7kph | ABS_HighMue_FrontOnly_MediumGradient_20%Slope<br>ABS_HighMue_FrontandRear_MediumGradient_20%Slope |
| Medium gradientbrake input on<br>gravel surfaces down slope | Speed at start of braking : 30kph<br>Surface: Dry gravel (μ = 0.3 - 0.5)<br>Slop : Down -10%<br><br>Brake apply:<br>Medium brake input to activate ABS<br>a) Front only<br>b) Front after Rear | 1) No deep slip or high lift-up<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) Vehicle is stable with enough deceleration.<br>5) ABS control down to 7kph | ABS_Gravel_FrontOnly_MediumGradient_10%Slope<br>ABS_Gravel_FrontandRear_MediumGradient_10%Slope |
| Spikebrake (high gradient brake input)<br>on high frictionsurfaces<br>down slope | Speed at start of braking : 15kph<br>Surface: Dry asphalt (μ = 0.8 - 1.0)<br>Slop : Down slop -20%<br><br>Brake apply:<br>Spike brake input to activate ABS<br>a) Front only<br>b) Front after Rear | 1) No deep slip or high lift-up<br>2) Enough decrease pressure<br>3) No deceleration loss<br>4) Vehicle is stable with enough deceleration.<br>5) ABS control down to 7kph | ABS_HighMue_FrontOnly_Spike_20%Slope<br>ABS_HighMue_FrontandRear_Spike_20%Slope |

**Table 4.1:** VTCs considered and respective simulation maneuvers defined

Section 4.2.1 describes the details of the defined inputs in the maneuver file and their usage within the simulation environment.

### 4.1.3 Automated VTC execution

The primary idea to enable automated VTC execution is to create batch items and link maneuvers with the respective batch items. For demonstrating automated test execution, the CSSim environment provides an Extensible Markup Language (XML) template. Figure 4.3 provides a brief summary of the methodology used to execute automated tests. The batch items are executed sequentially. Batch 2 can begin only once Batch 1 has been finished. The maneuvers associated with the batch items, on the other hand, are executed in parallel. The path to the simulation model file and maneuver definition files should be provided for each batch item.

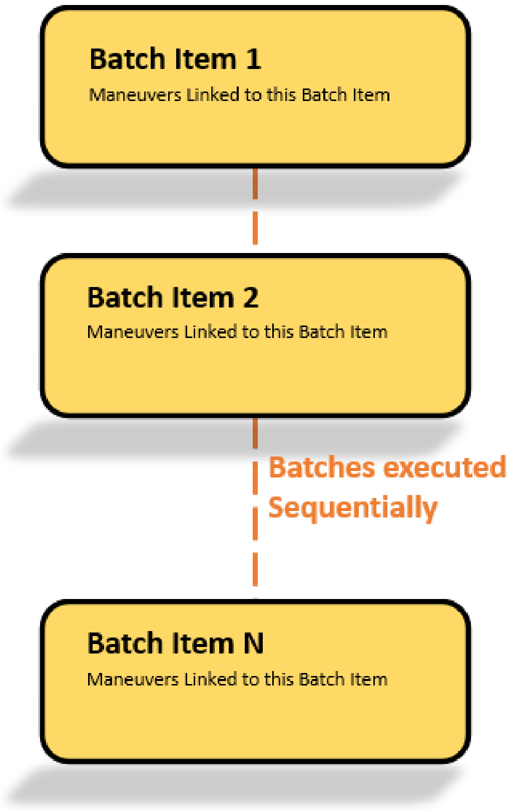
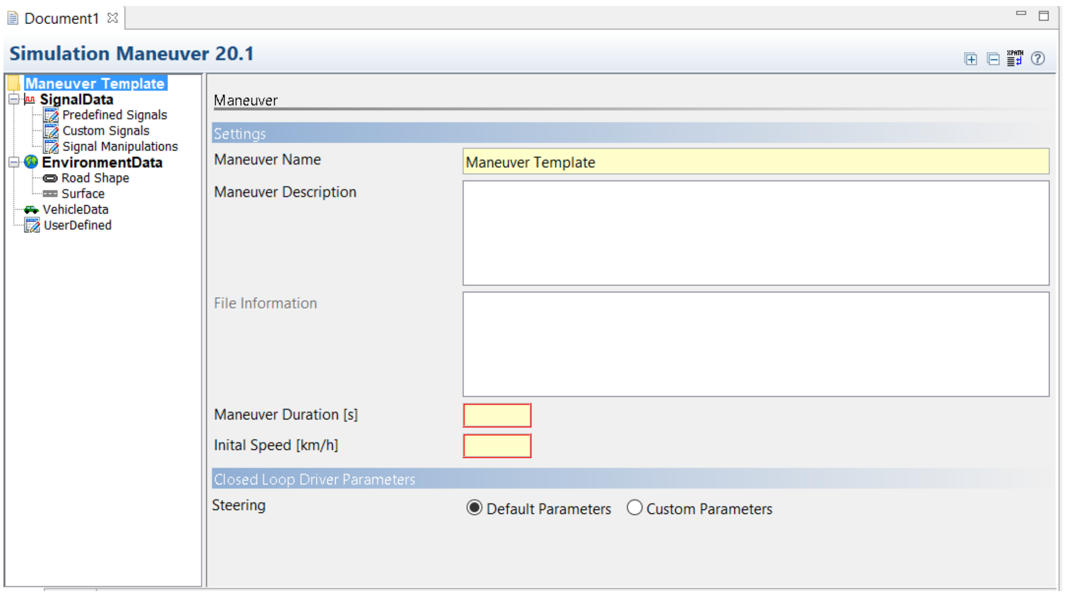

Batch Item 1
Maneuvers Linked to this Batch Item
Batch Item 2
Maneuvers Linked to this Batch Item
Batches executed Sequentially
Batch Item N
Maneuvers Linked to this Batch Item

**Figure 4.3:** Concept of Batch Execution

## 4.2 Implementation of SiL Environment with CSSim framework

### 4.2.1 Maneuver Creation

Maneuver definition is done by using one of the templates offered by JAXFront integrated XML editor (JixEditor), a Bosch internal tool. JixEditor is an XML editor that enables form-based and textual editing of XML documents. The JixEditor provides several templates for usage within Bosch. We use the simulation maneuver template of the JixEditor to define simulation maneuvers. The template is as shown in Figure 4.4.



Maneuver Name: Maneuver Template

Maneuver Description:

File Information:

Maneuver Duration [s]:

Inital Speed [km/h]:

Steering: [✓ Default Parameters] Custom Parameters

**Figure 4.4:** Simulation Maneuver Template

A maneuver template should be provided with a mandatory maneuver name and an optional maneuver description. Maneuver duration and initial speed are mandatory data that needs to be provided. The signal data field defines the primary inputs for the simulation environment, such as the input pressure on the front and rear wheels, and additional signals to manipulate sensor signals by adding noise, gain, and offset. The environment data field specifies the environment inputs such as the surface friction, slope of the road, etc. Additionally, the template also offers the option to specify additional vehicle-related data such as tire size, trailer load, etc.

#### Input Definitions in the Maneuver File

**1. Pressure Input on Front and Rear wheels:** The maneuver file defines the input pressure signals on the front and rear wheels that correspond to the rider's applied pressure. They vary depending on the type of brake applied (spike brake, medium gradient brake, slow gradient brake, and jerk brake). Figure 4.5 shows the four types of brake inputs used in our maneuver definitions. In a medium gradient, brake pressure increases with a gradient of 200 bar/s to reach the maximum pressure of 100 bars in 0.5 seconds. During spike braking, the gradient is 600 bar/s and reaches 100 bars in 0.17 seconds, whereas in a slow gradient, the maximum pressure of 100 bars is reached in 3.3 seconds.

During jerk braking, in the initial 0.2 seconds of the braking, we have a gradient of 200 bar/s, and later it is 600 bar/s. Table 4.2 shows the types of brake inputs, the respective gradient values, and the time required to reach the maximum pressure of 100 bars.

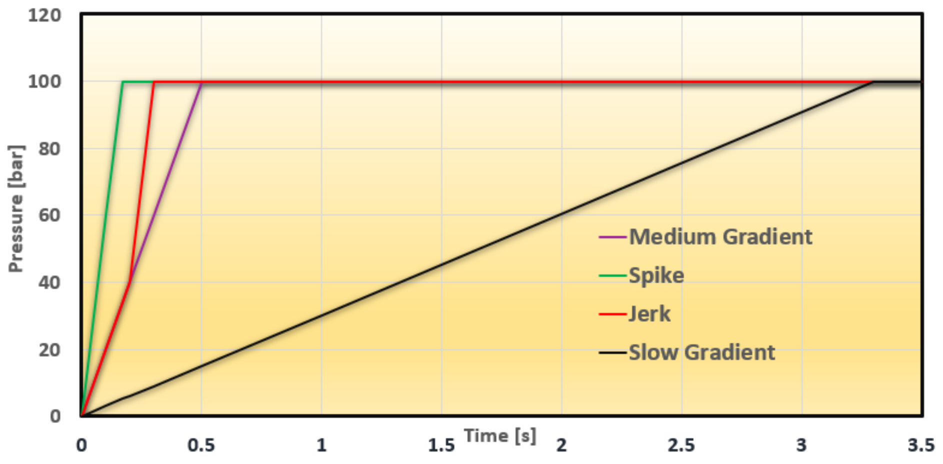


**Figure 4.5:** Different types of Pressure Inputs defined in Maneuvers

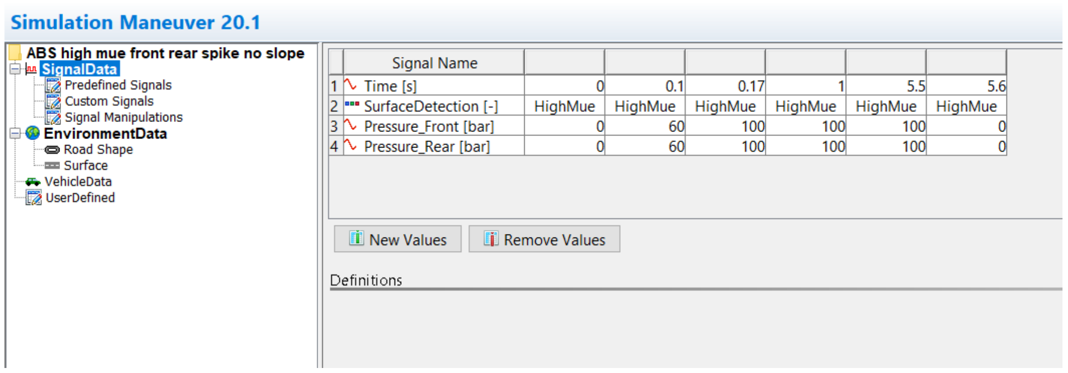
| Type of Brake Input | Gradient [bar/s] | Time to reach maximum pressure of 100 bars [seconds] |
|---|---|---|
| Slow Gradient | 30 | 3.3 |
| Medium Gradient | 200 | 0.5 |
| Spike Brake | 600 | 0.17 |
| Jerk Brake | 200 –>600 | 0 to 40 bar in 0.2 seconds<br>40 to 60 bar in next 0.1 second |

**Table 4.2:** Brake Inputs and Gradient Values

The *Pressure_Front* and *Pressure_Rear* signals correspond to pressure applied on the front and rear wheels, respectively, and the definition in the simulation maneuver is done as shown in Figure 4.6. The definition is done in the *SignalData* field of the maneuver template. The signal *time* corresponds to the duration of the simulation maneuver; for instance, if the simulation duration is 7 seconds, then the signals can be varied up to a time of 7 seconds.

The simulation maneuver template offers a set of predefined signals to define the maneuver, but these signals are related to passenger cars and are not relevant to our use case for eBike. Therefore, we define the pressure signals as custom signals by providing details such as name, unit, and signal type. Figure 4.7 shows the custom signal definition of the *Pressure_Front* signal, which has a unit of bar and is a continuous signal. The *Pressure_Rear* signal is also defined similarly to *Pressure_Front*.

**2. Surface Detection:** The *SurfaceDetection* signal specifies the type of road surface that is being simulated in the simulation maneuver. Based on the kind of surface, the tire characteristics vary, as does the braking performance. Figure 4.6 shows the *SurfaceDetection* signal definition in the simulation maneuver. This signal is added as a custom signal, and its definition is shown in Figure



Simulation Maneuver 20.1

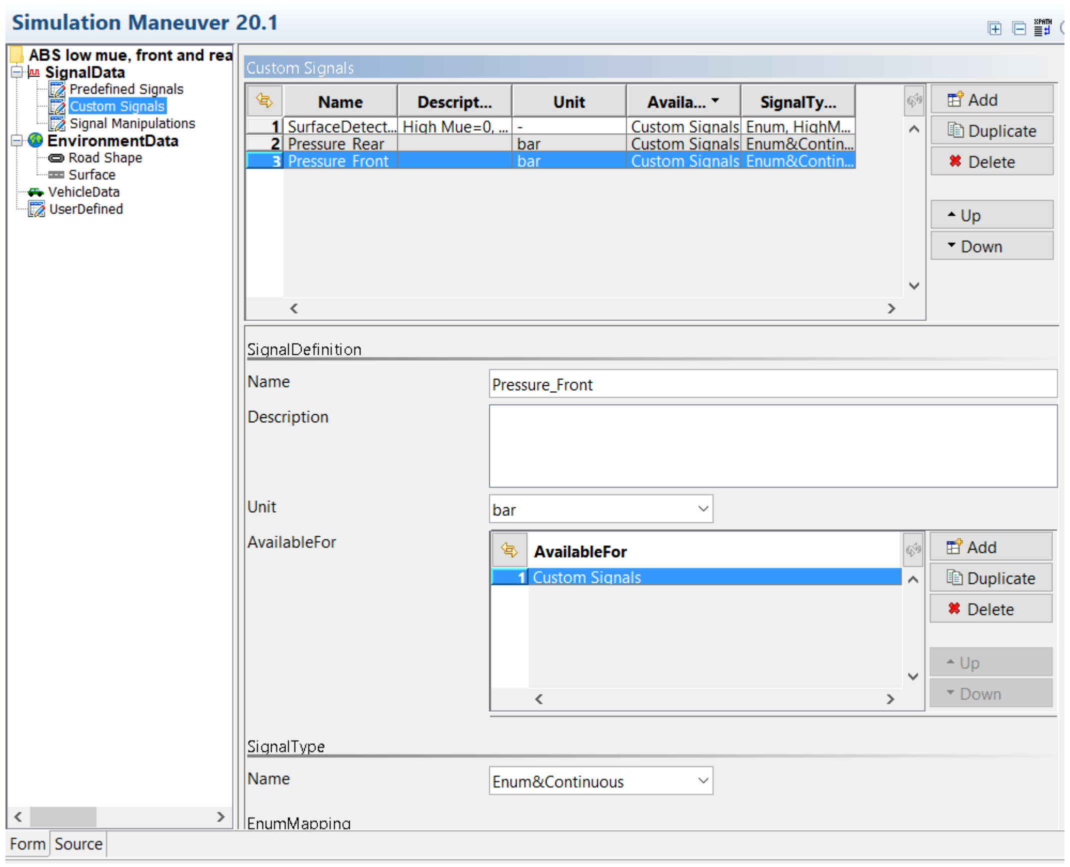
- ABS high mue front rear spike no slope
  - SignalData
    - Predefined Signals
    - Custom Signals
    - Signal Manipulations
  - EnvironmentData
    - Road Shape
    - Surface
  - VehicleData
  - UserDefined

| | Signal Name | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | Time [s] | 0 | 0.1 | 0.17 | 1 | 5.5 | 5.6 |
| 2 | SurfaceDetection [-] | HighMue | HighMue | HighMue | HighMue | HighMue | HighMue |
| 3 | Pressure_Front [bar] | 0 | 60 | 100 | 100 | 100 | 0 |
| 4 | Pressure_Rear [bar] | 0 | 60 | 100 | 100 | 100 | 0 |

New Values | Remove Values

Definitions

**Figure 4.6:** Signal Definition in Maneuver file



Simulation Maneuver 20.1

- ABS low mue, front and rea
  - SignalData
    - Predefined Signals
    - Custom Signals
    - Signal Manipulations
  - EnvironmentData
    - Road Shape
    - Surface
  - VehicleData
  - UserDefined

Custom Signals

| | Name | Descript... | Unit | Availa... | SignalTy... |
|---|---|---|---|---|---|
| 1 | SurfaceDetect... | High Mue=0, ... | - | Custom Signals | Enum, HighM... |
| 2 | Pressure_Rear | | bar | Custom Signals | Enum&Contin... |
| 3 | Pressure_Front | | bar | Custom Signals | Enum&Contin... |

Add | Duplicate | Delete | Up | Down

SignalDefinition

| | |
|---|---|
| Name | Pressure_Front |
| Description | |
| Unit | bar |
| AvailableFor | 1 Custom Signals |

SignalType

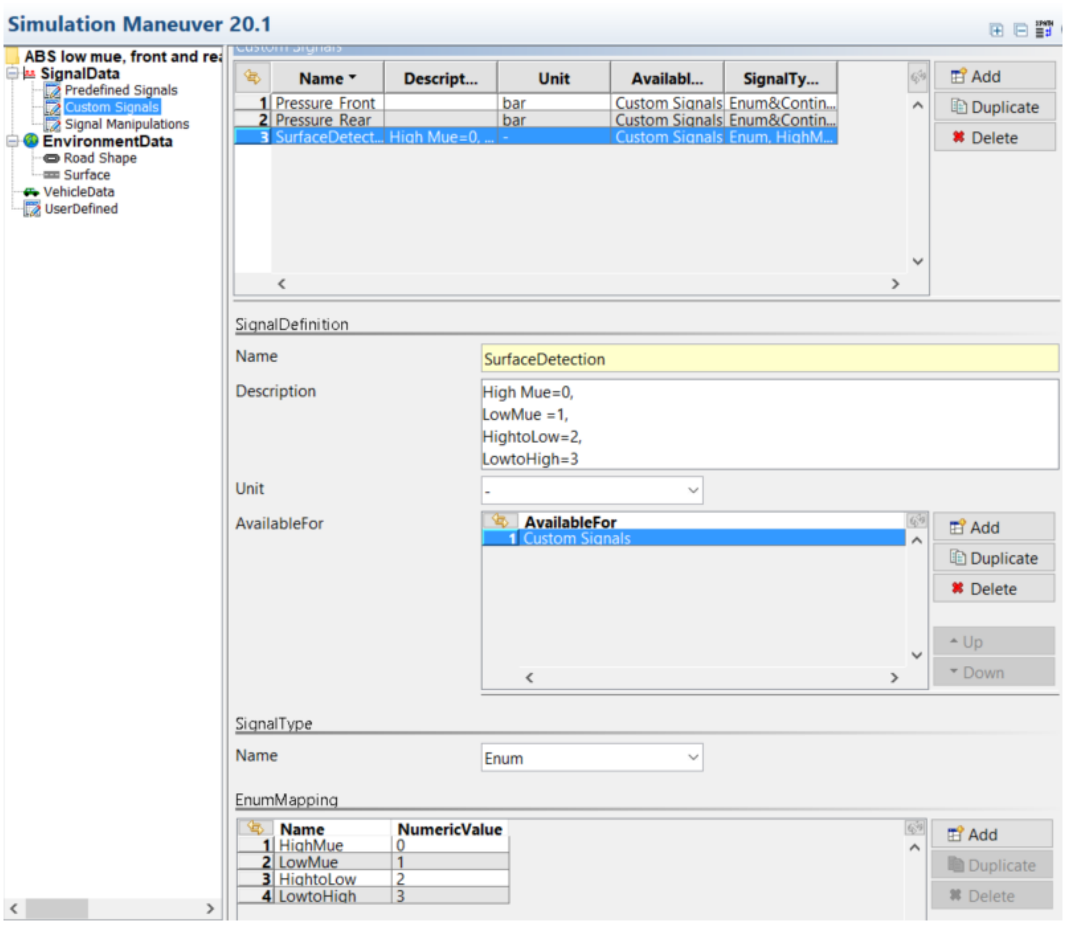
| | |
|---|---|
| Name | Enum&Continuous |

EnumMapping

Form | Source

**Figure 4.7:** Custom Signal Definition with an example of *Pressure_Front* Signal

4.8. It is defined as an enumeration (ENUM) signal with the following ENUM mapping: 0 (high mue), 1 (low mue), 2 (high to low transition), and 3 (low to high transition). The usage within the simulation environment is explained in Section 4.8.



| | Name | Descript... | Unit | Availabl... | SignalTy... |
|---|---|---|---|---|---|
| 1 | Pressure_Front | | bar | Custom Signals | Enum&Contin... |
| 2 | Pressure_Rear | | bar | Custom Signals | Enum&Contin... |
| 3 | SurfaceDetect... | High Mue=0, ... | - | Custom Signals | Enum, HighM... |

SignalDefinition

| | |
|---|---|
| Name | SurfaceDetection |
| Description | High Mue=0,<br>LowMue =1,<br>HightoLow=2,<br>LowtoHigh=3 |
| Unit | - |
| AvailableFor | 1 Custom Signals |

SignalType

| | |
|---|---|
| Name | Enum |

EnumMapping

| | Name | NumericValue |
|---|---|---|
| 1 | HighMue | 0 |
| 2 | LowMue | 1 |
| 3 | HightoLow | 2 |
| 4 | LowtoHigh | 3 |

**Figure 4.8:** Custom Signal definition : *SurfaceDetection*

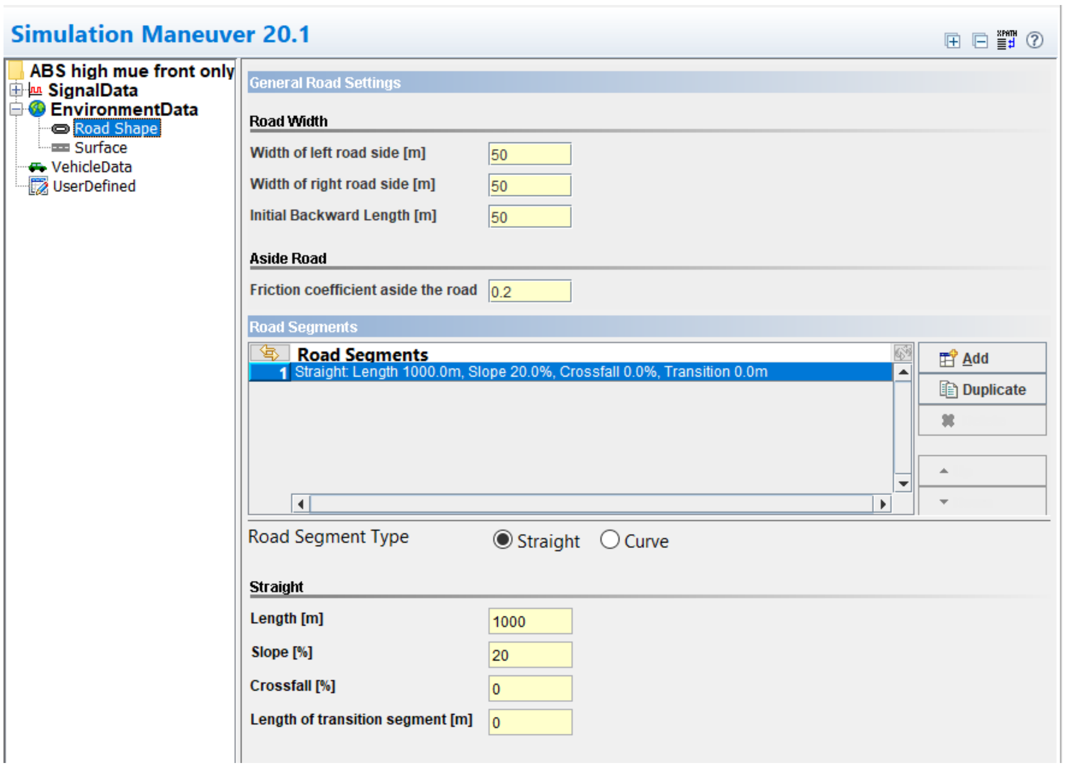
**3. Slope Percentage:** The slope is defined in the *EnvironmentData->Road Shape* field of the simulation maneuver and is shown in Figure 4.9. This is a constant value and specifies the down-slope in percentage. This is required as we have defined maneuvers to test the ABS braking during down-slope scenarios.

**4. Initial Speed and Simulation Duration:** The initial speed and simulation duration are specified in the simulation maneuver, as shown in Figure 4.10. The initial speed is defined in km/h and is used to set the speed of the bike during the start of the simulation. The simulation duration is specified in seconds and describes the total simulation duration for a particular maneuver.

### Usage of inputs in Simulation Environment

The inputs defined in the maneuvers are first loaded into a Matlab structure type (*CSSimManeuver-Data*) and then used inside the simulation environment. The mapping of the input signals in the maneuver to the Matlab structure is given in the table 4.3.

The input signals and parameters defined in the simulation maneuvers are used in the simulation environment as follows:



Road Width

Width of left road side [m]: 50

Width of right road side [m]: 50

Initial Backward Length [m]: 50

Aside Road

Friction coefficient aside the road: 0.2

Road Segments

1 Straight: Length 1000.0m, Slope 20.0%, Crossfall 0.0%, Transition 0.0m

Road Segment Type: (●) Straight ( ) Curve

Straight

Length [m]: 1000

Slope [%]: 20

Crossfall [%]: 0

Length of transition segment [m]: 0

**Figure 4.9:** Signal Definition: Slope Percentage

| **Signal Name** | **Mapping Name in matlab Struct** |
|---|---|
| Simulation Duration | CSSimManeuverData.Settings.SimulationTime |
| Initial Speed | CSSimManeuverData.Settings.InitialSpeed |
| Pressure applied on Front Wheel | CSSimManeuverData.Maneuver.Driver.Pressure_Front |
| Pressure applied on Rear Wheel | CSSimManeuverData.Maneuver.Driver.Pressure_Rear |
| Surface detection | CSSimManeuverData.Maneuver.Driver.SurfaceDetection |
| Down Slope Percentage | CSSimManeuverData.Maneuver.Environment.init.slopeX |

**Table 4.3:** Mapping of inputs from maneuver template to Matlab struct

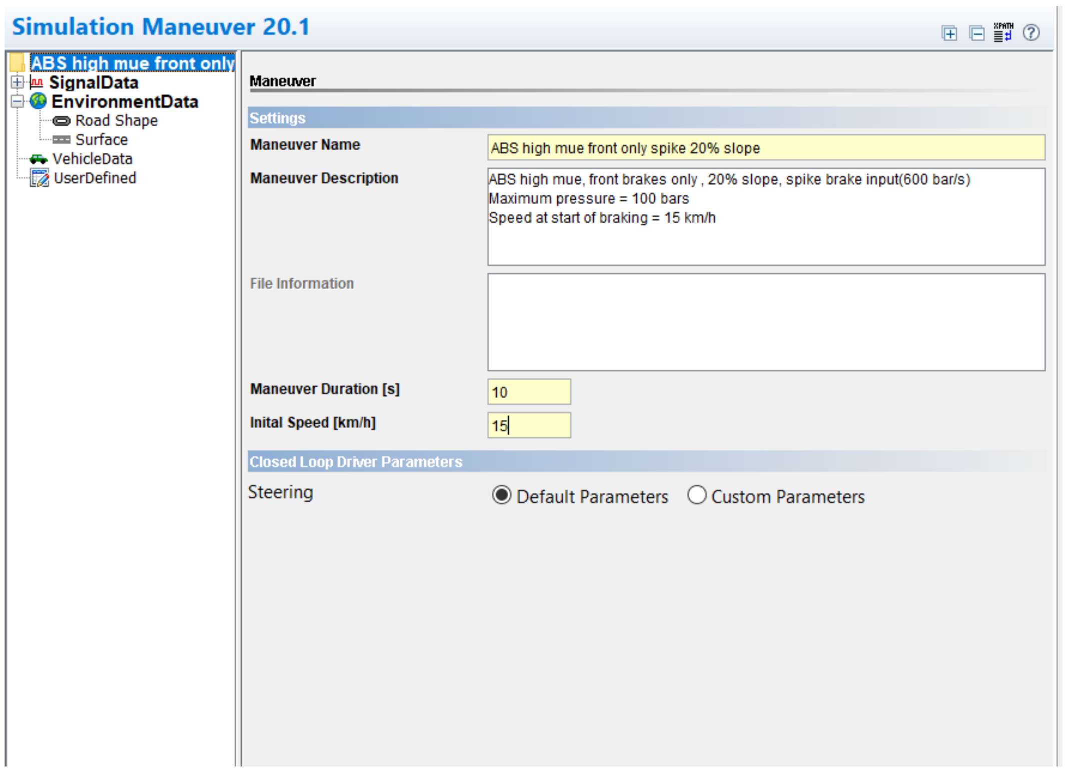
**1. Pressure Input on front and rear wheels:** CSSim framework offers a masked subsystem called *RBCS_aux_Sequence_exe* to use input signals defined in the maneuvers within the simulation environment. This is a time- or way-dependent signal definition. If the struct variable exists in the workspace, then this value will be used; otherwise, the default values specified in mask are used. The struct variable in the workspace must have the following structure:

- struct_name.xvalue

- struct_name.yvalue

- struct_name.mode: time or way

- struct_name.interpolation: on or off


Simulation Maneuver 20.1

ABS high mue front only
SignalData
EnvironmentData
Road Shape
Surface
VehicleData
UserDefined

Maneuver

Settings

Maneuver Name: ABS high mue front only spike 20% slope

Maneuver Description: ABS high mue, front brakes only , 20% slope, spike brake input(600 bar/s)
Maximum pressure = 100 bars
Speed at start of braking = 15 km/h

File Information:

Maneuver Duration [s]: 10

Inital Speed [km/h]: 15

Closed Loop Driver Parameters

Steering: [✓ Default Parameters] Custom Parameters

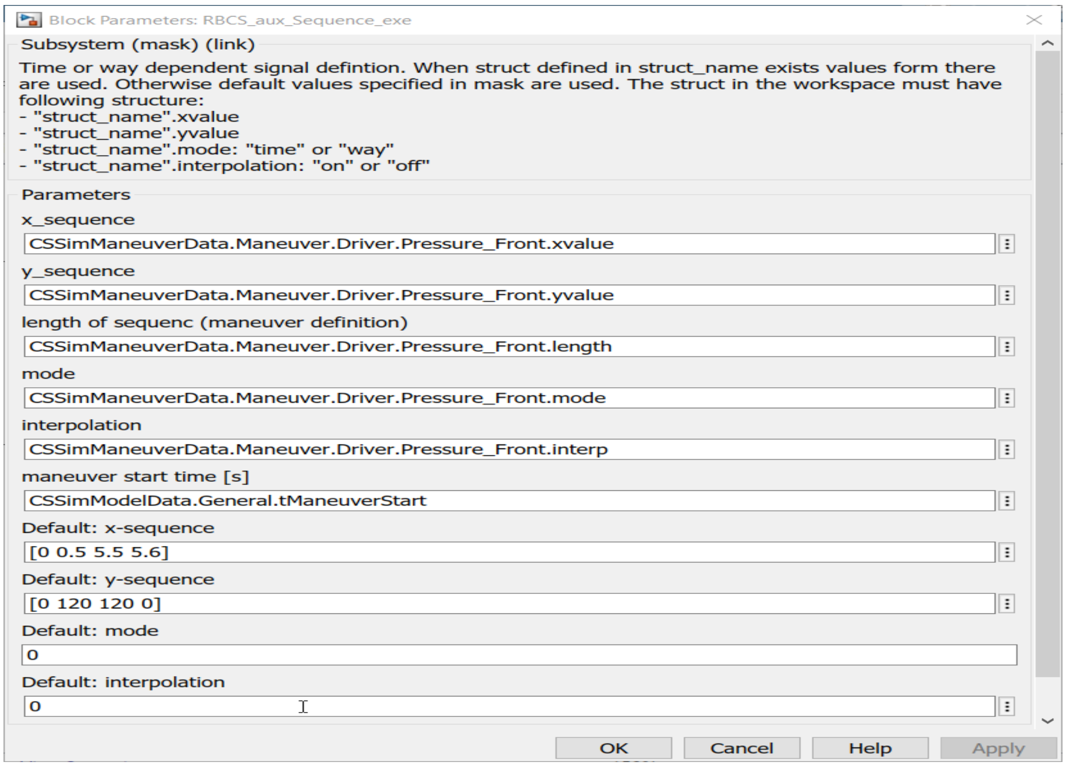
**Figure 4.10:** Signal Definition: Initial Speed and Simulation Duration

Figure 4.11 shows the definition of pressure input on the front wheel inside the simulation environment. Pressure input on the rear wheel is specified in a similar way.

**2. Surface detection:** This signal is used inside the bike model to calculate the longitudinal tire force on the front and rear wheels. Depending on the type of surface specified in the maneuver, the tire characteristics are changed for high friction and gravel surfaces. Additionally, this signal is also used to change the tire characteristics during high-to-low transitions and low-to-high transitions. For example, during a high-to-low transition, the high mue tire characteristics are used for the initial 1 second of the braking duration, and then the low mue tire characteristics are used for the remaining execution time. Figure 4.12 shows an example of the transition surface. The algorithm is created using a simple switch-case implementation, as shown in 4.1.

**3. Initial Speed and Simulation Maneuver duration:** This signal is used to set the initial speed in the bike model. The initial speed converted to m/s and used to set the overall bike speed in the bike model. Additionally, the individual wheel rotation speeds are also specified in rad/s using the initial speed defined in the simulation maneuver.

The simulation maneuver duration is needed mainly to vary the input signals. This is also used to set the stop time in Matlab. The total time to execute a maneuver is 3 seconds longer than the specified maneuver duration. This time of 3 seconds is required to stabilize the simulation environment during the start of execution.


Block Parameters: RBCS_aux_Sequence_exe

Subsystem (mask) (link)

Time or way dependent signal defintion. When struct defined in struct_name exists values form there are used. Otherwise default values specified in mask are used. The struct in the workspace must have following structure:
- "struct_name".xvalue
- "struct_name".yvalue
- "struct_name".mode: "time" or "way"
- "struct_name".interpolation: "on" or "off"

Parameters

| | |
|---|---|
| x_sequence | CSSimManeuverData.Maneuver.Driver.Pressure_Front.xvalue |
| y_sequence | CSSimManeuverData.Maneuver.Driver.Pressure_Front.yvalue |
| length of sequenc (maneuver definition) | CSSimManeuverData.Maneuver.Driver.Pressure_Front.length |
| mode | CSSimManeuverData.Maneuver.Driver.Pressure_Front.mode |
| interpolation | CSSimManeuverData.Maneuver.Driver.Pressure_Front.interp |
| maneuver start time [s] | CSSimModelData.General.tManeuverStart |
| Default: x-sequence | [0 0.5 5.5 5.6] |
| Default: y-sequence | [0 120 120 0] |
| Default: mode | 0 |
| Default: interpolation | 0 |

**Figure 4.11:** Usage in Simulation Environment

**Listing 4.1** Switch case implementation to adapt the tire characteristics

```
Switch SURFACE:
    Case HighMue:
        Use highmue tire characteristics
    Case Gravel:
        Use low mue tire characteristics
    Case High-to-low
        If time > 1  % transition occurs 1 second after start of braking
            Use low mue tire characteristics
        Else
            Use High Mue tire characteristics
    Case Low-to-High
        If time > 1  % transition occurs 1 second after start of braking
            Use highmue tire characteristics
        Else
            Use low Mue tire characteristics
    Default
        Use High Mue tire Characteristics
```

**4. Down Slope percentage:** This signal is used to specify the gravity vector in the bike model. In the default scenario without the slope, the gravity vector is given by 4.1 where $g$ is the acceleration due to gravity in negative Z direction and has the value -9.81 $m/s^2$.

$$[X \quad Y \quad Z] = [0 \quad 0 \quad g] \tag{4.1}$$

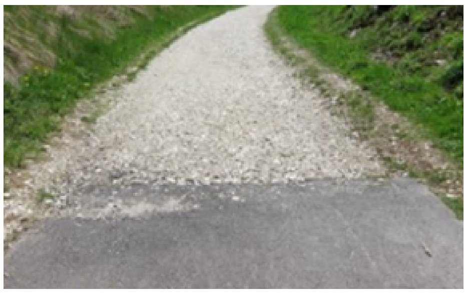
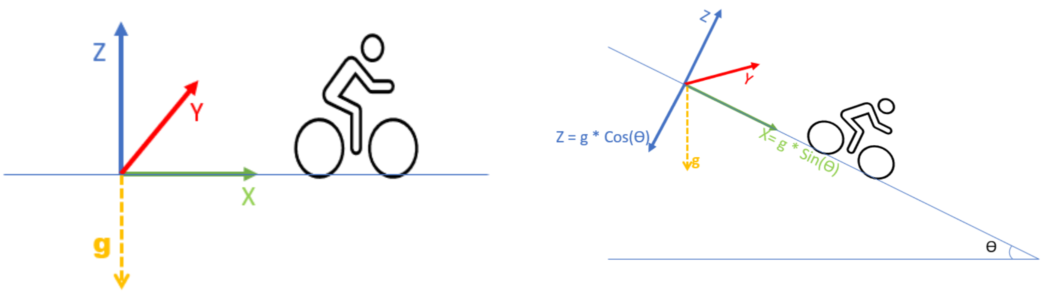


**Figure 4.12:** Mue jump surface

In case of a down slope of angle $\theta$, the gravity vector is given by 4.2. The Figure 4.13a represents the coordinate system at the Center of Gravity (COG) of the eBike. The X-axis points forward in the direction of travel parallel to the road. The Y-axis points to the left in the direction of travel, the Z-axis points upwards. In case of a road surface without slope, the gravity points downwards and the gravity vector is given as in equation 4.1, but, during a slope of angle $\theta$, the gravity vector is defined as in equation 4.2.

$$[X \quad Y \quad Z] = [g * sin(\theta) \quad 0 \quad g * cos(\theta)] \tag{4.2}$$

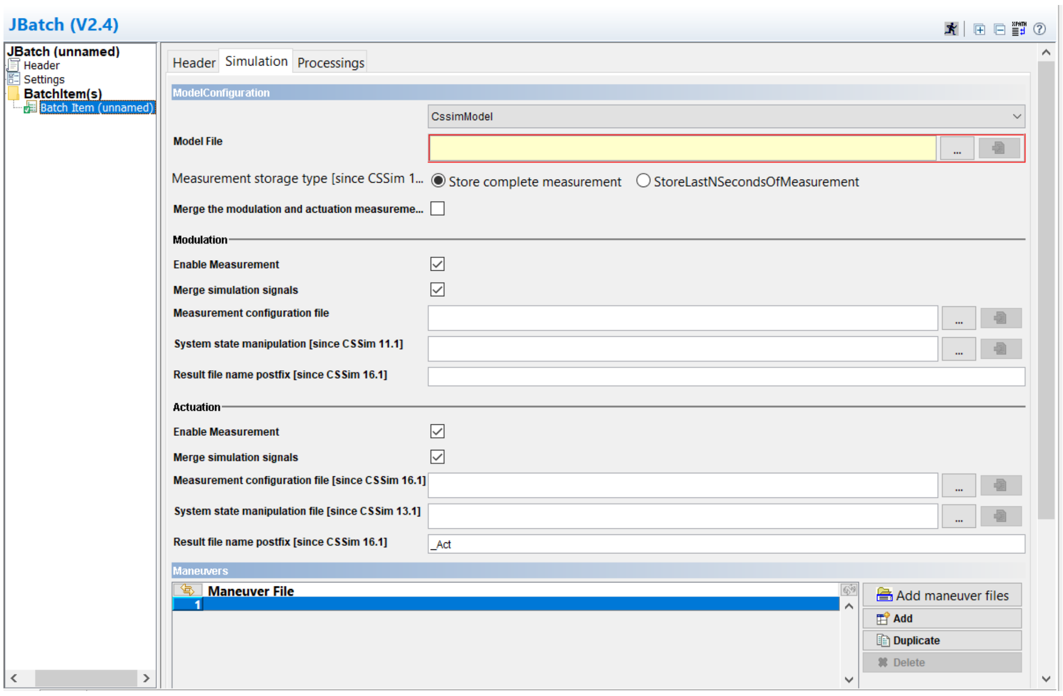


**(a)** No Slope **(b)** Gravity vector during a slope

**Figure 4.13:** Gravity vector during no slope and with slope

### 4.2.2 Automated Test Execution

The CSSim framework provides an XML template called *JBatch* to enable automated maneuver execution. With *JBatch*, it is possible to execute a set of batch items. Each of the batch items will refer to a simulation model and a set of simulation maneuvers. At execution time, for each of the maneuvers one simulation run will be started. The results of the simulation are collected to a result folder. In that results folder, a report and a log file are generated. The *Jbatch* template is as shown in Figure 4.14.



ModelConfiguration

CssimModel

Model File

Measurement storage type [since CSSim 1... [✓ Store complete measurement] StoreLastNSecondsOfMeasurement

Merge the modulation and actuation measureme... ☐

Modulation

| | |
|---|---|
| Enable Measurement | ✓ |
| Merge simulation signals | ✓ |
| Measurement configuration file | |
| System state manipulation [since CSSim 11.1] | |
| Result file name postfix [since CSSim 16.1] | |

Actuation

| | |
|---|---|
| Enable Measurement | ✓ |
| Merge simulation signals | ✓ |
| Measurement configuration file [since CSSim 16.1] | |
| System state manipulation file [since CSSim 13.1] | |
| Result file name postfix [since CSSim 16.1] | _Act |

Maneuvers

| | Maneuver File |
|---|---|
| 1 | |

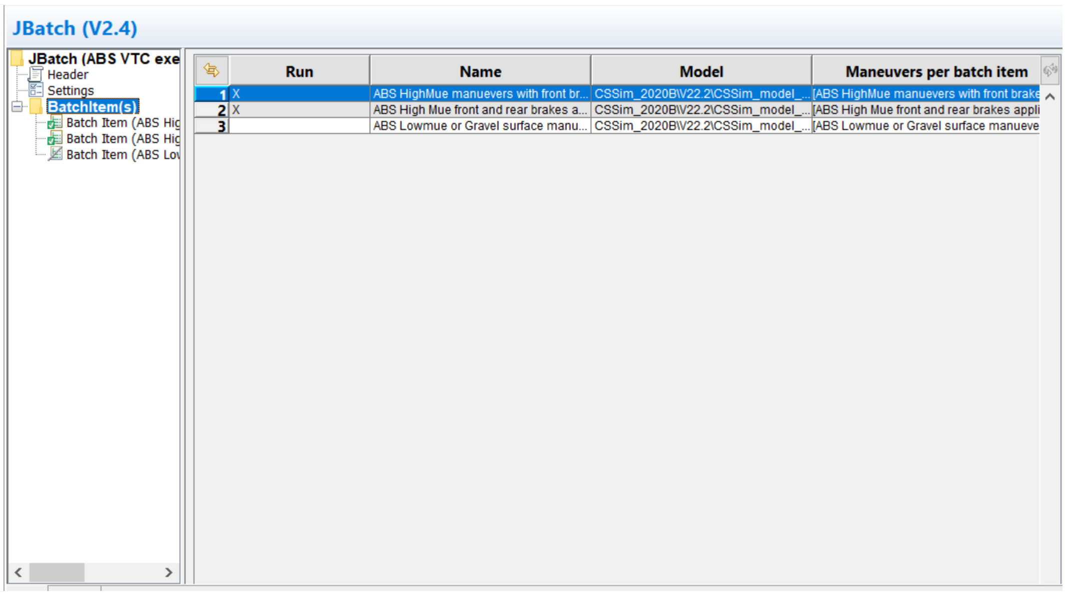
**Figure 4.14:** JBatch.xml template

In the *Header* section, a name and a description of the document have to be defined. In the *Settings* section, it is to enable or disable animation, whereas in our case, we disable the animation to save the simulation run time. The batch items section contains one or more batch items. Each batch item consists of three tabs: *Header*, *Simulation*, and *Processings*. In the *Header* tab, we can enable or disable the respective batch item and define the name and description of the batch item. In the *Simulation* tab, the path to the simulation model and the simulation maneuvers have to be provided. The *Processings* tab has some post-processing tasks that are not applicable for our use case.

The Figure 4.15 shows the *Jbatch* XML file created and used in our work. Here we have created three batch items: ABS HighMue maneuvers with the front brake applied, ABS HighMue maneuvers with both the front and rear brakes applied, and ABS LowMue or Gravel maneuvers. For the purpose of demonstration, only the first two batch items are enabled. Finally, the results generated are sorted and stored according to the batch items, and for every batch item, the results are named according to the respective maneuver file.

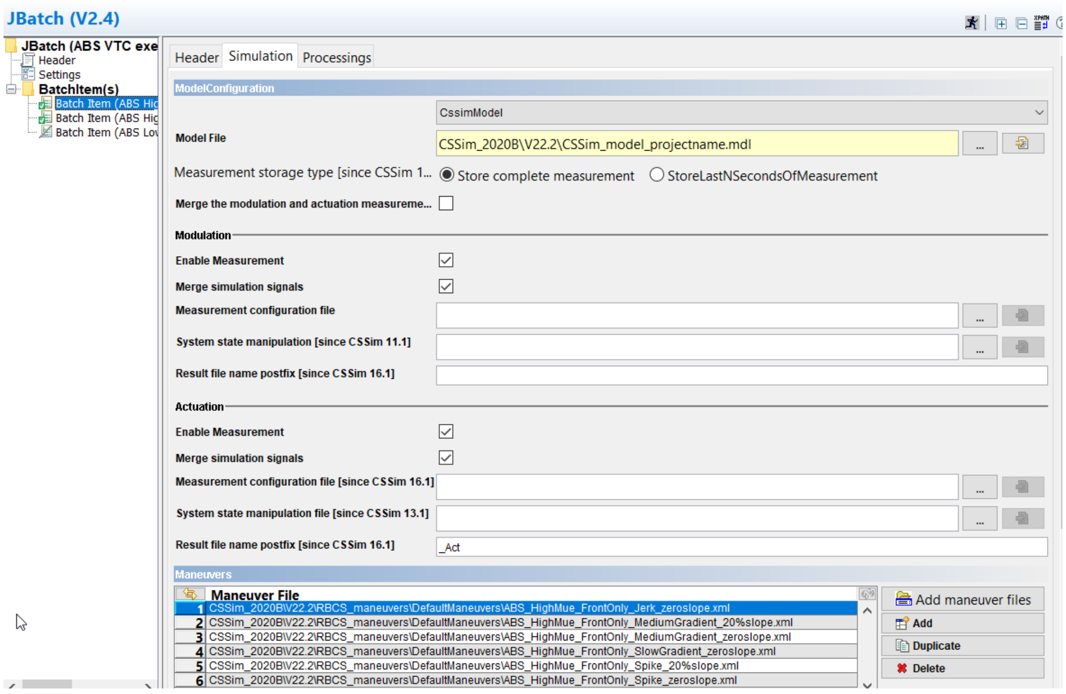
The Figure 4.16 shows one of the batch items with the linked maneuvers and simulation model file. If the name of the output file for each maneuver is not specified, the name is automatically created based on the file name of the referenced maneuver file.

Once the *Jbatch* is set up, click on the run button on the toolbar to start the execution. A run-time parameter dialog will appear where it is possible to specify the path to store the results and in the optional parameters field, we can specify the run time parameters if needed. We need to specify the parameter *simulation.PreferredBitness=64* as shown in Figure 4.17 because, by default Jbatch searches for the 32 bit Matlab installation and this needs to be specified in case 64 bit Matlab is used in the simulation model.



| | Run | Name | Model | Maneuvers per batch item |
|---|---|---|---|---|
| 1 | X | ABS HighMue manuevers with front br... | CSSim_2020B\V22.2\CSSim_model_... | [ABS HighMue manuevers with front brake |
| 2 | X | ABS High Mue front and rear brakes a... | CSSim_2020B\V22.2\CSSim_model_... | [ABS High Mue front and rear brakes appli |
| 3 | | ABS Lowmue or Gravel surface manu... | CSSim_2020B\V22.2\CSSim_model_... | [ABS Lowmue or Gravel surface manueve |

**Figure 4.15:** Batch Items in JBatch.xml



ModelConfiguration

| | |
|---|---|
| | CssimModel |
| Model File | CSSim_2020B\V22.2\CSSim_model_projectname.mdl |
| Measurement storage type [since CSSim 1... | [✓ Store complete measurement] StoreLastNSecondsOfMeasurement |
| Merge the modulation and actuation measureme... | ☐ |

Modulation

| | |
|---|---|
| Enable Measurement | ✓ |
| Merge simulation signals | ✓ |
| Measurement configuration file | |
| System state manipulation [since CSSim 11.1] | |
| Result file name postfix [since CSSim 16.1] | |

Actuation

| | |
|---|---|
| Enable Measurement | ✓ |
| Merge simulation signals | ✓ |
| Measurement configuration file [since CSSim 16.1] | |
| System state manipulation file [since CSSim 13.1] | |
| Result file name postfix [since CSSim 16.1] | _Act |

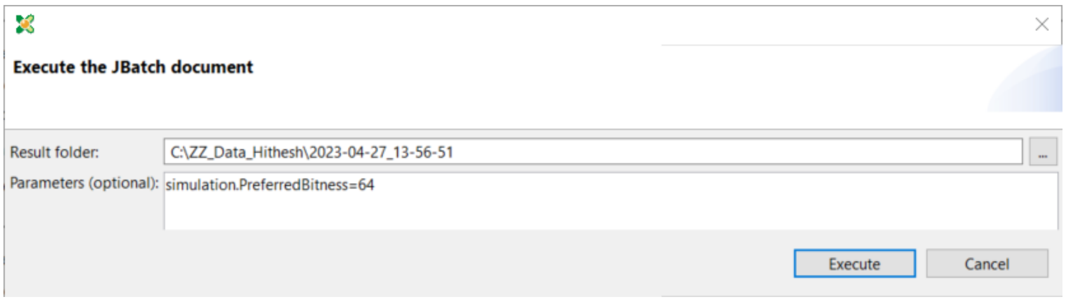
Maneuvers

| | Maneuver File |
|---|---|
| 1 | CSSim_2020B\V22.2\RBCS_maneuvers\DefaultManeuvers\ABS_HighMue_FrontOnly_Jerk_zeroslope.xml |
| 2 | CSSim_2020B\V22.2\RBCS_maneuvers\DefaultManeuvers\ABS_HighMue_FrontOnly_MediumGradient_20%slope.xml |
| 3 | CSSim_2020B\V22.2\RBCS_maneuvers\DefaultManeuvers\ABS_HighMue_FrontOnly_MediumGradient_zeroslope.xml |
| 4 | CSSim_2020B\V22.2\RBCS_maneuvers\DefaultManeuvers\ABS_HighMue_FrontOnly_SlowGradient_zeroslope.xml |
| 5 | CSSim_2020B\V22.2\RBCS_maneuvers\DefaultManeuvers\ABS_HighMue_FrontOnly_Spike_20%slope.xml |
| 6 | CSSim_2020B\V22.2\RBCS_maneuvers\DefaultManeuvers\ABS_HighMue_FrontOnly_Spike_zeroslope.xml |

**Figure 4.16:** One of the Btach Item

## 4.3 Design and Implementation of SiL with IMU sensor

Future generations of Bosch ebike ABS will utilize data from the IMU and WSS to measure the pitch angle, roll angle, and sideslip angle in order to increase braking efficiency in curves and prevent pitch-overs on high-friction surfaces. When a rider is descending a steep hill, for instance,


Execute the JBatch document

Result folder: C:\ZZ_Data_Hithesh\2023-04-27_13-56-51

Parameters (optional): simulation.PreferredBitness=64

**Figure 4.17:** JBatch Execution window

the ABS detects the pitch angle and modifies the brake pressure to prevent the eBike from pitching forward. In addition, if the rider applies the brakes while turning on a surface with high friction, the ABS system will adapt to prevent skidding and loss of control. The ABS control software, which comprises the state estimation algorithm with the sensor fusion of IMU and WSS, is currently available. Therefore, we extend the available eBike bike model by incorporating an IMU sensor model so that the relevant IMU inputs are provided to the control software.

However, the bike model available for the eBike is a 2D model, which means only straight-ahead riding can be simulated using the model. This bike model cannot simulate situations such as riding and braking on a curve, and this is a limitation in order to implement the IMU error model in SiL. Due to this limitation, we restrict ourselves in this thesis to evaluating ABS performance only in the case of IMU errors related to pitch angle.

The architecture of the proposed SiL model is shown in Figure 4.18. The SiL simulation model consists of the models for the eBike HU, pressure sensor model, WSS model, and bike model. Furthermore, the bike model is extended with the addition of a sensor model for the IMU sensor. The replay data consists of the measurements from the actual bike test. The ABS control algorithm consists of the ABS software exported as an ASCET PSL module.

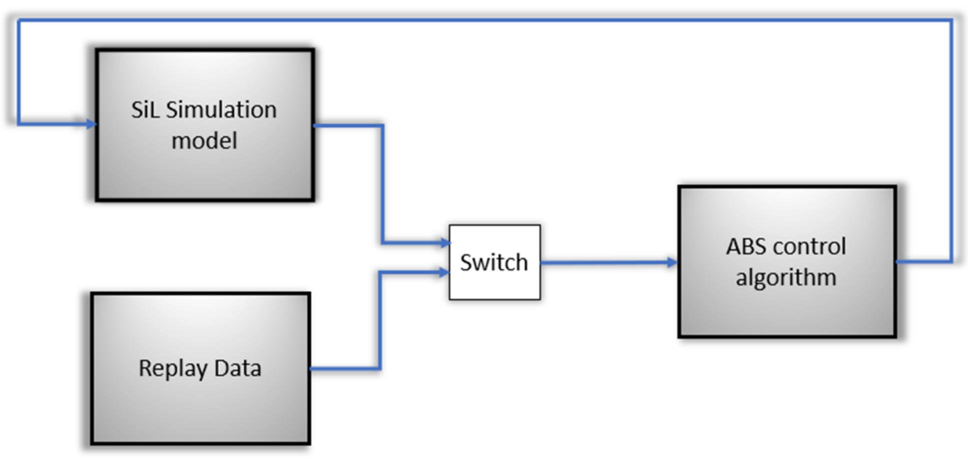


**Figure 4.18:** Architecture of the proposed SiL environment for IMU errors

The basic idea is that we use the measurements from the real bike testing to extract and provide the necessary inputs to the controller, and this is termed *replay data.* We observe if the reference pitch angle and roll angle from the replay data match the estimated roll and pitch angles calculated from the ABS control algorithm. We then identify the regions in the replay data that have a fairly constant vehicle speed and both the pitch and the roll angle are approximately zero, and then we switch the inputs to the controller from the replay data to the simulation environment. After this, the braking maneuver on a high friction or a gravel surface can be performed. Once the switching of the inputs is successfully achieved, we then induce errors into the IMU sensor and verify the braking of the ABS. The replay part is important so that the state variables (pitch, roll, and yaw) calculated from the ABS controller match the reference data. We identified the stable region to be at time 238, where the vehicle speed is 8 m/s, and we switched the input signals at this time.

The Figure 4.19 shows the results from the replay part of the simulation for vehicle speed, pitch angle, and roll angle. The reference signals from the input replay data and the estimated signals from the ABS controller match approximately. The values at the identified stable region at time 238 are plotted in the Figure 4.20, and the values are fairly constant.

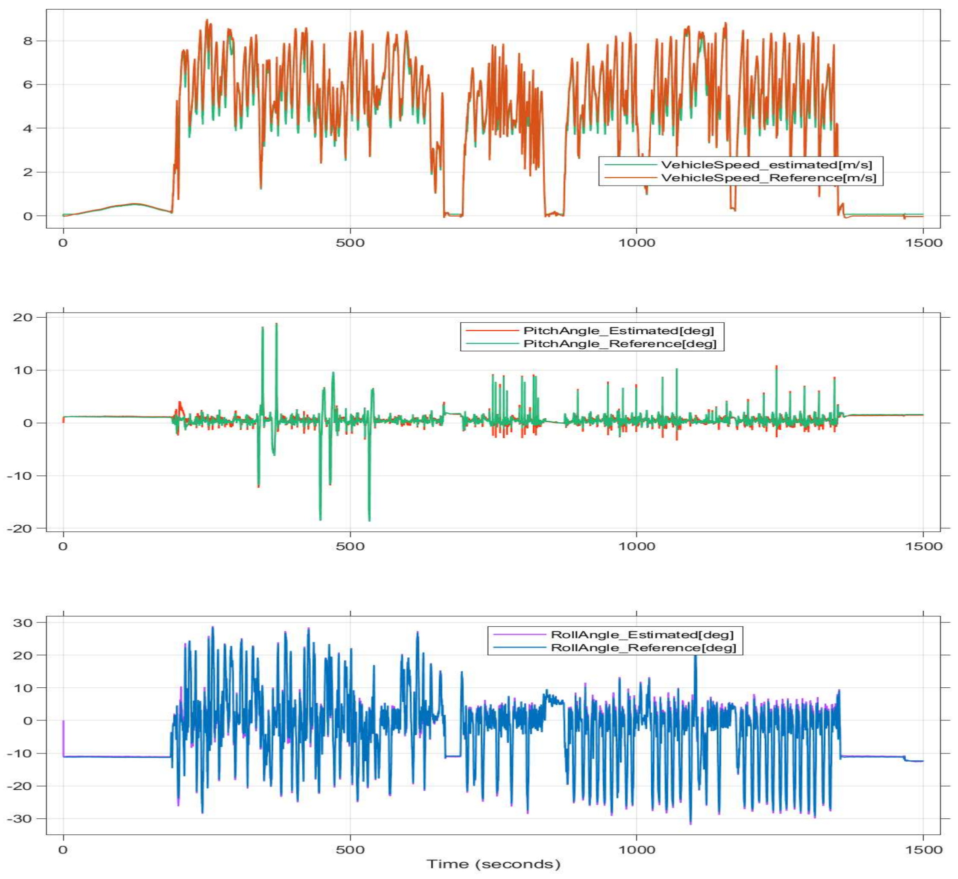


**Figure 4.19:** Graph of the replay data

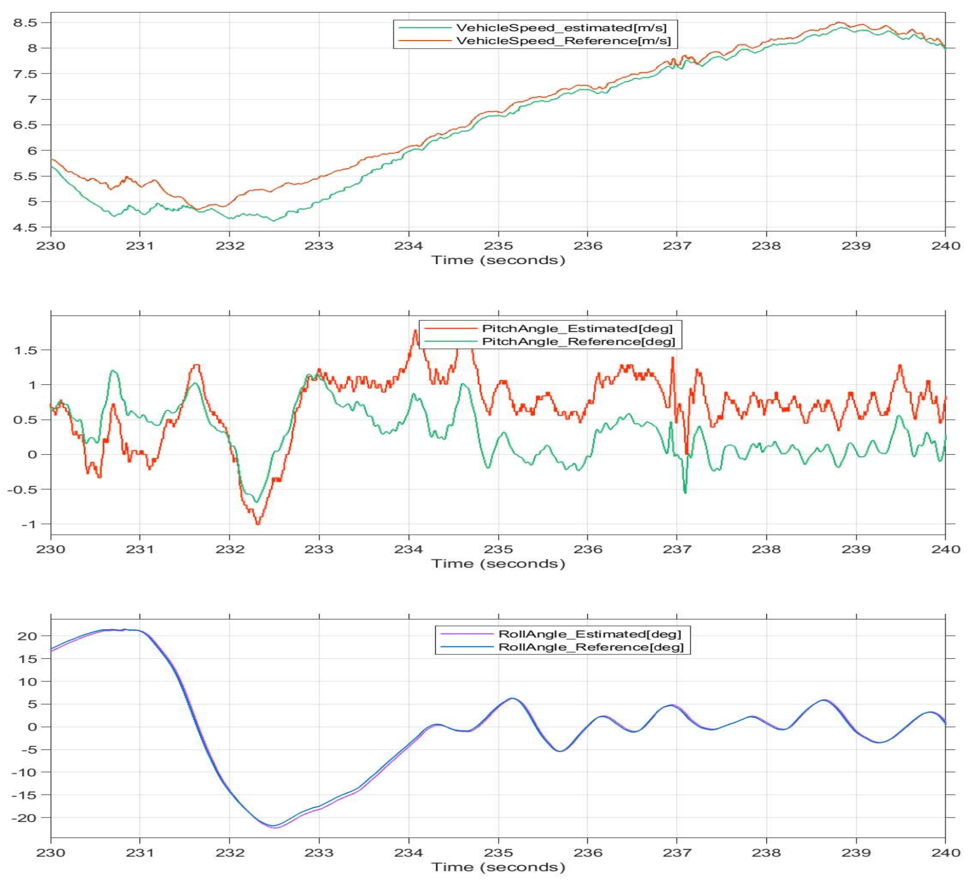


**Figure 4.20:** Graph at the identified stable region in the replay data

IMU sensor has to be modeled in the bike model to provide the IMU-related input signals to the ABS controller. The Figure 4.21 shows the implementation of the IMU sensor in the bike model. We use the transform sensor module of simulink to get the accelerations and the angular velocities. The IMU is assumed to be placed at the center of gravity of the eBike for the purpose of simulation. Therefore, the measured accelerations and angular velocities are with reference to the COG of the eBike. These values are then fed into the IMU sensor block of the simulink. The block outputs sensor measurements based on device motion. The inputs are in the local navigation frame. The outputs are in the local sensor frame. The parameters on the accelerometer, gyroscope, and magnetometer tabs can be set to match values on a sensor datasheet [Mat]. The sensor signals from the accelerometer and gyroscope are provided to the ABS control software and the values from the magnetometer are terminated. The output signals from the IMU are then rotated to match the sensor orientation defined in the ABS controller. The sensor orientation is defined to be as X-front, Y-left and Z-up.

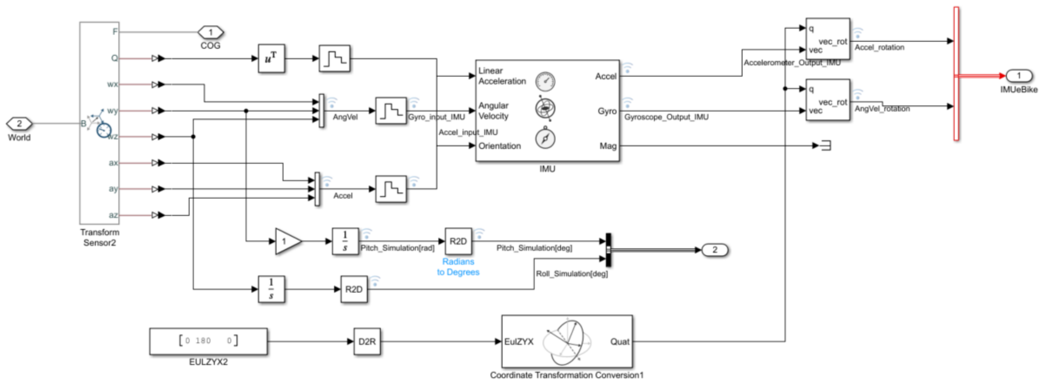


**Figure 4.21:** IMU sensor model implementation inside bike model

# 5 Evaluation of SiL simulation environments

This chapter illustrates the evaluation approach used to validate the SiL simulation models developed. We compare four different maneuvers in both SiL and bike tests to determine how well the SiL model with the CSSim framework performs. Furthermore, we identified the KPIs and conducted a sensitivity analysis to validate the influence on these KPIs by modifying various bike and environmental parameters. In the SiL model for IMU errors, we introduce offset errors associated with gyroscopes and then evaluate how these errors affect the ABS algorithm.

## 5.1 Evaluation of SiL with CSSim framework

In order to evaluate the developed SiL model, we compare the results with the actual bike measurements. High mue surface, gravel surface, low to high transition, and high to low transition maneuvers are considered. Trekking bike configuration is used in both SiL and in test bikes. The initial speed and the applied pressure from the rider are the same in SiL and in bike measurement. Additionally, same software version is used in the SiL environment and in the test bike. Finally, we evaluated a minimum of three graphs for each of the maneuvers and selected one graph for each maneuver for documentation purposes.

The signals considered for evaluation are as shown in the Table 5.1 along with their description and units.

| **Signal Name** | **Unit** | **Description** |
|---|---|---|
| vFL | m/s | Front wheel speed |
| vRL | m/s | Rear wheel speed |
| p_Target_FL | bar | Target pressure requested from the ABS controller |
| pMC_Front | bar | Rider applied pressure on front wheel |
| pMC_Rear | bar | Rider applied pressure on rear wheel |
| p_WC_HUModel | bar | Pressure applied at the wheel caliper |
| AvFL_Duration | Time in seconds | Duration of the outlet valve opening |
| EvFL_Duration | Time in seconds | Duration of the inlet valve opening |
| EvFL_Duty | - | PWM duty for the inlet valve |
| EvFL_Duty2 | - | PWM duty2 for inlet valve |

**Table 5.1:** Signals required for evaluation of SiL with CSSim

### 5.1.1 Evaluation of high-mue braking

**Objective:** The goal is to compare the results generated from SiL with the bike test while braking on a surface with high friction.

**Test Setup:** Initial speed is 8.8 m/s, pressure profile in SiL is the same as in a bike, and only front brakes are applied.

**Analysis:** The results from SiL and bike measurements are shown in Figures 5.1 and 5.2, respectively. During braking on a high-friction surface, the ABS should prevent lift-up by reducing the pressure on the front wheel, and similar behavior is observed both in SiL and in the bike test. The pressure control works as expected in the developed SiL model.

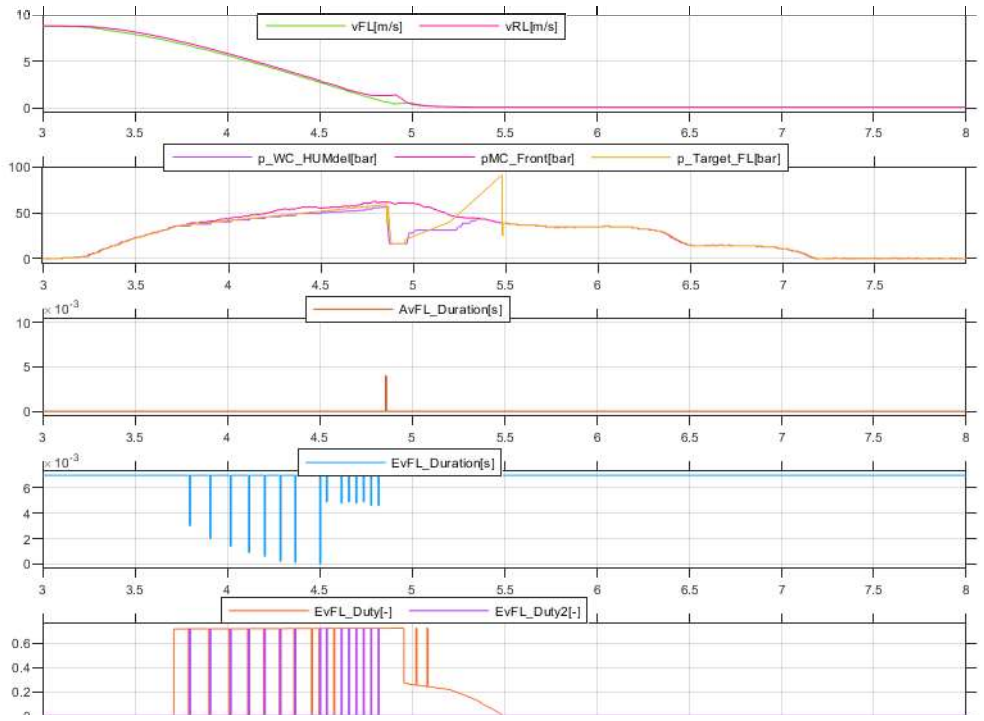


**Figure 5.1:** Braking on High mue surface: Result from SiL

### 5.1.2 Evaluation of Gravel surface braking

**Test objective:** To compare the results from SiL and bike measurements during braking on a gravel surface.

**Test Setup**: Initial speed is the same, the pressure profile in SiL is the same as in a bike, and the pressure is applied only on the front wheel.

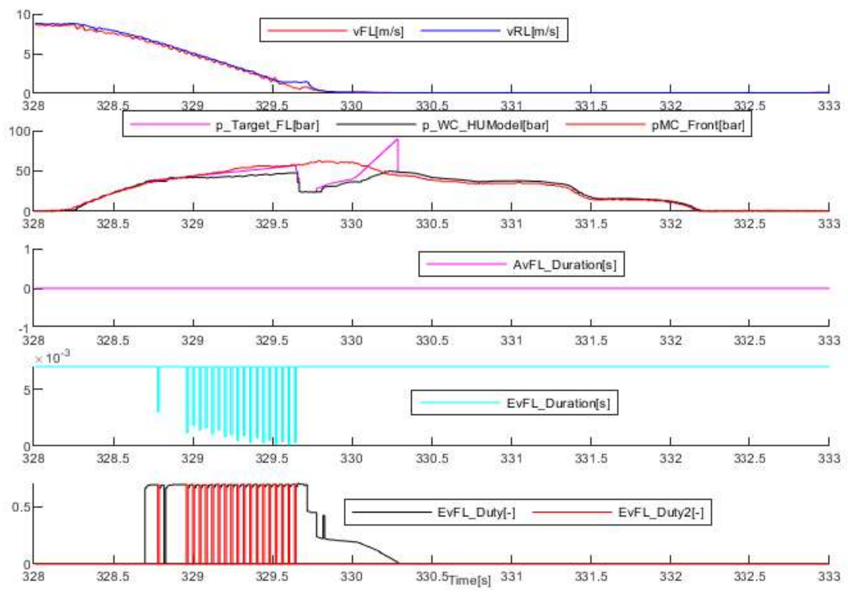


**Figure 5.2:** Braking on High mue surface: Result from bike measurement

**Analysis**: The results from SiL and bike measurement are shown in Figures 5.3 and 5.4, respectively. During braking on a gravel surface, the ABS should perform a slip control to prevent the front wheel from locking. The results from SiL and the bike test do not match exactly, but slip control occurs as expected in the SiL model. Additionally, the results from SiL show that the ABS performs better in comparison with the bike measurement.

### 5.1.3 Evaluation of low to high transition braking

**Test objective:** To compare the results from SiL and bike measurements during braking on a low mue to high mue transition surface.

**Test Setup:** Initial speed is 8.6 m/s, the pressure profile in SiL is the same as in bike, and the pressure is applied only on the front wheel.

**Analysis:** The results from SiL and bike measurements are shown in Figures 5.5 and 5.6, respectively. The ABS controller should perform slip control on the low-friction surface, and the wheels should quickly adapt to the high-friction surface after the transition. The results from SiL match the results from the bike tests with respect to the expected operation of the ABS controller. During the transition from low to high friction surfaces, the wheels should quickly adapt to the surface, and the ABS controller should quickly adapt the target pressure that needs to be applied to the wheel.

### 5.1.4 Evaluation of high to low transition braking

**Test objective:** To compare the results from SiL and bike measurement during braking on a high mue to low mue transition surface.

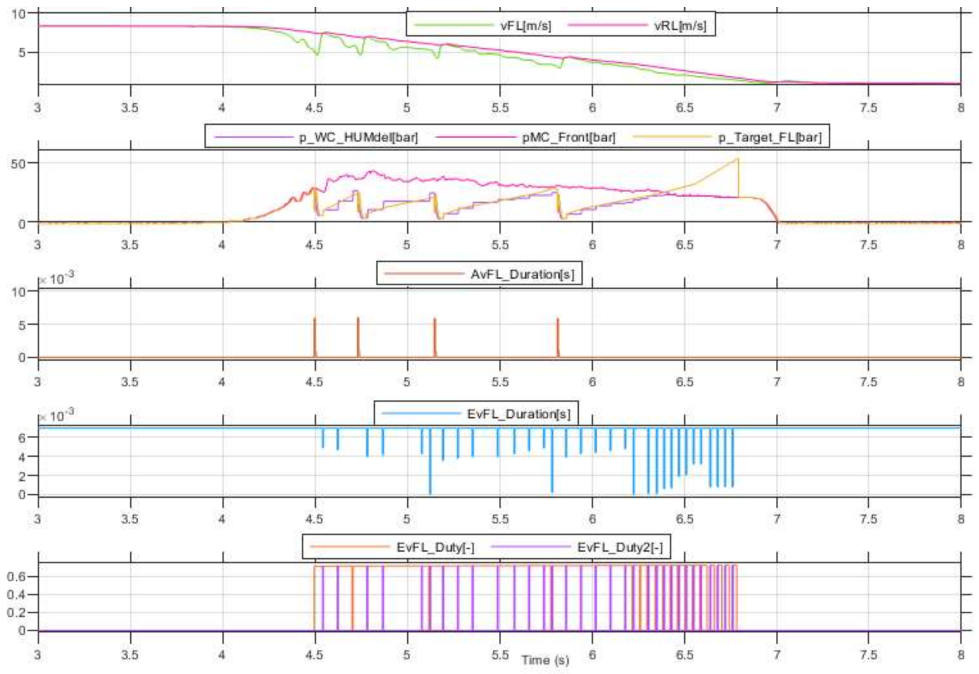


**Figure 5.3:** Braking on Low mue surface: Result from SiL

**Test Setup:** Initial speed is 8.5 m/s, the pressure profile in SiL is the same as in a bike, and the pressure is applied only on the front wheel.

**Analysis:** The results from SiL and bike measurement are shown in Figures 5.7 and 5.8, respectively. The transition occurs after 0.75 seconds since the maneuver started, and after the transition to a low mue surface, the ABS controller should perform a slip control to prevent wheel lockup. The results from SiL match the results from the bike tests with respect to the expected operation of the ABS controller. During transition, the wheels should quickly adapt to the surface, and the ABS controller should change the target pressure that needs to be applied to the wheel.

Overall, the results from the SiL look similar to the results generated from real bike testing. They do not match exactly because the tire characteristics defined in SiL may differ from those in a real-world scenario. Even a small rock in the road could impact the tire force calculation and might slightly alter the ABS behavior.

Additionally, the accumulator volume can also affect the amount of pressure decrease during each ABS control cycle. In SiL, the maneuvers always start with an empty accumulator, whereas in real bike tests, this might not always be the case.

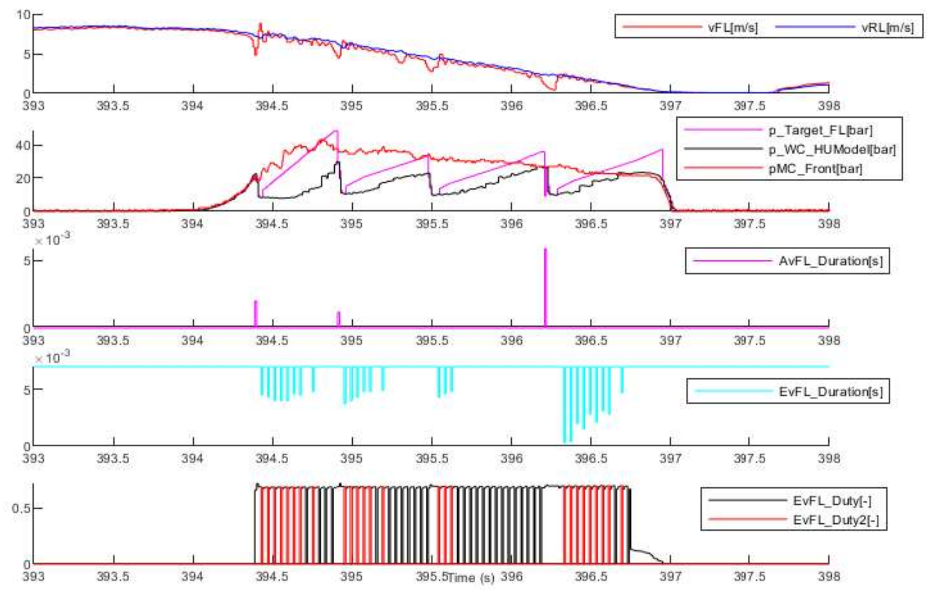


**Figure 5.4:** Braking on Low mue surface: Result from bike measurement

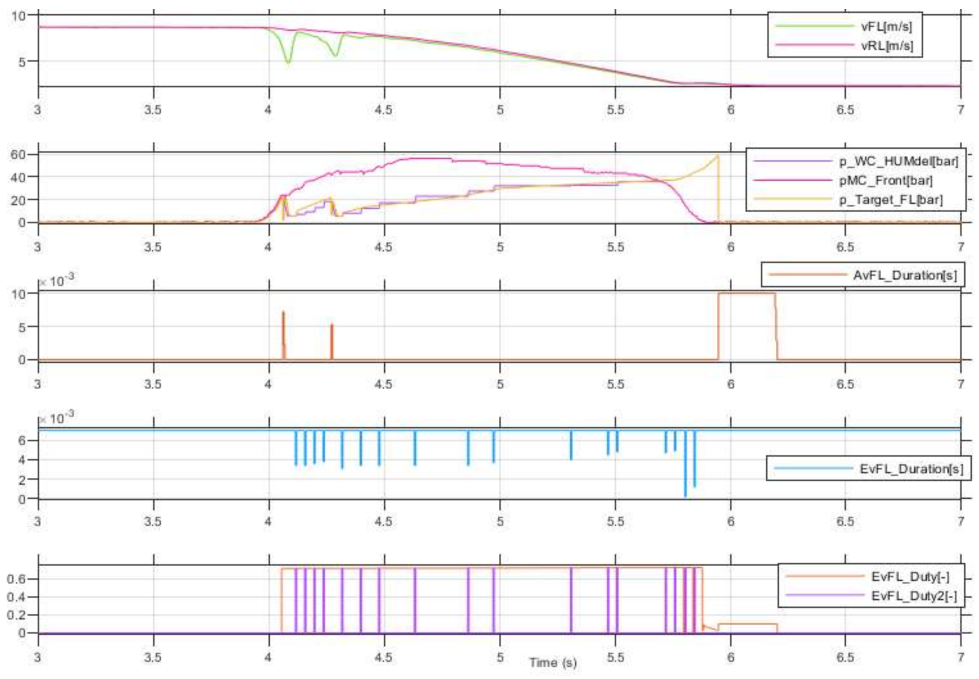


**Figure 5.5:** Braking on Low to high transition surface: Result from SiL

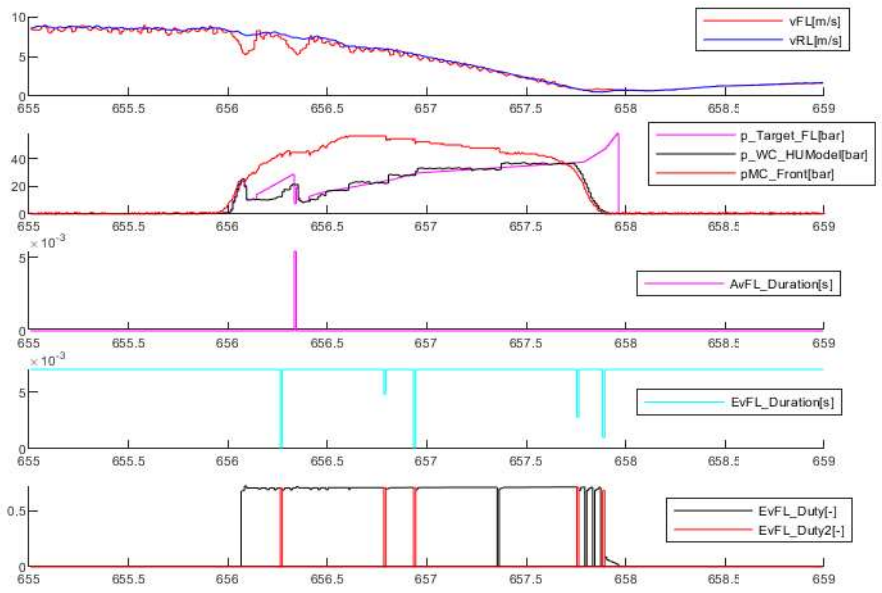


**Figure 5.6:** Braking on Low to high transition surface: Result from bike measurement

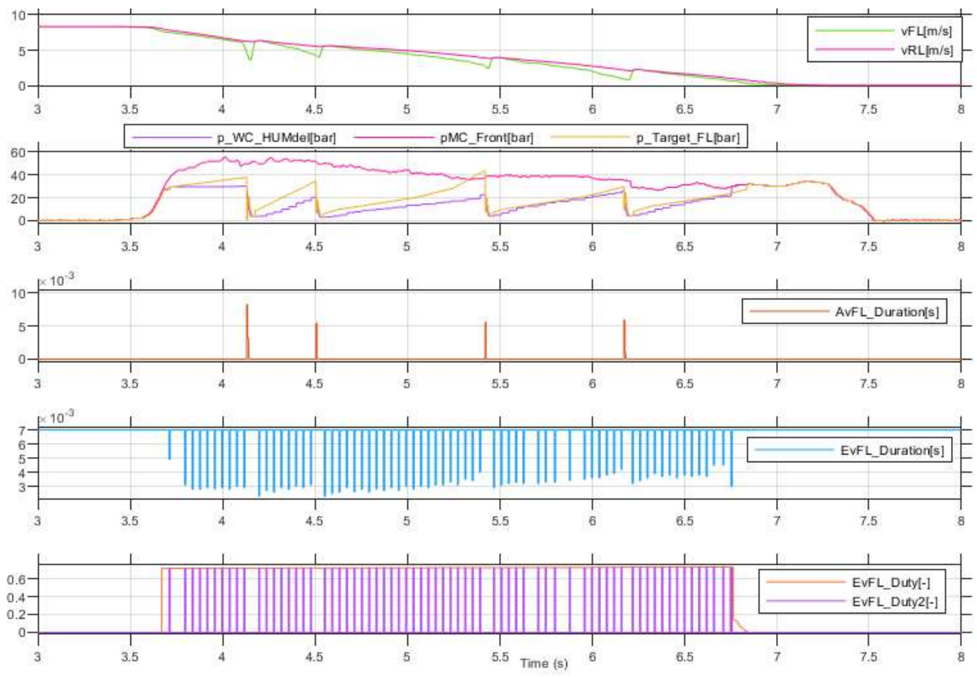


**Figure 5.7:** Braking on high to low transition surface: Result from SiL

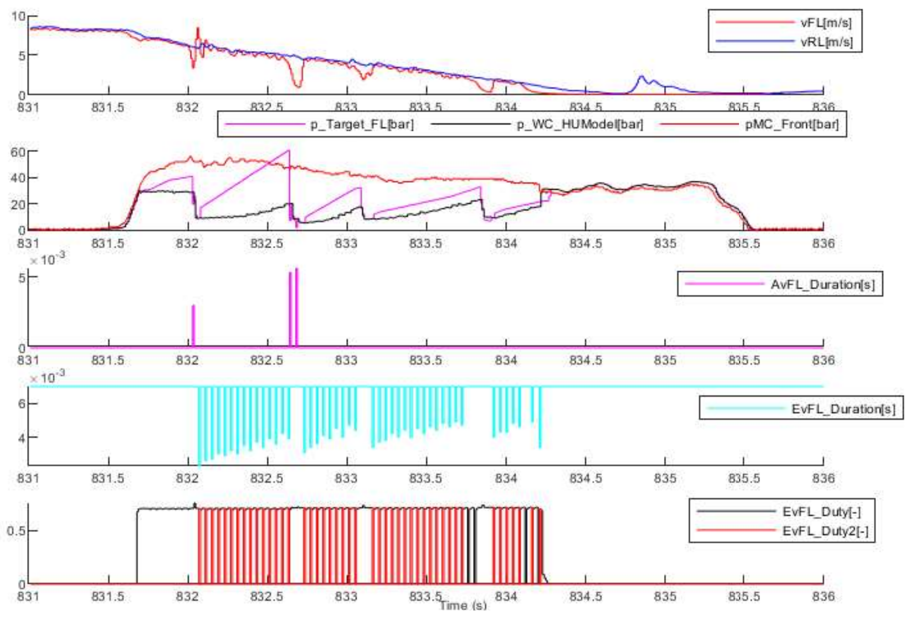


**Figure 5.8:** Braking on high to low transition surface: Result from bike measurement

## 5.2 Sensitivity Analysis with SiL simulation

The aim is to identify the KPIs for ABS and then perform a sensitivity analysis to see the impact on these KPIs by varying the environment and bike parameters and to derive useful results through SiL simulation.

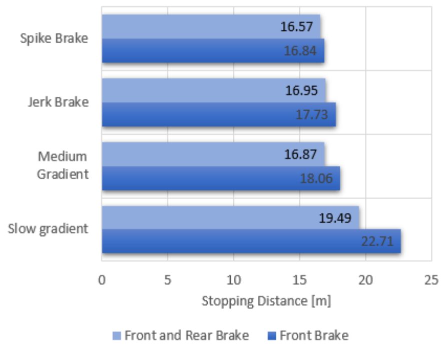
In [Beu20], the author has derived certain KPIs to assess the eBike ABS performance, and we have considered two KPIs that are important to assess the performance of the ABS controller in SiL, which are the stopping distance and the average deceleration. Stopping distance refers to the distance traveled since the start of braking until the bike is at a standstill, and average deceleration is the mean value of the achieved deceleration during the braking duration. Furthermore, the stopping distance and the average deceleration are measured after the brake input and until either the brake is released or the vehicle is at a standstill. Moreover, the surface slope percentage, height of eBike COG, and height of rider COG are some of the examples for the environment and bike parameters, and the aim is to find the influence on the KPIs by varying these parameters. Additionally, all the maneuvers considered for sensitivity analysis were executed using the developed automated VTC execution feature, and hence this additionally evaluates the automated VTC execution feature.

### 5.2.1 Stopping distance and average deceleration at different brake inputs

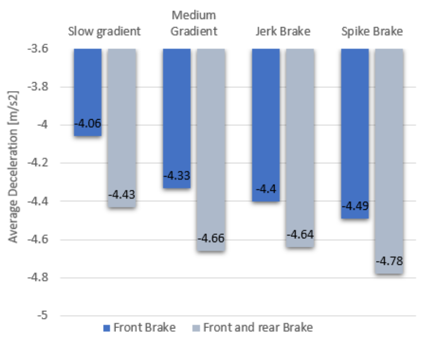
**Test objective:** To measure the stopping distance and average deceleration at different types of brake inputs.

**Test Setup:** Braking on a high-friction surface is considered with an initial speed of 47 km/h.

**Analysis:** Figures 5.9a and 5.9b show the stopping distance and deceleration achieved, respectively, with four different kinds of brake inputs. During spike braking, the stopping distance is comparatively less and a better deceleration is achieved, whereas during slow braking, the stopping distance is greater as there is a loss in the braking performance due to the slow gradient of the input brake pressure. Additionally, the stopping distance and average deceleration achieved are better when both the front and rear brakes are applied in comparison to when only the front brake is applied.



(a) Stopping distance at different brake inputs



(b) Average deceleration at different brake inputs

**Figure 5.9:** Stopping distance and average deceleration at different brake inputs

### 5.2.2 Maximum supported speed from ABS

**Test objective:** To check the maximum speed at different slope percentages where the ABS can prevent nose-over.

**Test Setup:** Braking on a high-friction surface and applying a medium-gradient brake input by varying the down-slope value.

**Analysis:** Figure 5.10 shows the graph for speed at the start of the braking at different down-slope values. The results derived from the SiL show that the ABS controller can prevent the nose-over until a speed of 89 km/h at zero slope. Additionally, the ABS controller cannot prevent the nose-over at a slope greater than 33 percent, even at a lower speed. The reason could be that the rear wheel lift-up is so fast that no matter how much the ABS tries to reduce the pressure, the nose-over cannot be prevented. Although these high speeds cannot be achieved during real bike testing, SiL can be used to derive such results, which are dangerous to perform on the eBike.

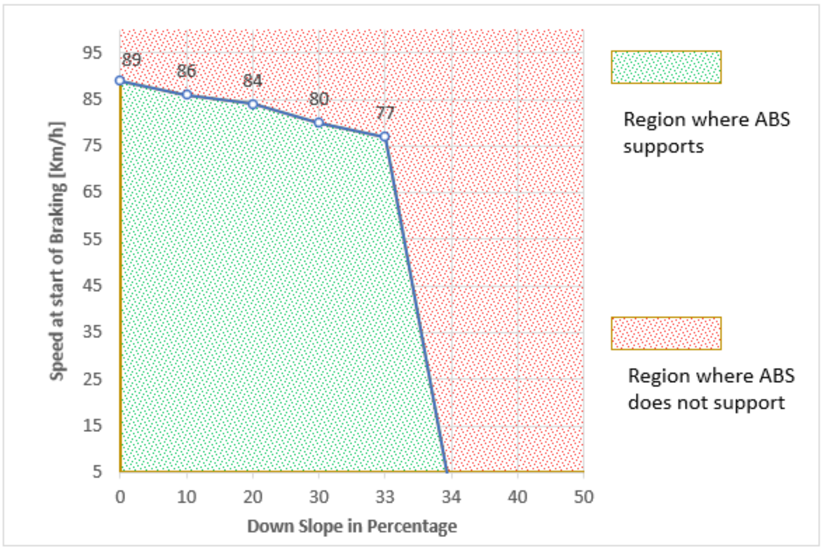


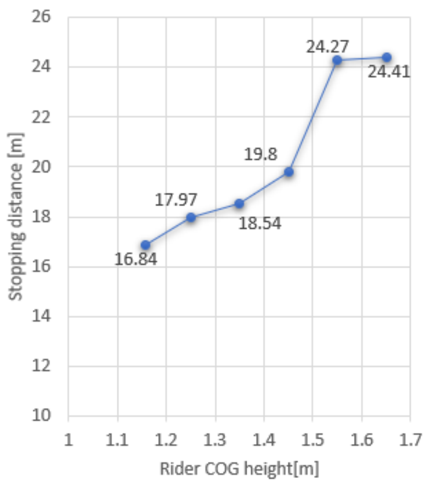
**Figure 5.10:** Speed at different down slope values

### 5.2.3 Impact of the rider COG on KPIs

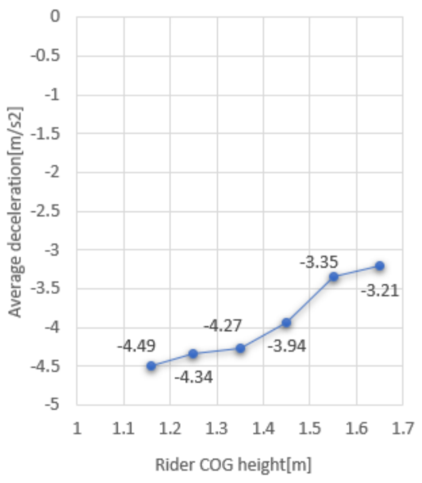
**Test objective:** To vary the height of the rider COG to simulate a standing rider and to observe the impact on the KPIs.

**Test Setup:** Braking on a high-friction surface with no slope and applying a spike brake input with an initial speed of 47 km/h.

**Analysis:** Figures 5.11a and 5.11b show the impact of varying the height of the rider's COG on the stopping distance and average deceleration, respectively. The default height of the rider's COG when the rider is riding the eBike is considered to be 1.135m from ground level. The results derived from the SiL show that as the rider's COG increases, the stopping distance increases, and the average deceleration achieved subsequently decreases. The reason is that as the rider's COG increases, the overall COG of the eBike shifts in height, and the tendency of lift-up increases, due to which the ABS has to perform more pressure reduction to counter this and prevent rear wheel lift-up. Due to this, the stopping distance increases and the average deceleration achieved decreases.



**(a)** Stopping distance vs height of rider COG



**(b)** Mean deceleration vs height of rider COG

**Figure 5.11:** Mean deceleration and stopping distance vs rider COG

## 5.3 Evaluation of SiL for IMU

### 5.3.1 SiL model evaluation for IMU-related ABS implementation

In this section, we evaluate the developed SiL model by identifying a stable region, switching the inputs from the replay data to the simulation, and then performing a high mue and a low mue maneuver to observe the behavior of the ABS controller. The main criteria to evaluate SiL with an IMU is that the pitch angle estimated from the ABS should match the actual pitch angle from the bike model. The necessary preconditions to understanding the graphs are as follows: Switching of the inputs takes place at time 238; braking starts at time 240. The speed at the time of switching is 8 m/s. The signal names, their units, and their meaning are described in Table 5.2.

| Signal Name | Unit | Description |
|---|---|---|
| vVeh | m/s | The overall bike velocity calculated from the ABS |
| vFL | m/s | Velocity of the front wheel |
| Sim_vRL | m/s | Velocity of the rear wheel fron the bike model |
| pTarget_FL | bar | Target pressure requested from the ABS controller |
| p_MC | bar | Pressure applied from the rider |
| p_WC_HUModel | bar | Pressure applied at the wheel caliper. This is calculated from the HU model |
| PitchAngle_reference | deg | Reference pitch angle from the simulation |
| PitchAngle_Estimated | deg | Estimated pitch angle from the ABS controller |
| RollAngle_reference | deg | Reference roll angle from the simulation |
| RollAngle_Estimated | deg | Estimated roll angle from the ABS controller |
| GyroY_Reference | rad/s | Pitch rate from the simulation |
| GyroY_Input | rad/s | Pitch rate provided as input to the ABS controller. This includes the defined errors |
| GyroX_Reference | rad/s | Roll rate from the simulation |
| GyroX_Input | rad/s | Roll rate provided as input to the ABS controller. This includes the defined errors |
| GyroZ_Reference | rad/s | Yaw rate from the simulation |
| GyroZ_Input | rad/s | Yaw rate provided as input to the ABS controller. This includes the defined errors |
| gMonFail | bool | This is a signal calculated from ABS to monitor for the failures in IMU signals. |

**Table 5.2:** Signals required for evaluation of SiL model for IMU

Figure 5.12 shows a high mue braking maneuver performed after switching the inputs at time 238. It can be seen that the pitch angle and the roll angle estimated from the ABS controller match the reference values. Additionally, the ABS controller uses the front wheel speed along with the estimated pitch and roll angle values to detect a tendency for rear wheel lift-up and thereby reduce the pressure to maintain the stability of the eBike. Additionally, Figure 5.13 shows a low mue braking maneuver where the ABS performs a wheel slip control to avoid the front wheel locking due to the high pressure applied. The estimated values for pitch and roll angle match the reference values from the simulation environment.

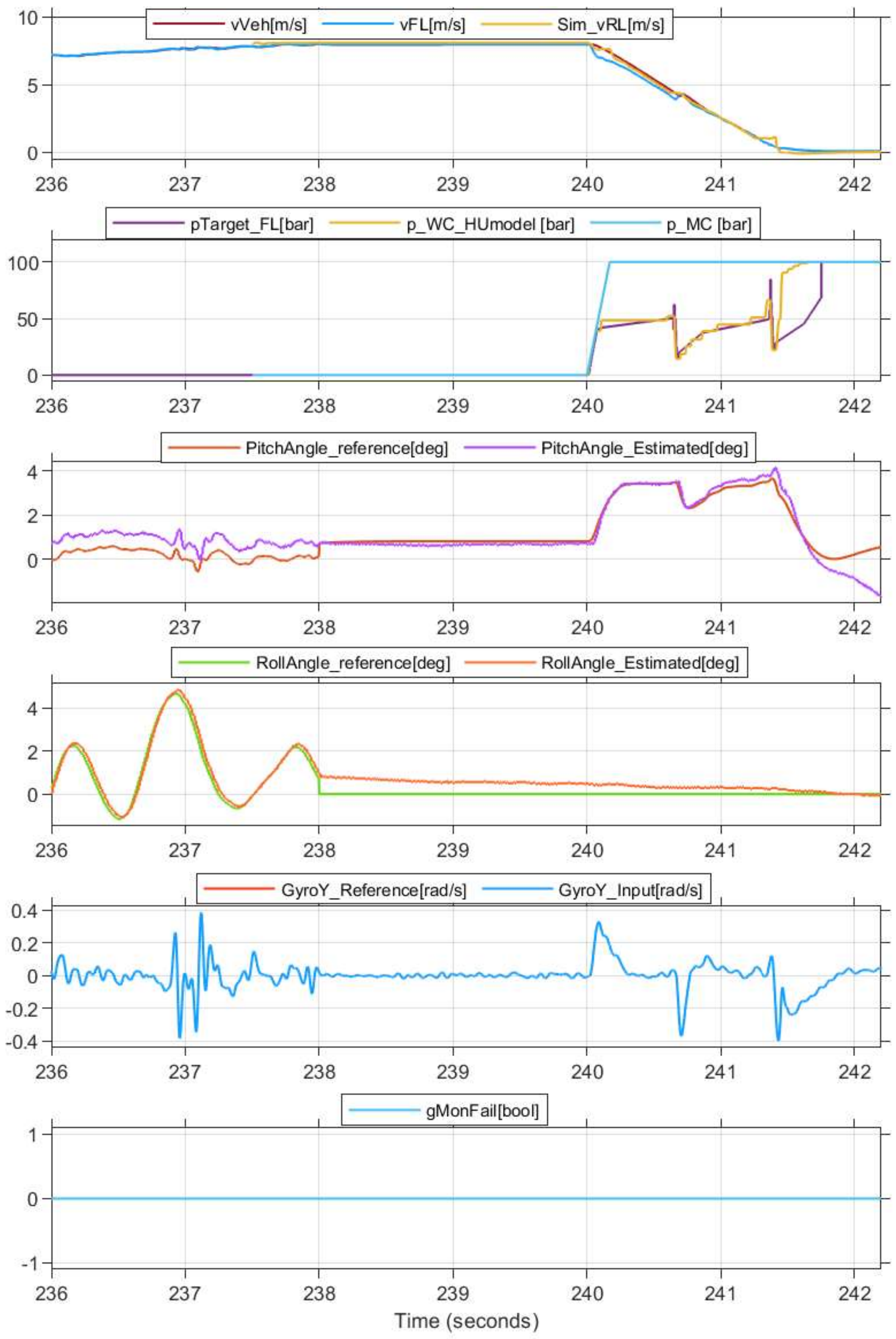


**Figure 5.12:** Braking on a high friction surface

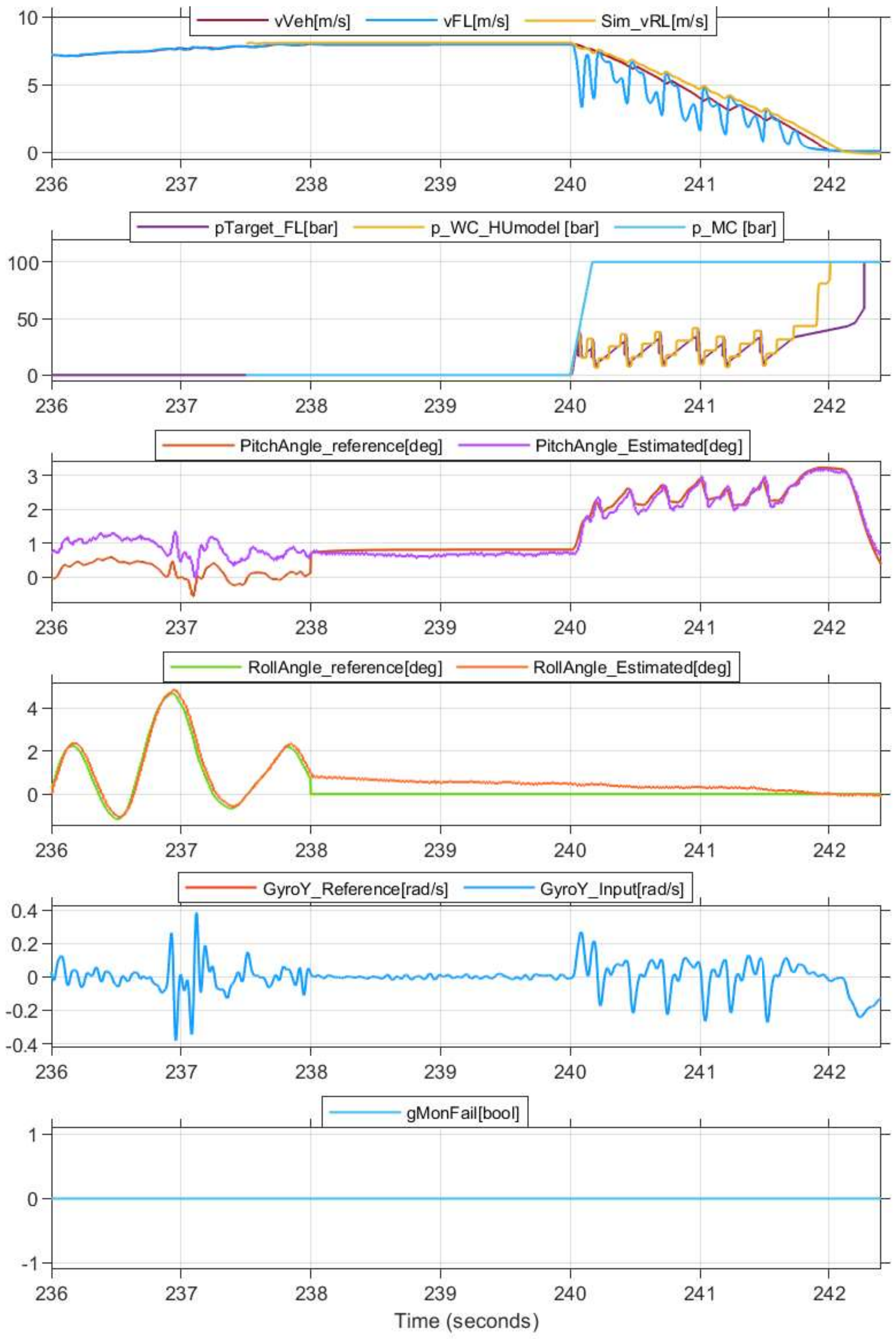


**Figure 5.13:** Braking on a gravel surface

## 5.4 Sensitivity Analysis by inducing IMU errors

In this section, we evaluate the ABS controller by inducing errors into the IMU signals given as input to the ABS. We primarily focus on inducing errors in the gyroscope signals by adding an offset bias to the reference signal (both positive and negative offsets). The distorted signals result in an incorrect estimation of the pitch and roll angle values. Furthermore, the ABS performance degrades due to the incorrect estimation and is depicted by various graphs.

### 5.4.1 Offset bias on the pitch rate (GyroY) signal

Here we add a constant offset bias of 20 deg/s to the pitch rate (GyroY) signal of the gyroscope. The offset is added for a period of 3.5 seconds between times 237 and 240.5 of the execution. Sensor disturbance for a short duration can be the cause of this high offset. The braking is executed on a high friction surface with a spike brake input. Figure 5.14 depicts the ABS braking maneuver with a constant offset bias of 20 deg/s. The offset results in an incorrect estimation of the pitch angle from the ABS controller, which results in frequent ABS interventions as it assumes a higher pitch angle. This scenario is termed *over-braking*. Due to this *over-braking*, we lose out on performance as the stopping distance increases and the average deceleration is lower in comparison to the error free situation. Moreover, the ABS controller monitors for such deviations in pitch angle, and during such deviations, the estimated pitch angle falls back to zero, as can be seen between times 241 and 242. The signal gMonFail is set in case a failure is identified by the ABS controller.

Figure 5.15 shows the ABS braking when a negative offset of 20 deg/s is added to the pitch rate signal. During this scenario, the pitch angle estimation deviates strongly from the reference signal. Even though there is a tendency for lift-up, ABS cannot identify this as the estimated pitch angle is around -50 deg due to the negative offset bias. This results in a bigger lift-up at the rear wheel in comparison to the error free braking maneuver.

### 5.4.2 Offset bias on the roll rate (GyroX) signal

Here a positive offset of 20 deg/s is added to the roll rate (GyroX) signal for a duration of 3.5 seconds between times 237 and 240.5 and is shown in Figure 5.16. This results in an incorrect estimation of roll angle from the ABS controller. The estimated roll angle deviates from the reference value by approximately 60 degrees. Due to this high deviation, the reference velocity calculation is severely affected, and ABS assumes that the braking is performed on a low friction surface, which results in severe under-braking of the front wheel and an increased stopping distance.

### 5.4.3 Offset bias on the yaw rate (GyroZ) signal

Here a positive offset of 20 deg/s is added to the yaw rate (GyroZ) signal for a duration of 3.5 seconds between times 237 and 240.5 and is depicted in Figure 5.17. This results in an incorrect estimation of roll angle and pitch angle from the ABS controller. The estimated roll angle deviates from the reference value by approximately 10 degrees, whereas the pitch angle deviates by 2 degrees. The incorrectly estimated roll angle might influence the ABS to assume that the brakes are applied while riding in a curve. Generally, the ABS controller is less sensitive when braking in a curve as the tendency of hard braking in a curve is very low, and due to the wrong estimation, ABS performance degrades.

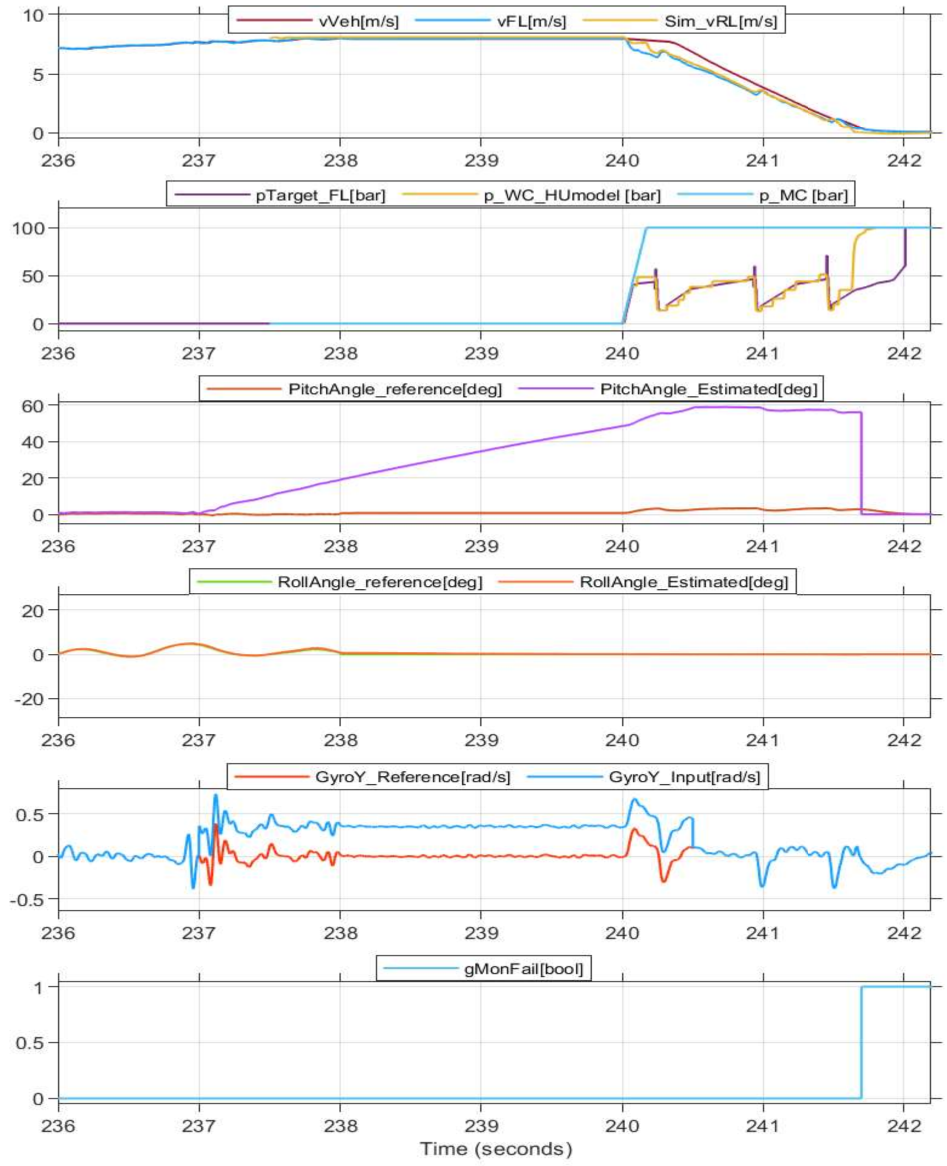


**Figure 5.14:** Offset bias of 20 deg/s on pitch rate signal, high mue maneuver

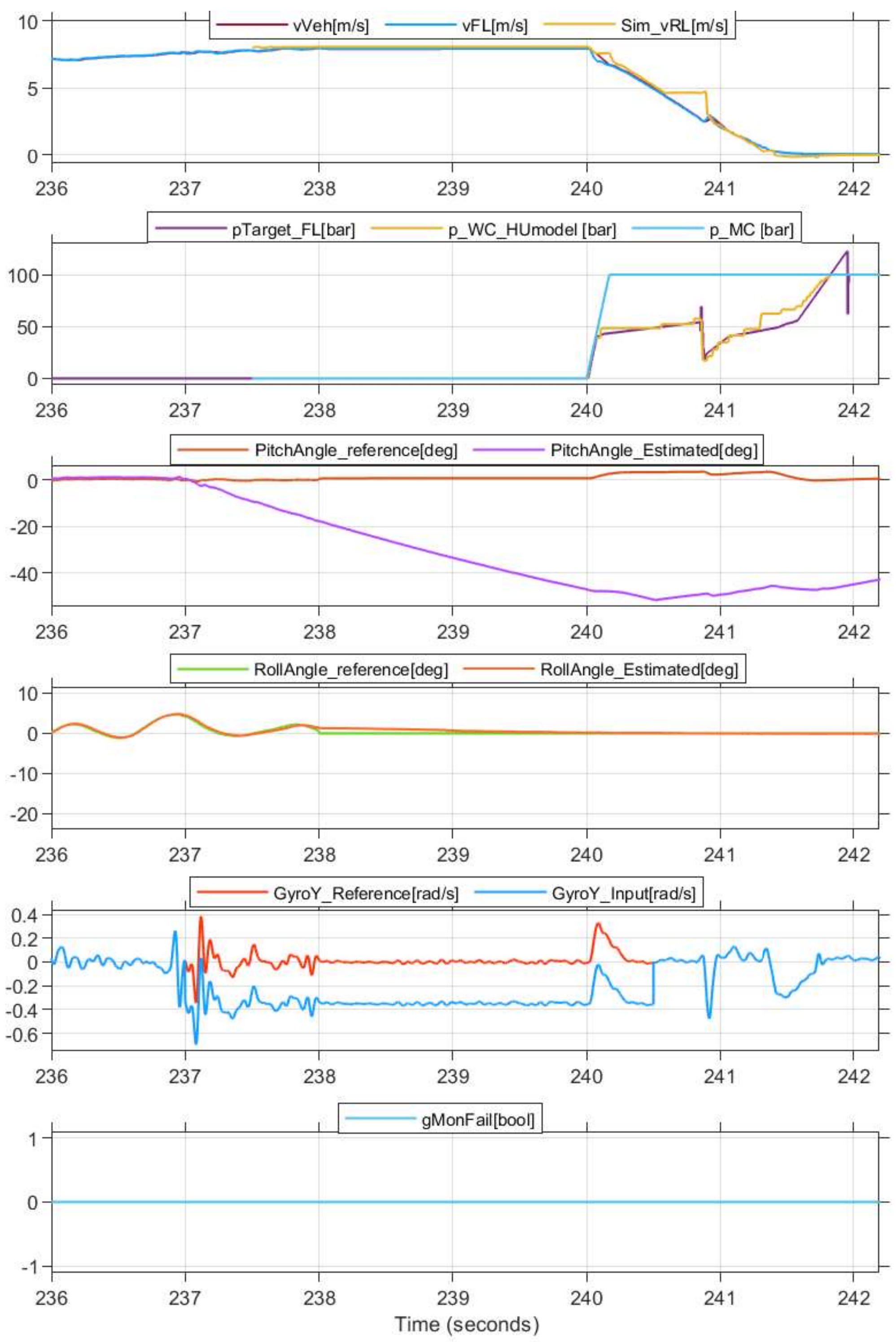


**Figure 5.15:** Offset of -20 deg/s on pitch rate, high mue maneuver

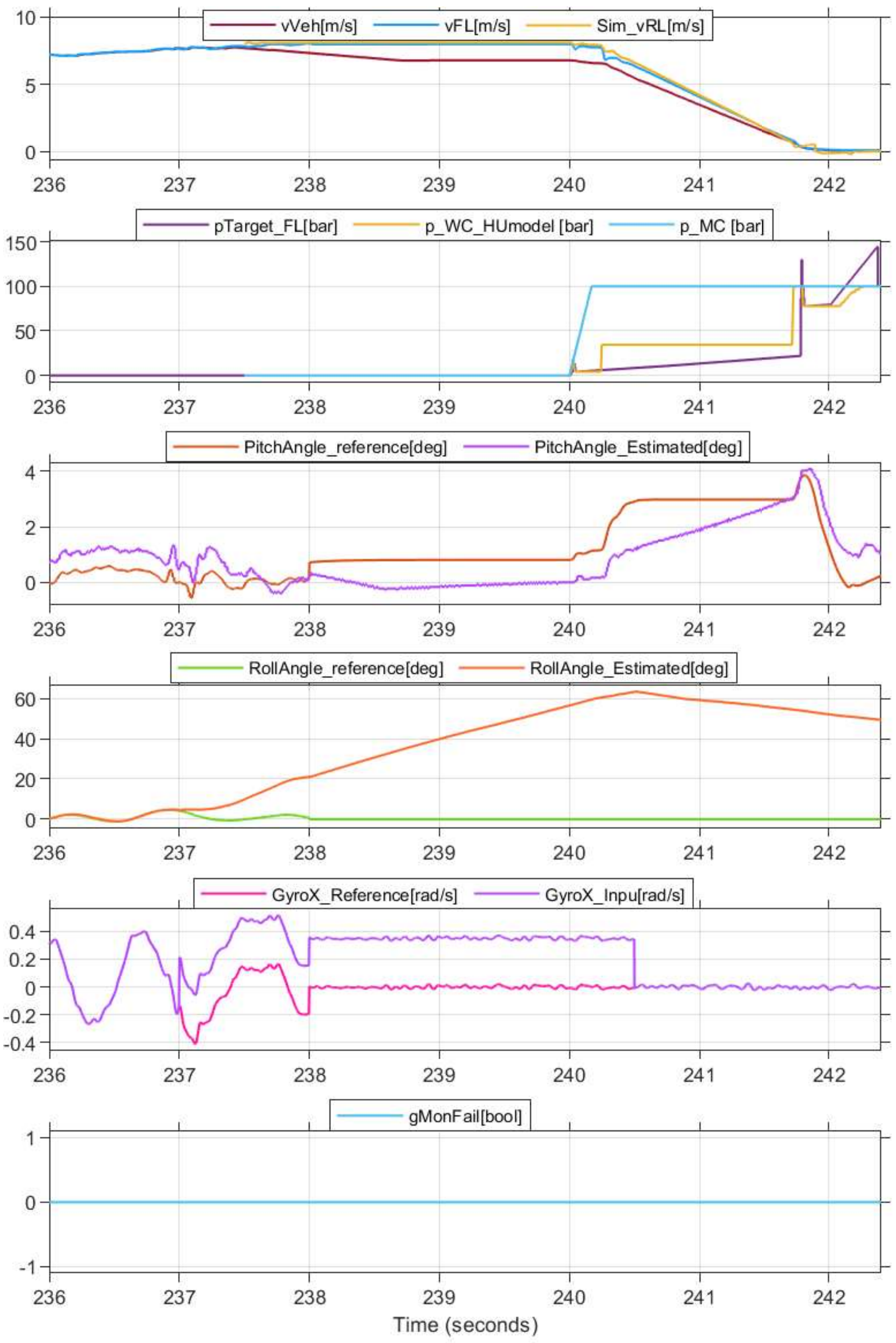


**Figure 5.16:** Offset of 20 deg/s on roll rate, high mue maneuver

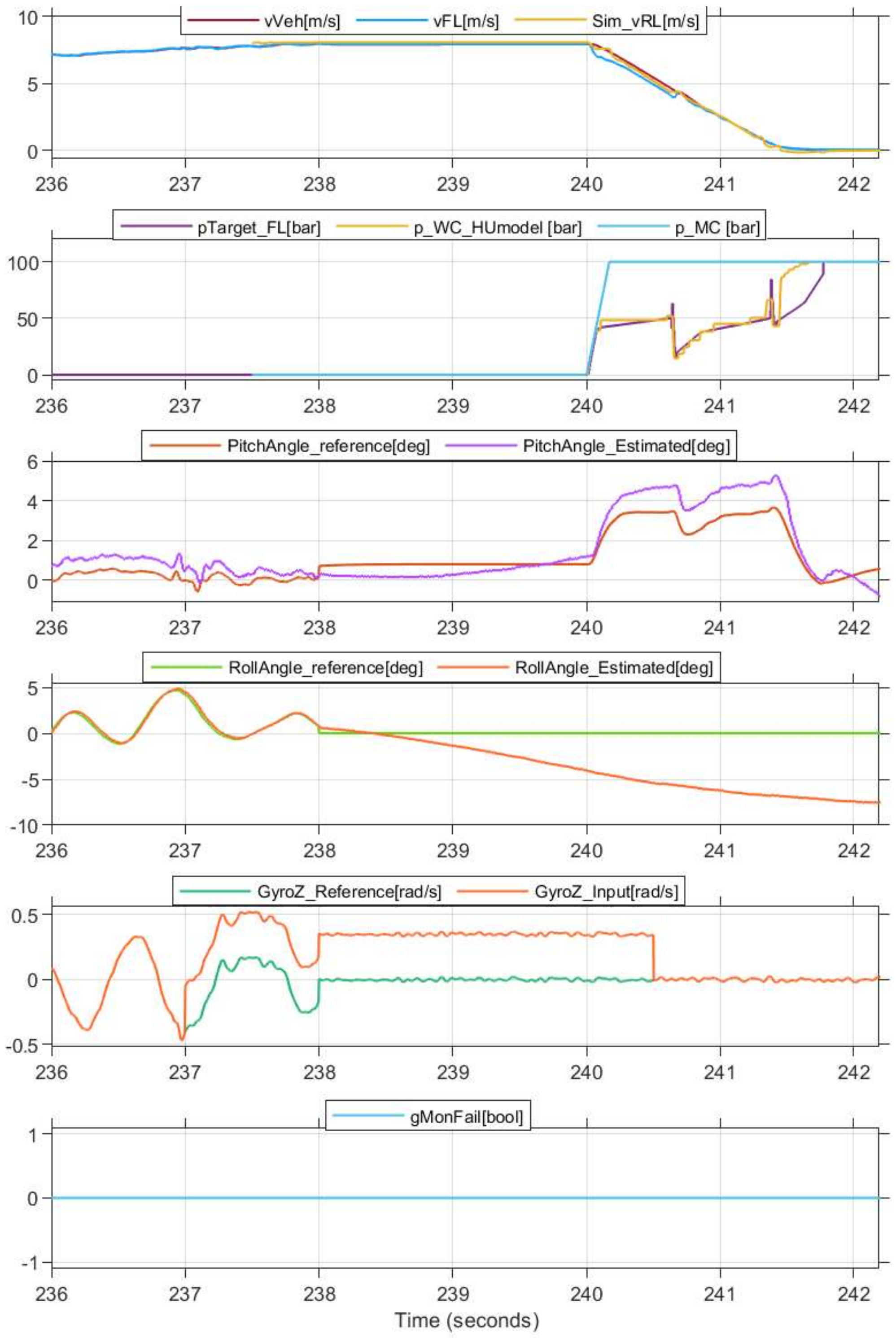


**Figure 5.17:** Offset of 20 deg/s on yaw rate, high mue maneuver

# 6 Conclusion and Outlook

Through the course of this thesis, we mainly developed a SiL simulation environment based on the framework of CSSim. The advantage of using the developed SiL model is that the engineers do not entirely have to depend on the bike tests to validate the ABS software. SiL can be used both during the development phase for prototyping new algorithms and during the release phase to run the VTCs in the simulation rather than on the real bike. On the contrary, SiL simulation results might be a little bit off compared to real-world testing. VTCs are more relevant to understand the riding dynamics. Additionally, to meet the legal requirements, vehicle testing is still needed. Furthermore, SiL can act as a support for engineers to reduce the burden of doing vehicle tests, even for a small software change.

Additionally, a second SiL environment, which includes an IMU sensor model adapted for the bike model, is also achieved. This can be used to induce errors related to the IMU and then assess how these errors impact ABS braking in a variety of situations. This tool is crucial as it eliminates the need to induce errors on a physical bike, which can be dangerous and catastrophic for the rider. We first used the replay data from the bike measurement as inputs so that the state variables could be correctly estimated from the ABS controller, and later we switched the inputs to simulation by identifying a stable region. As an evaluation, the pitch angle estimated from the ABS controller matched the reference pitch angle from the simulation.

The *first question* [Q1] is addressed by defining the simulation maneuvers using the framework of CSSim. We identified the relevant inputs and their definition in simulation maneuvers and usage within the simulation environment. Standard VTC definitions are referred to, and simulation maneuvers are defined primarily for high mue, low mue, high to low, and low to high transition surfaces. Furthermore, maneuvers during different downslope values and different types of brake inputs are also defined. Additionally, we utilized the Jbatch feature of CSSim to execute the maneuvers in an automated manner. All the results generated for evaluation and sensitivity analysis are executed through automated VTC execution. The *second question* [Q2] is addressed by evaluating the SiL results with bike measurements. The pressure input and the initial speed are set the same as in bike measurements. We comprehensively evaluated four different riding scenarios, including high mue, low mue, high to low, and low to high transitions. The results for high mue, low mue, and mue transitions were very similar to those from the bike measurement, with little deviation. The deviation could be due to the fact that the signals in SiL do not have any additional noise, whereas in a real bike, there is some noise. The environmental parameters could differ during the bike test; for instance, the weight of the rider and the riding position could drastically vary the behavior of ABS. To answer the *third question* [Q3], we primarily evaluated the influence of offset bias errors on ABS state estimation and braking behavior. We introduced an offset error of +-20 deg/s for a duration of 3.5 seconds individually on the pitch rate, roll rate, and yaw rate signals. The pitch angle estimation changes by 60 degrees with the positive pitch rate offset. This results in severe overbraking because the ABS thinks that there is a high pitch angle and tries to reduce the pressure. On the other hand, the pitch angle deviates by -60 degrees during negative offset, and here there is severe liftup due to the underbraking as the ABS thinks there is no liftup due to the incorrect estimation.

Finally, the *fourth question* [Q4] is answered through the sensitivity analysis by varying the bike and environment parameters and measuring the KPIs. We identified the stopping distance and average deceleration achieved as the main metrics to evaluate ABS in SiL, as the impact on these metrics due to the variation of any parameters can be clearly measured and analyzed in SiL. Furthermore, rear wheel liftup duration could be one more metric to assess performance in high mue braking but is not addressed in our work.

## Outlook

The following improvements and additions to the thesis work are possible:

- The SiL model can be extended to simulate other riding scenarios like rough roads, bumps, snow, etc. The tire characteristics and the bike model should be reworked according to the surface.
- A 3D bike model is required to simulate riding through a curve. This helps in simulating IMU errors related to roll angle and sideslip angle.
- We focused on only the offset errors of the gyroscopes in this thesis. This can be further extended to study the impact of other IMU errors like random walks, noise, and bias instability of the accelerometer and gyroscope on the ABS algorithm.
- The entire SiL environment can be placed in an automated environment where the developer can create a new software version and instantly verify the software by performing various tests through the SiL simulation.

# Bibliography

[Apt] Aptive. *What Is hardware-in-the-loop testing?* URL: https://www.aptiv.com/en/insights/article/what-is-hardware-in-the-loop-testing (cit. on p. 27).

[AUT] AUTOSAR. *AUTOSAR*. URL: https://www.autosar.org/ (cit. on p. 28).

[AWS14] H. Altinger, F. Wotawa, M. Schurius. “Testing methods used in the automotive industry: results from a survey”. In: *Proceedings of the 2014 Workshop on Joining AcadeMiA and Industry Contributions to Test Automation and Model-Based Testing - JAMAICA 2014* (2014). DOI: https://doi.org/10.1145/2631890.2631891 (cit. on p. 27).

[Bar21] K. Bardyova. “Modellierung des hydraulischen Aktors des Antiblockiersystems am elektrifzierten Fahrrad.” Msc Thesis. 2021 (cit. on pp. 16, 33).

[Beu20] T. Beutel. “Leistungsbewertungskonzept Für Antiblockiersysteme Am Elektrifizierten Fahrrad”. Bachelor Thesis. Aug. 2020 (cit. on p. 58).

[dSp] dSpace. *HIL Testing for Electronic Control Units (ECU)*. URL: https://www.dspace.com/en/pub/home/applicationfields/foo/hil-testing.cfm (cit. on p. 27).

[ETAa] ETAS. *ASCET-DEVELOPER*. URL: https://www.etas.com/de/portfolio/ascet-developer.php (cit. on p. 29).

[ETAb] ETAS. *Complex engine simulation using LABCAR-MODEL*. URL: https://www.etas.com/en/company/realtimes-2021-complex-engine-simulation-using-labcar-model.php (cit. on p. 27).

[FGW18] D. Felix, D. Görges, A. Wienss. “Experimental Analysis of Sensor Requirements for eBike Rider Assistance Systems”. In: *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. 2018, pp. 1–6. DOI: 10.1109/ICVES.2018.8519585 (cit. on p. 30).

[Gmb11] R. B. GmbH. *CSSim (Chassis Systems Simulation)*. 2011 (cit. on pp. 16, 29).

[Gmb22] R. B. GmbH. *E-bikes the Bosch Way – Drive Expertise for the Bicycle*. 2022. URL: https://www.bosch.com/research/know-how/success-stories/e-bikes-the-bosch-way-drive-expertise-for-the-bicycle/#:~:text=Bosch%5C%20eBike%5C%20Systems%5C%20has%5C%20been (cit. on p. 15).

[Gmb23a] R. B. GmbH. *e-bike ABS: Trust in the hard stop*. 2023. URL: https://www.bosch.com/stories/e-bike-abs/ (cit. on p. 15).

[Gmb23b] R. B. GmbH. *eBike ABS: sicheres Abbremsen für Elektrofahrräder*. 2023. URL: https://www.bosch-ebike.com/de/produkte/abs (cit. on p. 20).

[JKL16] S. Jeong, Y. Kwak, W. J. Lee. “Software-in-the-Loop simulation for early-stage testing of AUTOSAR software component”. In: *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. 2016, pp. 59–63. DOI: 10.1109/ICUFN.2016.7536980 (cit. on p. 28).

[Man22] F. Mangold. “Anpassung, Parametrierung Und Validierung Eines Modells Zur Simulation Von ABS-Bremsvorgängen Am Pedelec”. MSc Thesis. 2022 (cit. on pp. 16, 33).

[Mat] Mathworks. *Simulink - Simulation and Model-Based Design*. URL: https://in.mathworks.com/products/simulink.html (cit. on pp. 29, 49).

[MW14] M. Mitschke, H. Wallentowitz. *Dynamik der Kraftfahrzeuge*. Vol. 5. Springer-Verlag, Nov. 2014. ISBN: 9783658050689 (cit. on p. 19).

[Zwe23] Zweirad-Industrie-Verband. *Marktdaten 2022*. Mar. 2023. URL: https://www.ziv-zweirad.de/marktdaten/detail/article/marktdaten-2022/ (cit. on p. 15).

All links were last followed on May 31, 2023.

## Declaration