CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Performance Evaluation of Active Safety Sensor Models for Virtual Verification Tool Chain

Master's thesis in Embedded Electronic System Design

Xin Cao
Xin Zhang

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

Master's thesis 2019

Performance Evaluation of Active Safety Sensor Models for Virtual Verification Tool Chain

XIN CAO

XIN ZHANG

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2019

Performance Evaluation of Active Safety
Sensor Models for Virtual Verification Tool Chain
XIN CAO
XIN ZHANG

Supervisor at Chalmers: Jan Jonsson, Department of Computer Science and Engineering
Advisor: Siddhant Gupta, Volvo Car Corporation
Examiner: Per Larsson-Edefors, Department of Computer Science and Engineering

Master's Thesis 2019
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2019

Performance Evaluation of Active Safety
Sensor Models for Virtual Verification Tool Chain
XIN CAO
XIN ZHANG
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Today's active safety systems within automotive industry are developing and becoming more complex. Along with this development, much more testing and validation of these systems are required. Due to their high time and cost consumption, Computer Aided Engineering (CAE) simulations are playing a more vital role in the validation stage. Since the performance of active safety systems strongly depends on the performance of sensors, it is important to validate the sensor models for the development of related active safety functions. Traditionally, CAE simulation uses ideal sensor models that reflect the ground truth of measured data. However, active safety systems need sensor models to capture the same inaccuracies as the physical sensors to achieve better agreement with tests in real traffic. As a result, sensor error models that capture the inaccuracies are developed. The combination of ideal sensor model and sensor error model is used to imitate the output of an integrated radar and camera system.

In this thesis work, a method for validating sensor error models is presented. The whole validation process is divided into two parts: sensor level validation and function level validation. In sensor level validation, we verify the standalone sensor error model by comparing the simulated sensor error with the measured error using several comparison metrics. In function level validation, the sensor error model is validated under different CAE environment including Model-in-loop (MIL) and Hardware-in-loop (HIL) frameworks. Two system responses that can indicate the performance of active safety systems are proposed. Then the experimental results of these responses are compared with measured data as well. In this evaluation level, we also propose several validation metrics to show the discrepancy between simulated and measured data.

Keywords: statistical sensor error model, sensor validation, validation metric, active safety system, Model-in-loop (MIL), Hardware-in-loop (HIL)

Acknowledgements

We would like to express our sincere thanks towards all the people who gave us support and help during the work, especially:

Siddhant Gupta, our supervisor from the Driver Assistance and Active Safety Group at Volvo Car Corporation, for his valuable support, clear guidance and excellent motivation during this thesis, which helped us to have better understanding in the active safety tool chain as well as sensor validation.

Prof. Jan Jonsson in Department of Computer Science and Engineering at Chalmers University of Technology, our thesis supervisor for his kind support, helpful discussions and advises during the thesis.

Fredrik Björkqvist and **Yury Tarakanov**, managers from Driver Assistance and Active Safety Group at Volvo Car Corporation, for their constant support and advises.

Prof. Per Larsson-Edefors in Department of Computer Science and Engineering at Chalmers University of Technology for being our thesis examiner as well as for all his valuable feedback and advises.

Other master thesis students in the same group at Volvo, for their encouraging and fun ping-pong sessions.

Xin Cao, Xin Zhang, Gothenburg, 09, 2019

Contents

Acronyms

AEB Autonomous Emergency Braking. x, 1–3, 6, 15–18, 21, 32
AIOHMM Auto-regressive Input/Output Hidden Markov Model. x, 1, 27, 30, 32, 33, 35
ASM Active Safety Manager. ix, x, xiii, 5–7, 15, 16, 18, 20, 21

CAE Computer Aided Engineering. v, x, 1, 5, 15
CAN Controller Area Network. ix, x, 19–21
CCRs Car-to-Car Rear Stationary. x, 10, 21, 27–30, 32, 35
CORA Correlation and Analysis. ix, x, xv, 22–24

ECU Electric Control Unit. x, 5, 7, 18, 20, 21, 35
EEARTH Enhanced Error Assessment of Response Time Histories. ix, x, 22, 24, 25
Euro NCAP European New Car Assessment Program. x, 9, 10, 35

GP Gaussian Process. x, 1, 27, 28, 30, 32

HIL Hardware-in-loop. v, ix, x, xiii, xv, 1, 3, 5–8, 15, 17–20, 33–35

I/O Input/Output. x, 7
IQR Interquartile Range. x, 21, 22
ISO International Organization for Standardization. ix, x, 5, 22, 23, 32–35

JSD Jensen-Shannon Divergence. x, 13, 14

KLD Kullback-Leibler Divergence. x, 12–14
KPIs Key Performance Indicators. x, 3

MIL Model-in-loop. v, ix, x, 1, 3, 5–7, 15–17, 30, 32–35

RadCam Radar and Camera System. x, 2, 3, 5, 9–11, 15, 18, 21
RMSE Root Mean Squared Error. x, 12, 13, 27, 29

SEM Standard Error of the Mean. x, 11
SIL Software-in-loop. x, 5
SPAS Simulation Platform for Active Safety. x, 6, 16, 30, 33
SPI Serial Peripheral Interface. x, 19

UART Universal Asynchronous Receiver-Transmitter. x, 19

VCC Volvo Car Corporation. x, 1, 5–7

List of Figures

List of Tables

1
Introduction

Safety is the highest priority at Volvo Car Corporation (VCC) with zero serious injury vision in a new generation of Volvo car in 2020. Development of active safety systems, which aims to help avoid possible automobile accidents, is the main focus of VCC to achieve this 2020 vision. Sensors are a crucial part of these active safety systems. The raw data from perception sensors such as camera and radar is fused in order to provide an characterization of the traffic environment where the car is driven. The output of this fusion process is a list of detected objects with their status information such as position, speed and acceleration etc. This information is further utilized by active safety functions such as Autonomous Emergency Braking (AEB) function. The performance of these active safety functions massively depends on the performance of the sensors, which have inherent inaccuracies.

Traditionally, active safety functions are assessed in test tracks or in real traffic, where their performance affected by sensor inaccuracies is tested. However, this method involves a large amount of driving and easily exposes driver to dangerous situations. To speed up the product development cycle and avoid dangerous driving, these tests are first rigorously performed in CAE simulation environments.

Having a good agreement between the CAE environments and the physical environment is vital to achieve high confidence in simulations. Sensor modelling is a typical method that is used to model the real sensor behavior in such virtual simulations. In order to achieve high correlation with in-car tests, the simulated sensor model has to possess the same inaccuracies as the real sensor. This in turn requires the development of sensor error model where sensor errors are captured. These models are trained using statistical methods based on a large dataset where sensor responses to real objects and sensor errors are recorded. By adding the error to the output of an existing ideal sensor model which represents the ground truth of detected vehicles, we can model a more realistic sensor behaviour. We term the combination of error model and ideal model as non-ideal sensor model, which is illustrated in Figure 1.1.

1.1 Problem Description and Thesis Objective

This master thesis focuses on so-called black box statistical sensor error models evaluation in both MIL and HIL simulation environments. There are three error models available in VCC for evaluation: Regression model, Gaussian Process (GP) model and Auto-regressive Input/Output Hidden Markov Model (AIOHMM). The

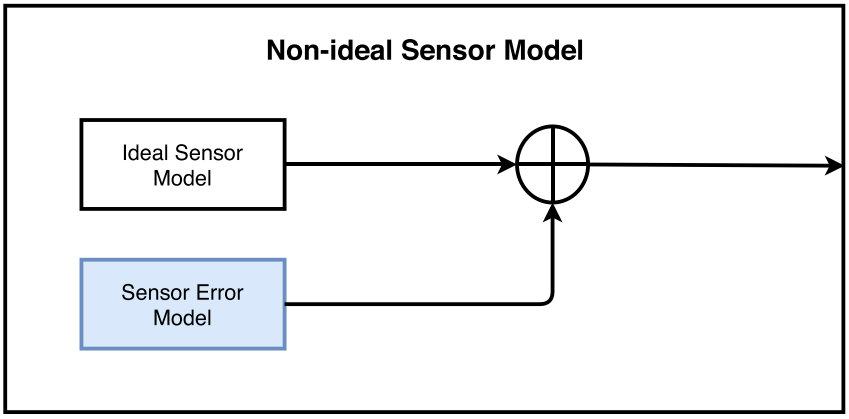

Figure 1.1: Structure of non-ideal sensor model

non-ideal sensor model is used to model one output of integrated Radar and Camera System (RadCam) [1], which fuses outputs from radar and camera in a complicated module by employing data fusion algorithms. This technology and system can help reduce the probability of the traffic accidents and injury, also keep cost lower. Additionally, the RadCam enables various active safety functions (e.g. Adaptive Cruise Control and AEB), which becomes one of the vital parts for car manufacturers and active safety system. This system has been equipped for the product since the start of 2015 Volvo XC90, which usually is mounted under window shield on a vehicle.

The RadCam system reports information of detected objects as well as information about surrounding road environments. Several factors can yield to the inaccuracies of sensor output. For example, the application of analog-to-digital converter in the sensor results in quantization error. RadCam might also be sensitive to the current environment such as temperature and external illuminations. In our thesis, only evaluation of error models of longitudinal distance between host vehicle and target vehicle is considered. In other words, the sensor error models under test in this thesis predict the error of longitudinal distance between host vehicle and target vehicle reported by RadCam.

Before a virtual sensor error model can be applied with confidence in simulation for the development and validation of active safety functions, it is necessary to validate it against the real world. By validation, Oberkampf and Trucano [2] describe it as the process of determining the realism of a virtual model compared to the real world from the aspect of the intended use of this model. The American Society of Mechanical Engineers Standards committee [3] defines model validation as a two step process: First, quantitatively comparing the computational and experimental results for the response of interest. Second, determining whether there is acceptable agreement between the model and experiment for the intended use of the model.

Based on the definition in [2, 3], we proposed to validate our models in terms of two levels: sensor level and function level. In sensor level, the work will focus on validating the agreement between output of standalone sensor error models and

measured sensor error. In function level, AEB function identifies potential collision object ahead of the car depending on longitudinal distance signal from RadCam, so validating the performance of AEB function will be also conducted. Key Performance Indicators (KPIs) on non-ideal sensor model will be compared with the same function KPIs on ideal sensor models and KPIs on field-test data.

Traditionally, people compare simulation data and measured data graphically. Though these comparison is valuable, quantitative metrics are still desired to quantify the discrepancies between simulated and real data. After comparisons, we are supposed to answer the following questions:

- How to quantitatively measure the discrepancies between simulated and real data in both sensor and function level? Can we identify the winning error model that has the best performance based on the quantitative metrics we proposed?

- Do the error models improve the fidelity of the simulation environment compared to ideal sensor model in MIL and HIL frameworks?

1.2 Thesis Outline

The rest of this thesis is organized as follows: Chapter 2 illustrates the simulation environments and systems used in our work. Chapter 3 presents the methodology of sensor level validation. Chapter 4 explains how the simulations are setup for function level validation in detail. Chapter 5 gives the results obtained from both sensor level and function level validations. The conclusions followed by discussions about the future work are described in Chapter 6.

2
Simulation Environment

Traditional system development methodology requires engineers from both system level and software level. System design engineers specify overall system specification and hand it over to software engineers, who will implement the system in required software languages. However, software engineers might have difficulty in interpreting solutions directly from system engineers. Thus, there exists a risk that the solutions might be misunderstood during the implementation. Instead of initially using extensive codes, Model-based Design methodology [4] enables developers to visualize system by utilizing mathematical models that represent multiple components so it is commonly applied in embedded software development.

The development of vehicle simulation environments is a complex process. Various CAE environments can be categorized into Software-in-loop (SIL), Model-in-loop (MIL) and Hardware-in-loop (HIL). The usage of these CAE environments can vary depending on use cases and scenarios being tested. This chapter will start with an introduction of the Electric Control Unit (ECU) where the sensors are deployed. Then we will explain basic idea of MIL and HIL framework and elaborate corresponding CAE environments in the Active Safety department of VCC.

2.1 Active Safety Manager (ASM)

ASM is an ECU in Volvo that includes active safety functions and builds the interface between RadCam and actuators of vehicle. It is a complex real-time embedded system that combines sensors, sensor fusion and various sensor-based active safety functions. Based on International Organization for Standardization (ISO) 26262 [5] hardware design and Autosar software architecture, it allows the flexibility to deal with different active safety functions. High-performance microprocessor from NVIDIA is deployed in ASM since sensor data processing and interpretation of surroundings require high computational demands.

The illustration of ASM is shown in Figure 2.1. Raw sensor data from radar and camera is first fused yielding to a list of classified objects with the information about object status, such as position, speed, acceleration, etc. Our models under test predict the error of longitudinal distance, which is one of these fused objects signals. Apart from object information collected by sensor RadCam, current host vehicle states such as host speed, braking force and steering wheel torque are collected from different components of the vehicle by other related sensors. Based on the object

information and current host vehicle states, ASM will output the next desired states to act as the control signals for other vehicle components like brake and steering. For example, AEB function generates the desired deceleration request signal when faced with emergency situation in order to control the brake. ASM is called our system under test and is integrated in the virtual simulation environments.

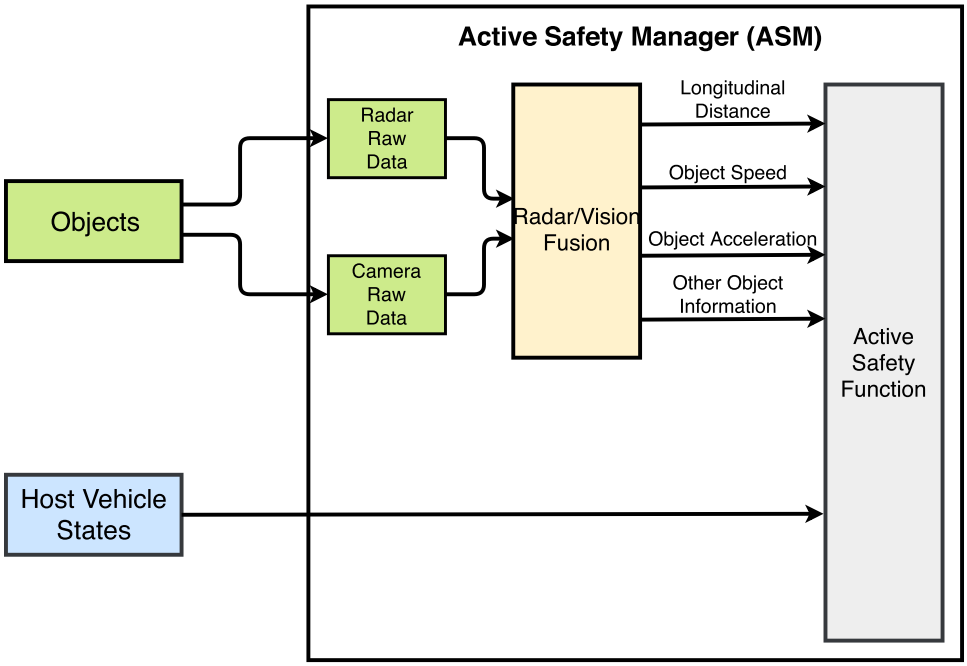

Figure 2.1: Figure of ASM

2.2 Model-in-Loop (MIL)

MIL technique is generally applied at an early stage of function development without including any hardware. It is conducted in an offline model based environment, usually in Simulink environment. The advantage of MIL testing is that it could be easily updated according to different testing requirements, which can reduce the time cost in setting up simulation environments. However, it still has drawback that the simulation results from MIL testing are not as realistic as the results obtained from HIL test.

Simulation Platform for Active Safety (SPAS) is an in-house MIL environment in VCC that supports function testers to set up virtual test. It consists of mathematical models representing different components of entire vehicle, e.g driver, chassis, brake and steering, together with models of surrounding traffic environments, such as roads, objects and traffic signs. An overview of SPAS environment is shown in Figure 2.2. In SPAS, states of the virtual host vehicle are calculated and captured. Ground truth for the relative relationship between host vehicle and target object can be conveniently obtained.

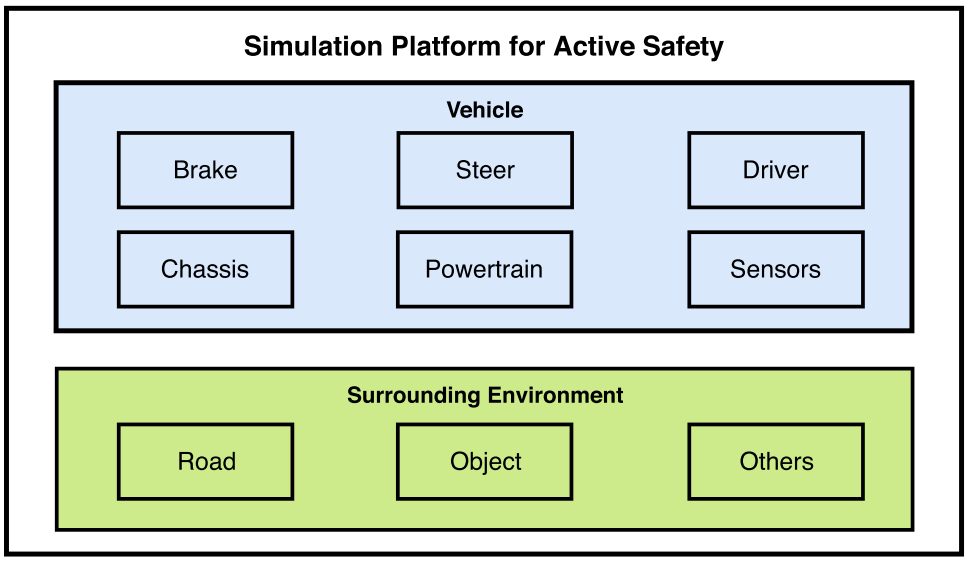

Figure 2.2: Overview of Simulation Platform for Active Safety (SPAS)

2.3 Hardware-in-Loop (HIL)

Hardware-in-loop technique helps create a more realistic simulation environment by including physical embedded system under test within the loop. HIL simulation is capable of finding possible errors that cannot be identified at the MIL stage since it enables simulating the system with the same timing constrains as the real world. Developers can find potential problems before they integrate control system with plants. Therefore this real-time simulation is commonly used to develop and test ECU in the automotive industry.

A typical HIL test system consists of the HIL simulator, software running on host computer and ECU under test. The general structure is illustrated in Figure 2.3. HIL team in Active Safety department of VCC utilizes dSPACE simulator to emulate the real time behaviour of a full virtual vehicle. Simulink vehicle models are built into executable code and loaded to the processors. Meanwhile, dSPACE supplies software for the configuration and the control of HIL test including control desk, model desk and motion desk, which runs on the host computer. ASM is the ECU under test in our case.

The dSPACE simulator includes standard processors and Input/Output (I/O) boards to ensure easier adaptions and expandability. The processor boards deployed in dSPACE are designed for high computational demand simulations and function as the interface between real time models and I/O board. It also supports multiprocessor and multicore setup for large scale real time applications. Multiple I/O boards with easy configurable interfaces are also included to connect the simulator and the ECU. Furthermore, dSPACE offers the Electronic Load Modules to provide realistic simulated loads for the ECUs.

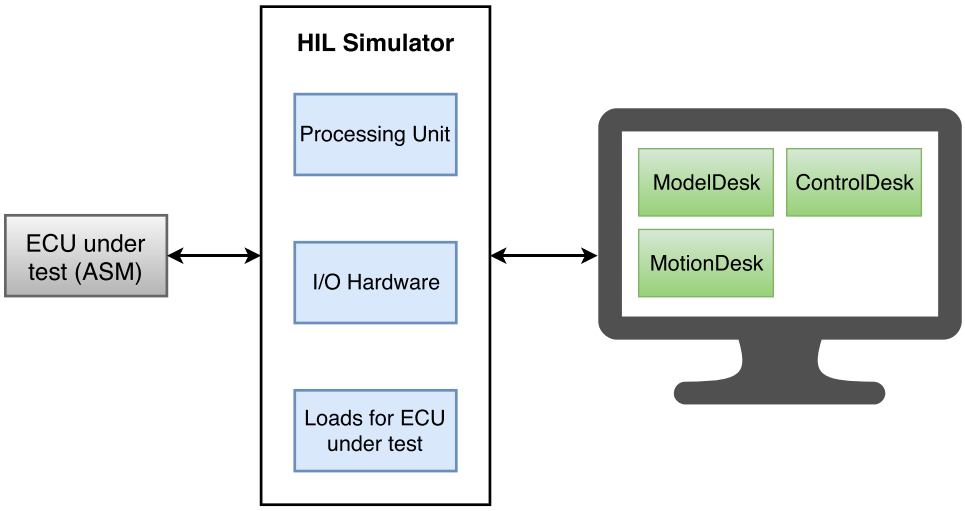

Figure 2.3: Structure of a typical HIL test system

dSPACE also provides the software for the purpose of the creation, configuration and visualization of driving simulation environment. It covers from the generation of traffic scenarios to the simulation of the real time behaviour of each component in the vehicle. These software are also capable of logging and managing test data automatically. By connecting the HIL simulator with the host computer, we are able to set up and control the HIL test.

3

Sensor Level Validation

As mentioned in the first chapter, sensor level validation aims to measure the correctness of the standalone error models. The error models involved in this thesis were implemented using statistical methods [6]. All those models were trained using an expedition dataset which contains detailed Radar and Camera System (RadCam) data and the accurate logging of surrounding traffic environment. The inputs of error models so-called predictors are selected from the accurate data by feature selection method [7]. Those inputs mainly consist of two categories: relative position/speed/acceleration between host and target vehicles, and host states information such as host speed. Given all the predictors, the error models can make the prediction of the relative longitudinal distance error as the output.

Methods presented in this chapter are intended for validation of sensor error models in the automotive industry. Although the sensor errors can be modeled on different levels like raw data level and object list level, the models we focus on in this thesis are the longitudinal distance error models, i.e. fused object list level. Our methods can also be generalized to error models of other signals in fused object level. Furthermore, since the evaluation process includes the comparison among different models, we propose that evaluation should be independent of modeling methods.

The essential steps are listed below, which are also illustrated in Figure 3.1:

1. Real field test data recording. All the environmental data is logged when performing real test drives according to European New Car Assessment Program (Euro NCAP) [8] protocols. Logging data from RadCam is the initial step. Besides, in order to obtain accurate data which can be used as the inputs of models, it is crucial to log environmental data from a high accuracy sensor, see section 3.1;
2. Real field test data pre-processing. Since the real world raw data are generally deficient and containing outliers, pre-processing of the data is a vital method to resolve such issues. See section 3.2;
3. Simulated signal recording. Recording the simulated longitudinal distance error when feeding environmental data from high accuracy sensor to the sensor error models, see section 3.3;
4. Comparison metrics. Comparing the simulated longitudinal distance error with the measured longitudinal distance error by using several comparison metrics. The validation metrics used in the sensor level will be introduced in the section 3.4.

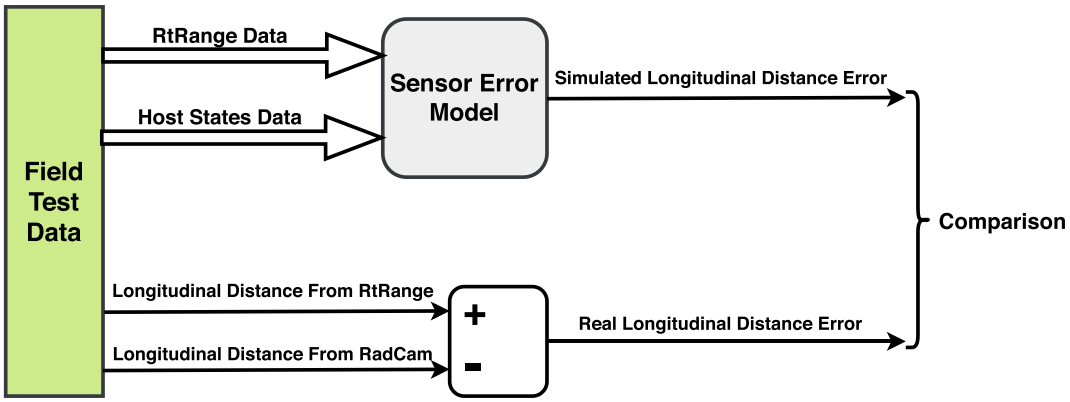

Figure 3.1: The essential steps of sensor level validation

3.1 Real Test Drive Data Recording

Apart from training data, test data that should be independent from the training data is also required for further validation. Therefore, additional field tests under Euro NCAP Car-to-Car Rear Stationary (CCRs) scenarios are conducted to collect necessary test data since Euro NCAP is a typical and widely used standard for testing vehicle safety functions. In CCRs scenarios, the host vehicle moves towards the static target vehicle at various initial speeds, which ranges from 10 km/h to 50 km/h with an increment of 10 km/h.

In addition to RadCam itself, a GPS-based sensor called RtRange [9] is also mounted on test vehicles in order to acquire the accurate driving description of both host and target. It possesses a very high precision with a maximum error of 3 cm. Therefore, we can calculate the ground truth of RadCam longitudinal distance error by subtracting the longitudinal distance reported by RtRange from the one reported by RadCam. Besides, part of the predictors are extracted from RtRange outputs. Apart from RtRange data, we again record the host states information as the remaining predictors of sensor error models.

3.2 Real Field Test Data Pre-processing

Before the collected test data can be applied to evaluate the model performance, a few necessary pre-processing steps are required to prepare for further simulation. There are mainly two steps to ensure more reliable simulation results.

First, since it is risky for statistical models to predict the response of predictors that lie outside the range of predictors in the training data, such test data samples have to be removed as long as one of the predictors is not within the range of its counterpart in the training data.

Second, RtRange and RadCam have different sampling frequencies. Since RtRange data is needed as part of the predictors and also is needed to calculate the ground truth of longitudinal distance error, RtRange data and RadCam data should be processed in a way that they are sampled at the same timestamp. Therefore, RtRange data is down-sampled so as to has the same sampling rate as the RadCam data. Predictors related with the host states are down-sampled as well.

3.3 Simulated Signal Recording

Simulations with standalone error model are performed after the pre-processing of the field test data. The down-sampled predictors are fed into each sensor error model separately in order to acquire the simulated longitudinal distance error signals. Measuring the discrepancy between the simulated error and real error is an essential method to evaluate the error models. To ensure the reliability of validation, we performed multiple measures and comparisons by using various statistical metrics.

Since the models involved in this thesis are all statistical models, the key issue before running simulations is to determine how many runs are sufficient to obtain reliable evaluation. When a model is deterministic, simulation only needs to be performed once. When a model is stochastic, it is not sufficient to represent the real distribution of predictions simply based on few simulations. When we run the stochastic model more times, the distribution of predictions can stand for the real distribution better, and the mean and standard deviation will be more stable. However, we can not run simulations as many times as possible because of time cost. According to [10], authors proposed several methods to identify how many runs are needed. One of the methods uses Standard Error of the Mean (SEM) as a criterion, which represents the error in measuring the mean of a distribution. The SEM indicates that the real mean is within a range of the predicted mean $\pm 1.96 * SEM$ with 95% confidence. If SEM is small enough, we can get a reliable estimation of the real mean. 400 simulation runs are finally chosen in our case. Simulation results for each run are logged for later comparison.

3.4 Comparison Metrics

The statistical comparison metrics used in this thesis can quantitatively measure the discrepancy between simulated logs with real logs. Those metrics can be categorized into deterministic and stochastic groups, as shown in Figure 3.2. Their definitions and formulas will be briefly explained below:

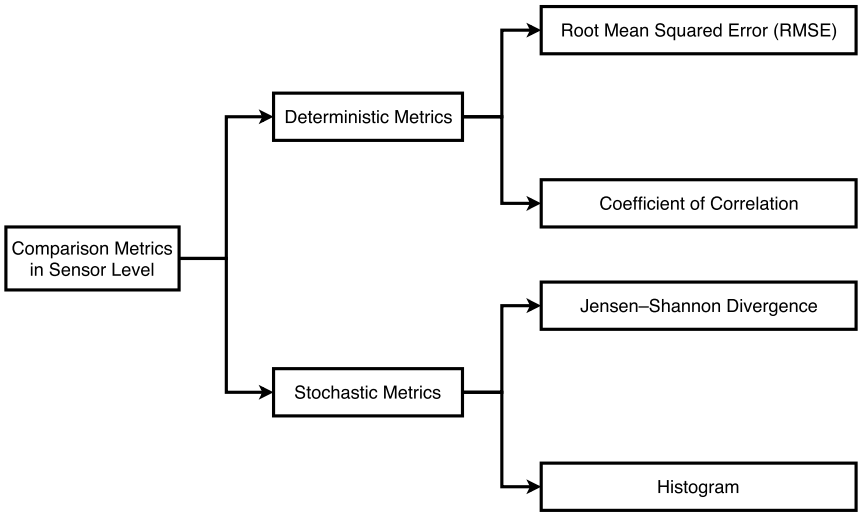

Figure 3.2: Validation metrics in sensor level validation

Root Mean Squared Error (RMSE):

RMSE [11] is a commonly used metric to estimate the difference between predicted values of a model and the real observed values. In other words, it can measure the errors of predicted values. Assume that S is a vector of predicted values and R represents the vector of observed values. The Root Mean Squared Error is defined as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_i - R_i)^2}, \tag{3.1}$$

where N is number of sample size.

Coefficient of Correlation:

The coefficient of correlation [12] measures the linear relationship between two signals or variables. If it equals or is close to 1, it indicates a strong positive linear relationship between variables. On the contrary, a strong negatively linear relationship exists if the coefficient of correlation is -1. For 0, it means that there is no linear relationship between two variables. The coefficient of correlation is defined as:

$$\rho = \frac{n\sum_{i=1}^{n} S_i R_i - \sum_{i=1}^{n} S_i \sum_{i=1}^{n} R_i}{\sqrt{n\sum_{i=1}^{n} {S_i}^2 - (\sum_{i=1}^{n} S_i)^2}\sqrt{n\sum_{i=1}^{n} {R_i}^2 - (\sum_{i=1}^{n} R_i)^2}} \tag{3.2}$$

Kullback-Leibler Divergence (KLD):

Suppose that two probability distributions P and Q. KLD [13] can be used to measure the information lost when using one probability distribution (e.g. P) to

explain the other one (e.g. Q). The Kullback-Leibler Divergence between P and Q is defined as:

$$KLD(P||Q) = \sum_x log(\frac{p(x)}{q(x)})p(x), \tag{3.3}$$

However, this KLD is not a metric since it is not symmetric, which means that $KLD(P||Q) \neq KLD(Q||P)$.

Jensen-Shannon Divergence (JSD):

JSD [14] can measure the similarity between two probability distributions P and Q. The advantage of the JSD than the KLD is that it is symmetrized and smoothed. It indicates how much degrees a probability distribution can explain the other one, rather than the true "distance" between them. The definition is shown in below:

$$JSD(P||Q) = \frac{1}{2}KLD(P||Q) + \frac{1}{2}KLD(Q||M), \tag{3.4}$$

where $M = \frac{1}{2}(P + Q)$.

The square root of the Jensen-Shannon Divergence, also called the Jensen-Shannon Distance, is commonly used as a metric.

Histogram:

The histogram graph [15] can show the probability distribution of a dataset. In a probability histogram, data is broken into groups, and the percentage of the samples that fall into each group is displayed. Histogram is very useful to visually show if the data follows a specific distribution or if two datasets have the similar distribution.

To conclude this section, we introduce several metrics that indicate the model prediction performance by evaluating the difference between predicted values and actual values. If only one metric can be selected to evaluate how good the models fit the real world, RMSE would be the most important one. It is a simple metric but with practical and explainable meaning, which tells the averaged magnitude of prediction errors. However, since the errors are squared first before they are averaged, RMSE is sensitive to large errors. Low RMSE values indicate high accuracy of model prediction. Coefficient of correlation is another commonly used metric that indicates whether there is a strong linear relationship between two variables. An value close to 1 means a strong positive relationship, while a value close to -1 depicts a strong negative relationship. Coefficient of correlation reveals whether there is a similar trend between the simulated data and the actual data by using a value ranging from -1 to 1. But it does not reflect the exact difference between simulated data and real data as what RMSE does.

Apart from RMSE and coefficient of correlation, we also proposed some metrics that are related with comparing the probability distribution of two groups of data.

Histogram is a graphical display of the data distribution. By visually checking the shape of each group of data in the histogram, we can intuitively find which model generates the most similar distribution to the actual one. In addition, we introduce Kullback-Leibler Divergence (KLD) and Jensen-Shannon Divergence (JSD) to quantitatively measure the difference between two probability distribution, which are complement to histogram. Considering that KLD is not a symmetric metric, only JSD is applied for comparison. By using the metrics mentioned in this section, we are able to evaluate the model prediction performance in the sensor level validation.

4

Function Level Validation

Sensor level validation is valuable but not sufficient for the evaluation of sensor models. A model that achieves high accuracy in terms of certain metrics in the sensor level might not result in good function performance as expected. Since sensor fusion is an essential part of Active Safety Manager (ASM), it is highly important to validate sensor error model on a higher level. Oberkampf and Trucano [2] proposed that model validation should be performed in a hierarchy that the responses of tiers right above the model under test should also be evaluated. In this way we can get a more thorough evaluation of model accuracy in terms of its intended use. In this thesis, we suggest to assess the performance of Autonomous Emergency Braking (AEB) function in ASM, which is the direct recipient of RadCam longitudinal distance signal. Even though other active safety functions in ASM accept this signal as well, AEB function is the one influenced mostly by it. Therefore, we mainly consider AEB function performance in function level validation.

This chapter describes three different types of simulations conducted for the validation of error models in the function level. Each simulation is designed for its own unique purpose. We will explain more about their settings below. Furthermore, system responses we selected and corresponding comparison metrics will also be elaborated.

4.1 Standalone Error Models Integration

Since both Model-in-loop (MIL) and Hardware-in-loop (HIL) frameworks are built in Simulink, the python-based error models are converted to Simulink structure at first so as to be integrated into CAE environments. The output of error model is added to the output of ideal sensor model, which then becomes the input of AEB function. The error models in Simulink format will be used for the further simulations and testings in function level validation.

4.2 Closed-loop Simulation in MIL Framework

In a closed-loop simulation, the outputs of system under test will be routed back to the the system's inputs and eventually influence the inputs. In our case, AEB function will start to intervene based on the perception of RadCam if emergency

happens. The outputs of AEB, e.g. the deceleration request, will then be propagated to the brake to control the motion of host vehicle, which in turn adjust the system's inputs such as host speed and relative speed between host and target object.

Our closed-loop simulation is the virtual imitation of real field test. Theoretically, it is expected to behave the same as the real field test so as to ease the burden on further excessive field tests. Therefore, it is rather important that the simulation platform comprises critical vehicle component models and renders surrounding traffic environment models.

A general overview of the simulation setup is illustrated in Figure 4.1. It is a combination of our system under test (ASM) and SPAS. We take advantage of SPAS directly because it satisfies perfectly our requirement specified above for a closed-loop simulation platform. SPAS includes the important vehicle models with high fidelity and meanwhile provides target object information. In order to validate the function performance in closed-loop simulation against real ones, it is assumed that the virtual vehicle components in SPAS are ideal, i.e. they own the same behaviour as the real ones.

In addition, AEB function software together with the error model are integrated into SPAS. Instead of simply using ideal sensor fusion, errors are added to the ideal sensor output with an intention of representing a more realistic sensor output. Moreover, we ensure that the same AEB software as the field test is integrated in the closed-loop simulation since different versions of software can yield to discrepancy as well.

As can be seen from Figure 4.1, AEB function produces the control signal, i.e. deceleration request, based on sensor perception of the surrounding environment and host states. Once brake receives the deceleration request, it successively lowers the host speed, which is one of the most important inputs of AEB. Although host states reported by other vehicle components affect function performance too, brake modular is the most influential one and therefore is shown separately. ASM and SPAS constitute our closed-loop simulation environment. The purpose of closed-loop simulation is to validate the applicability of error models in the SPAS tool chain.

4.3 Open-loop Simulation in MIL Framework

In a open-loop simulation, the inputs of system under test are no longer dependent on the system's outputs. There is no feedback from the outputs to the inputs. Contrast to the closed-loop simulation that includes all important components of a vehicle, open-loop simulation extracts relevant standalone models in order to eliminate possible discrepancies introduced by other models in the simulation since these models cannot represent the exact same behaviour as the real ones.

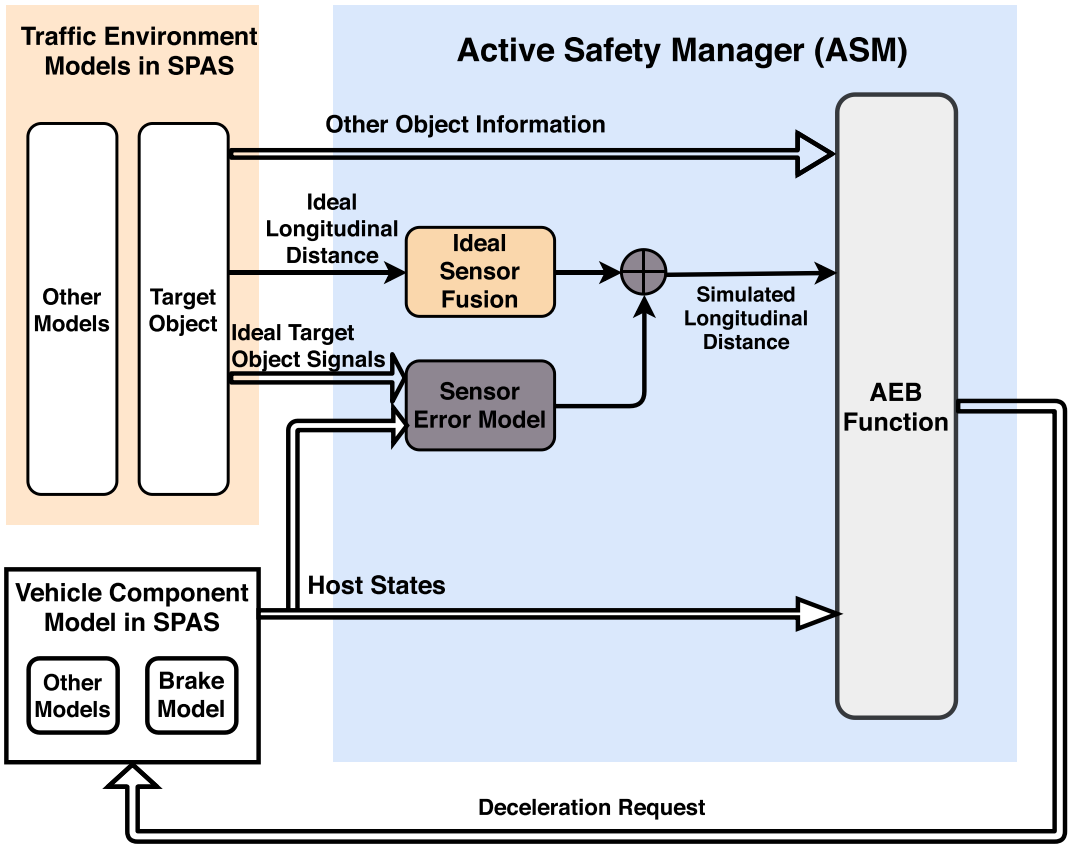

Figure 4.1: Overview of closed-loop simulation in MIL Framework

The general overview of open loop simulation with non-ideal sensor model is illustrated in Figure 4.2. In our case, open-loop simulation consists of only sensor model and AEB function software. Models such as brake and traffic environment model are excluded from this simulation. The required information from these models will be directly extracted and fed from real log. Only the longitudinal distance is the simulated one. Function performance will heavily depends on sensor model accuracy. Output signal of AEB, i.e. the control signal for brake, will be logged for later comparison.

Open-loop simulation is a vital part of the evaluation. By conducting this simulation, we are able to evaluate the AEB function performance while eliminating possible discrepancies introduced by other models in the closed-loop simulation.

4.4 Closed-Loop Simulation in HIL Framework

Simulation in MIL framework does not take the real time behaviour into account. In order to increase the confidence of our simulation, we propose to perform the simulation in HIL framework as well. The structure of this setup is shown in Figure 4.3.

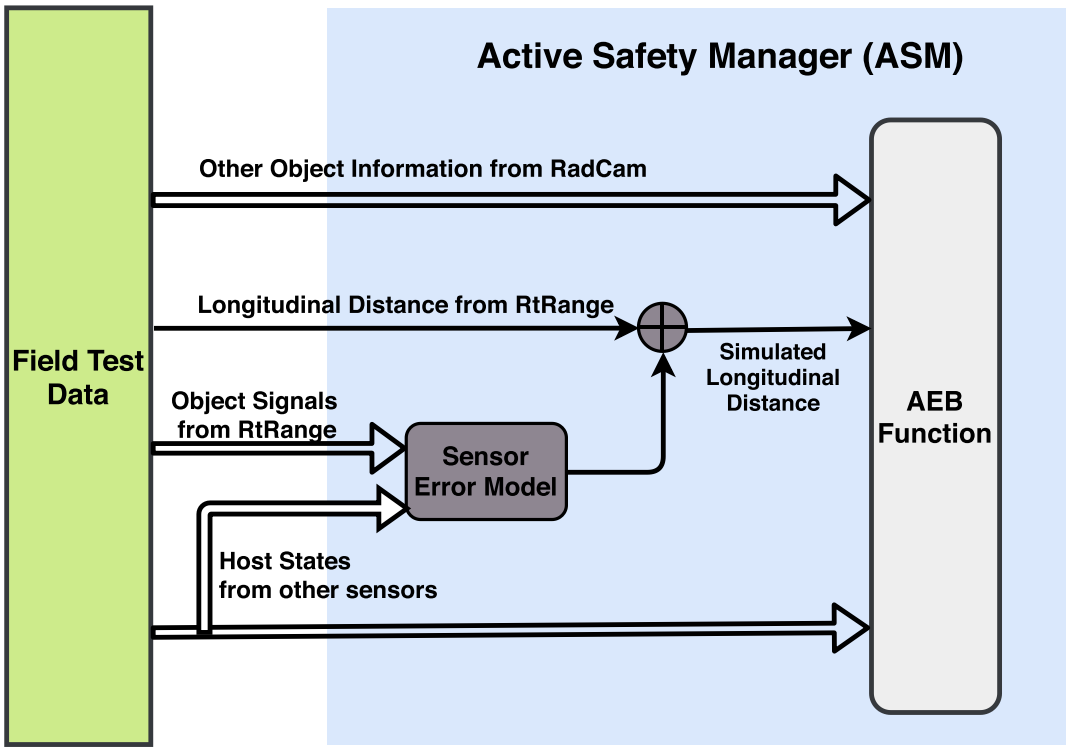

Figure 4.2: Overview of open loop simulation in MIL Framework

ASM software in Simulink environment is now replaced with the physical ASM ECU. Instead of virtual bus, physical bus with standard communication protocol is now used to send the traffic environment information and host vehicle state information to ASM.

After evaluating the current traffic situation, ASM outputs the control signal for the brake, which will later influence the host vehicle states. Among all these states, host speed is the most important one for AEB function. Since braking component is a crucial part for the control of host speed, physical brake is deployed in the HIL test system so that we can reduce the discrepancies between simulation and field test significantly. Other vehicle models in Simulink are now built into executable codes and loaded to the dSPACE real-time simulator.

For the traffic environment part, the target object list information except for longitudinal distance, such as object speed and acceleration, is extracted from the real world RadCam data and directly injected into the ASM via the physical bus. Longitudinal distance is still the sum of RtRange one and the simulated error. More details related with the injection solution will be explained in subsection 4.4.2. ASM will take over after the object information is injected. Deceleration request will be recorded for later comparison.

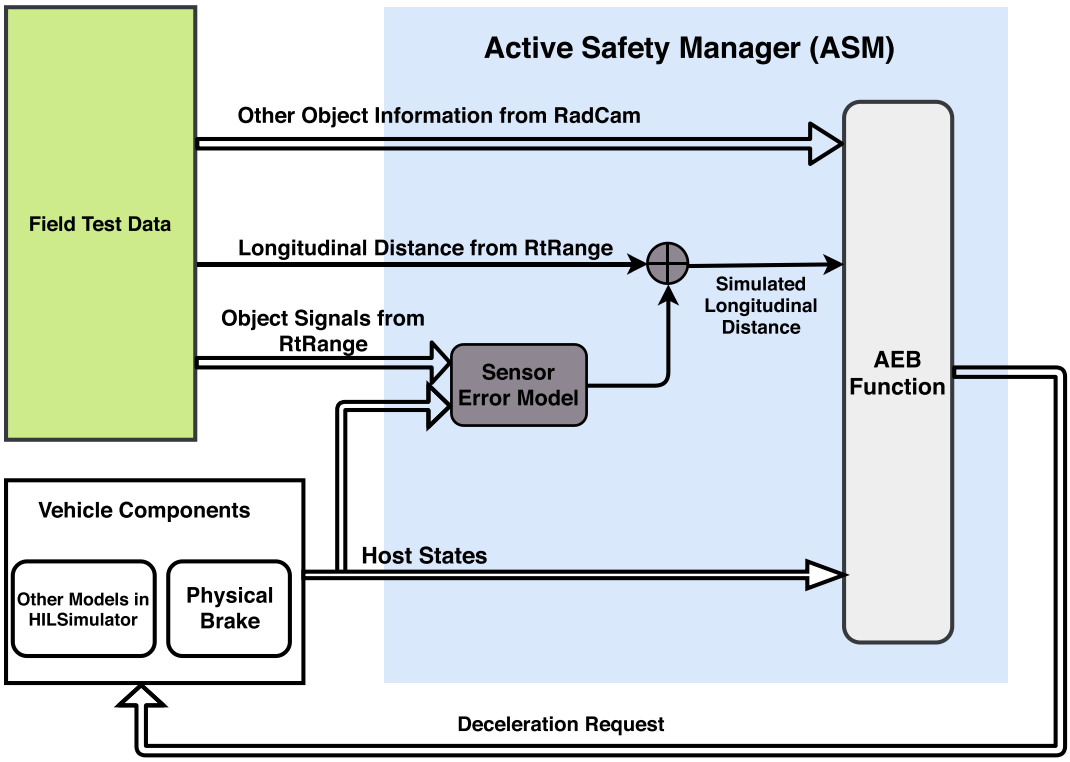

Figure 4.3: Structure of HIL simulation setup

4.4.1 Controller Area Network (CAN) and CANoe

Before we explained more details about the simulation setup, the physical bus protocol and the tool to implement this protocol in the HIL simulation environment are briefly introduced.

If two or several microprocessors are able to communicate with each other successfully, an unified message format and how data will be transmitted through devices must be defined by a standard protocol. The most common protocol used for automotive applications and industry is CAN. CAN typically uses a two-wire bus to guarantee the real-time communication between a group of electronic nodes. It is a multi-master serial protocol, which means any node can transmit a message on the CAN bus when the bus is free, and also all other nodes could receive the message. The message transmission and error handling will be handled by CAN controller automatically. The message transmitted on a CAN bus may contain multiple bytes of data, which is a significant advantage compared to other serial protocols including Universal Asynchronous Receiver-Transmitter (UART) [16] and Serial Peripheral Interface (SPI) [17] that can only transfer single byte of data every time. An identifier is also included in a message to determine the destination on the CAN bus.

CANoe is a user-friendly development, measurement and testing environment for a CAN-based network or system. This software can interconnect multiple nodes and perform communication transmission based on accurate timestamp. CANoe can

provide testing and measurement of a real-time distributed system by setting the rest of system to virtual while the module under test to real. CANoe also allows user to define data variables and messages in a database [18].

Figure 4.4 shows an example of test environment setup. In this figure, a laptop and module under test are both connected to a CAN bus. Some of the CAN nodes are simulated by CANoe and one real module is under test in this system.

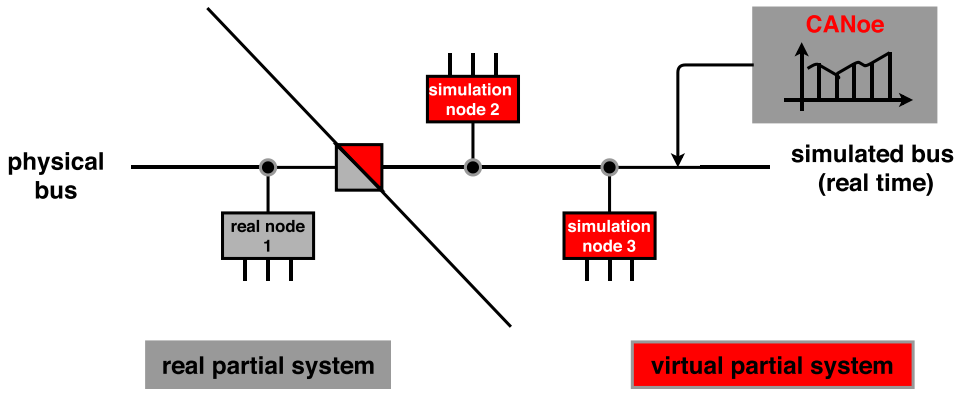

Figure 4.4: An example of CANoe testing environment

Typically the behaviour of a node in CANoe is designed with a C-based programming language: Communication Access Programming Language (CAPL). However, the node modelling capability of CANoe is strengthened by adding the Simulink environment, which shortens the development cycle. Users can use CANoe together with Simulink both in online and offline modes. In offline mode, simulation can be run in Simulink with the communication bus provided by CANoe. In online mode, one can simulate the network environment to test and validate a real hardware.

4.4.2 Injection Solution

One of the most critical setups of HIL test boils down to how to inject target object information to the ECU. The communication between computer and ASM ECU is on a standard CAN bus. Given the advantage of CANoe/Simulink interface mentioned earlier, we modeled a virtual node in Simulink that transmits a sequence of target object information to the ECU with the same frequency as the real ones.

In order to send the object information directly to the ECU, we have to first bypass the sensor fusion part, i.e. activate the injection mode. A user defined command is first sent to start this mode so that the real fusion data is discarded. After entering the injection mode, the main objective of this virtual node would be to transmit fused objects information with a period of 25 ms.

Among all the injected messages, there is a special one called 'timestamp', which is the timestamp used for sensor fusion. ASM also has its own timestamp. It is important to ensure the consistency between these two timestamps, otherwise ASM will

not makes safety-critical interventions. Our solution is to read the timestamp from ASM first with a higher frequency than the transmission task. Then the retrieved timestamp is wrote back to the CAN bus every 25 ms. Besides, we set the priority of the reading task higher than writing task. Every time before sending fusion timestamp, the simulated node retrieves it from the ECU so that we can ensure the consistency of this message.

4.5 System Responses for Comparison

It is important to understand the deceleration characteristics of a car for different traffic applications [19], especially for the scenarios considered in this thesis, CCRs scenarios. Since the performance of AEB function is highly influenced by the RadCam outputs, selecting proper system responses which can indicate the performance of AEB function would be quite important to evaluate the error model.

In our case, we first proposed the distance between host and target vehicle when the speed of host vehicle reduces to zero with brake. We termed this as braking distance. This system response can depict the braking performance and capability intuitively. Since the brake component is not involved in open-loop simulation, this system response will be only employed in closed-loop simulation.

The other indicator is deceleration request of AEB function. This system response is a time history signal which can indicate the whole braking process including the start and stop braking time as well as the maximum braking rate. Note that this response would be invested in both closed-loop and open-loop simulations.

4.6 Comparison Metrics

Depending on whether the system response is a single value or it is a time history, we proposed different metrics for comparing real system response and simulated system response in this section.

4.6.1 Box plot for single value comparison

Box plot is an intuitive way graphically illustrating the distribution of several groups of numerical data using quartile. It earns this name due to the box in the middle of a plot. An example of box plot can be seen from Figure 5.5. The middle line inside the box is the median, which is also called the second quartile. The lower end of the box(first quartile) represents that 25% of the data lie below this line. The third quartile is the upper bound of the box where 25% of the data locates above. Therefore, the box includes intermediate 50% of the entire data. The distance between third quartile and first qurtile is called Interquartile Range (IQR). Extending from

the box bounds, there are another two lines (whiskers) implying the data variability outside of the box. The upper/lower whiskers are 1.5 times of IQR away from the box upper/lower bound, respectively. If any data lies outside of the area between two whiskers, they will be regarded as outliers.

4.6.2 ISO Objective Rating Metric for time histories comparison

The ISO Objective Rating metric [20] is usually used to measure the discrepancy between time histories. It is based on ISO/TR 16250:2013 [21], which gives the essential calculations of relationship between two time series and is verified under different vehicle safety scenarios. This approach makes combination of two different metrics together to assess the correlation of two time histories more trustingly and reliably. The two metrics are Correlation and Analysis (CORA) method and Enhanced Error Assessment of Response Time Histories (EEARTH) method. The structure of ISO metric is shown in Figure 4.5. The corridor method is applied in CORA metric, to calculates the deviation between two curves. The EEARTH metric is coming from Error Assessment of Response Time Histories method [22], which divides difference between two curves into three independent error measurements: phase, magnitude and slope error. The detailed explanation of the metrics and methods mentioned above will be shown in following subsections.

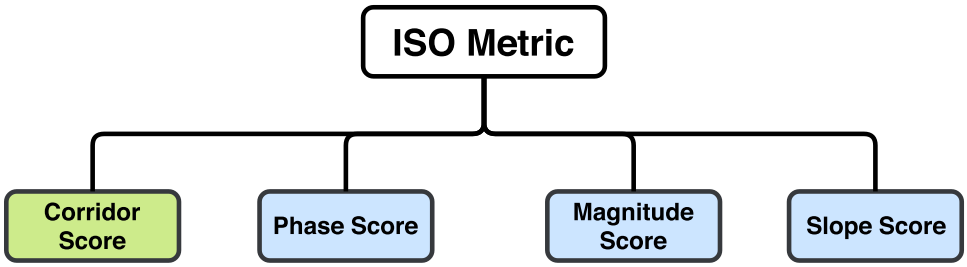

Figure 4.5: The structure of ISO metric

4.6.2.1 Overall ISO Rating

The overall ISO rating R is the linear combination of the four metrics:corridor score Z, phase score E_P, magnitude score E_M and slope score E_S, which can be seen in Formula 4.1. The description of the four weighting factors are in Table 4.1.

$$R = w_Z \cdot Z + w_P \cdot E_P + w_M \cdot E_M + w_S \cdot E_S, \tag{4.1}$$

where $w_Z + w_P + w_M + w_S = 1$.

Table 4.1: The weighting factors in overall ISO rating

	Value	**Description**
w_Z	0.4	The weighting factors of the corridor score
w_P	0.2	The weighting factors of the phase score
w_M	0.2	The weighting factors of the magnitude score
w_S	0.2	The weighting factors of the slope score

4.6.2.2 CORA Method

The basic idea of CORA method [23] is to define two corridors around reference signal R: inner corridor and outer corridor. If the simulated signal S locates within the inner corridor, the score will be 1. On the contrary, if the simulated signal is at the outside of outer corridor, the score will be 0. For the signal locating between the inner and outer corridors, the score will be in the range of 0 to 1. The corridors are calculated at each sample time, so the overall corridor score is the average of all corridor scores over time. The calculation of corridor score is shown below:

The absolute half width of inner corridor and outer corridor are defined in Formula 4.2 and 4.3:

$$\delta_i = a_0 \cdot R_{norm} \qquad 0 \leq a_0 \leq 1, \tag{4.2}$$

$$\delta_o = b_0 \cdot R_{norm} \qquad 0 \leq b_0 \leq 1 \;\; and \;\; a_0 < b_0, \tag{4.3}$$

where a_0 and b_0 are parameters, which represent weighting factors of half width of the inner and outer corridors, and R_{norm} is defined as $max(|min(R)|, |max(R)|)$. The value of a_0 and b_0 is shown in Table 4.2.

The calculation of the corresponding lower and upper bounds of inner and outer corridors are shown in Formula 4.4 and 4.5, separately.

$$\delta_i(t) = R(t) \pm \delta_i, \tag{4.4}$$

$$\delta_o(t) = R(t) \pm \delta_o, \tag{4.5}$$

Based on the range of inner and outer corridor, the corridor score $Z(t)$ is calculated at each sample time, which can be seen in Formula 4.6.

$$Z(t) = \begin{cases} 1, & \text{if} \quad |R(t) - S(t)| < \delta_i. \\ (\frac{\delta_o - |R(t) - S(t)|}{\delta_o - \delta_i})^{k_Z}, & k_Z \in N_{>0}, \\ 0, & \text{if} \quad |R(t) - S(t)| > \delta_o. \end{cases}, \tag{4.6}$$

where k_Z is penalty of the CORA score. The value of k_Z in ISO metric is shown in Table 4.2.

Hence, the overall corridor score over time is the average of all corridor scores at each evaluation time, which is shown in Formula 4.7

$$Z = \frac{\sum Z(t)}{N}, \tag{4.7}$$

Table 4.2: The weighting factors in CORA corridor score

	Value	**Description**
a_0	0.05	The weighting factors of the half width of inner corridor
b_0	0.5	The weighting factors of the half width of outer corridor
k_Z	2	The penalty of the CORA score

4.6.2.3 Enhanced Error Assessment of Response Time Histories (EEARTH)

The EEARTH metric [24] can quantitatively measure the discrepancy between the two time histories. It indicates the difference between two time series data based on three errors: phase error, magnitude error and slope error [22]. However, it is considerable tricky to separate these three errors completely. A method called Dynamic Time Warping [25] is employed to separate the reciprocity among those three errors. The EEARTH metric gives a final rating score by merging three error scores into one since those three errors mentioned above are in the different range and it can not thoroughly display the inconsistency between two signals only based on any single error. The detailed definitions of three error scores are shown in below:

EEARTH Phase Score:

The EEARTH phase error n_ε [22] is introduced to measure the phase or time lag between the two time series. This method is based on the cross-correlation coefficient. The basic idea is to shift one of the time history signals to reach the maximum cross-correlation coefficient. This shift that leads to the maximum value is the phase error n_ε.

The phase error score E_P then is calculated based on phase error n_ε obtained above. The mathematical calculation is shown in Formula 4.8.

$$E_P = \begin{cases} 1, & \text{if} \quad n_\varepsilon = 0. \\ \frac{\varepsilon^* \cdot N - n_\varepsilon}{\varepsilon^* \cdot N}, & \\ 0, & \text{if} \quad n_\varepsilon \geq \varepsilon^* \cdot N. \end{cases}, \tag{4.8}$$

ε^* is the maximum percentage of time shift that is allowed, which we set it to 50%. N is the number of sample size.

EEARTH Magnitude Score:

Magnitude Score [22] is used to measure the difference in magnitude between two time series. Before calculating the magnitude error, the time lag and slope error must be removed. Firstly, simulated and real signals are shifted and truncated

to minimize the phase lag. For the purpose of removing slope error, the shifted and truncated signals are warped by performing Dynamic Time Warping method. Hence, the magnitude error ε_{mag} is calculated in Formula 4.9:

$$\varepsilon_{mag} = \frac{\parallel S^{ts+w} - R^{ts+w} \parallel_1}{\parallel R^{ts+w} \parallel_1}, \tag{4.9}$$

where S^{ts+w} is the truncated and warped simulated log, and R^{ts+w} is the truncated and warped real log.

Formula 4.10 is used to calculate the EEARTH Magnitude Score E_M, which has the similar structure as the Formula 4.8.

$$E_M = \begin{cases} 1, & \text{if} \quad \varepsilon_{mag} = 0. \\ \frac{\varepsilon^*_M - \varepsilon_{mag}}{\varepsilon^*_M}, & \\ 0, & \text{if} \quad \varepsilon_{mag} \geq \varepsilon^*_M. \end{cases}, \tag{4.10}$$

where ε^*_M is the maximum of magnitude error that can be acceptable.

EEARTH Slope Score:

The slope score [22] is used to measure the discrepancy in slope between two time series. The slope in this metric indicates the slope of each point in the time history. In order to minimize the influence of phase lag, the shifted and truncated time histories are obtained. Furthermore, the slopes of each point in time history are calculated and DTW method is performed to minimized the effect of magnitude error afterwards. Formula 4.11 is used to calculate the slope error.

$$\varepsilon_{slope} = \frac{\parallel S^{ts+s+w} - R^{ts+s+w} \parallel_1}{\parallel R^{ts+s+w} \parallel_1}, \tag{4.11}$$

where S^{ts+s+w} is the shifted, truncated and warped simulated slope curve, and R^{ts+s+w} is the shifted, truncated and warped real slope curve.

The slope score is calculated by Formula 4.12

$$E_S = \begin{cases} 1, & \text{if} \quad \varepsilon_{slope} = 0. \\ \frac{\varepsilon^*_S - \varepsilon_{slope}}{\varepsilon^*_S}, & \\ 0, & \text{if} \quad \varepsilon_{slope} \geq \varepsilon^*_S. \end{cases}, \tag{4.12}$$

where ε^*_S is the maximum of slope error that can be acceptable.

This section explains the metrics for the comparison of system response. An intuitive way called box plot is used for the braking distance comparison. Regarding the deceleration request comparison, a complex ISO metric is introduced. By employing these metrics, we are able to evaluate the model applicability for different simulation setup in the function level validation.

5
Results

In this chapter, we present the results from both sensor and function level validations. For the sensor level, we will compare the simulated errors of all three models with real logs by using the metrics we introduced in section 3.4. For the function level, we will compare the simulated braking distance and deceleration request signals against the ground truth. Simulated function performance using both ideal and non-ideal sensor models will be compared with real function performance. Thus the research question "Do the error models improve the fidelity of the simulation environment compared to ideal sensor model?" will be answered in this chapter.

5.1 Sensor Level Validation

In this section, the comparison results between simulated and real longitudinal distance error for all three models will be shown and discussed.

In Figure 5.1 and Figure 5.2, the averaged RMSE (see Equation 3.1) and coefficient of correlation (see Equation 3.2) obtained from 400 simulation runs of CCRs scenarios are shown, respectively. In order to evaluate the error model's performance in different CCRs scenarios, we plot the values of metrics against the initial host speed, which range from 10 to 50 km/h with 10 km/h incremental step.

The mean of RMSE of 400 runs and the corresponding 95% confidence interval for each model are shown in Figure 5.1. Among all three models, Regression model achieves the best performance in terms of this metric since it results in the smallest RMSE value for most of the CCRs scenarios, which means the simulated data from Regression model has the smallest average difference to the real data. While GP model is the worst considering that it results in the highest RMSE value for most scenarios. Note that when host speed is 10 km/h, the value of RMSE is smaller than those of other scenarios, which applies for all three models.

Sensor error model performance in terms of the coefficient of correlation can be seen in Figure 5.2. The simulated longitudinal distance error from AIOHMM model has the strongest positive linear relationship with the real longitudinal position error. In contrast, GP model is the worst with even negative linear relationship. The Regression model has a weak positive relationship that the maximum coefficient of correlation value of all scenarios is still smaller than 0.5. Overall, AIOHMM model can capture the trend of error over time dimension more accurately than other two

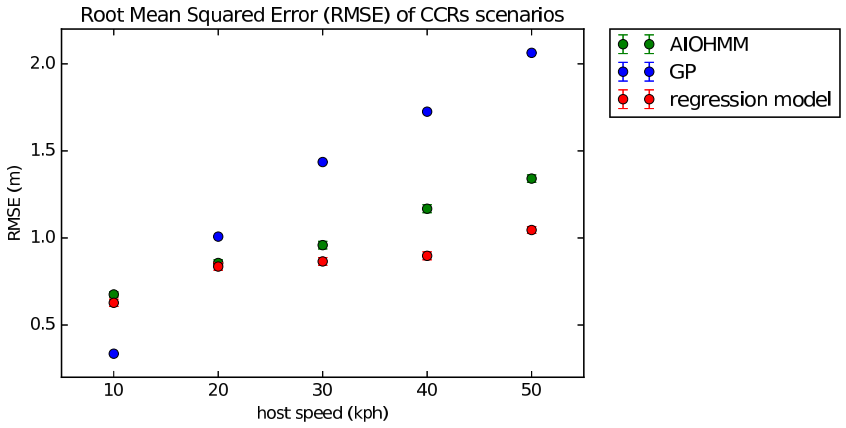

Figure 5.1: Root Mean Squared Error (RMSE) of CCRs scenarios for all three models

models, while GP model has the worst capability.

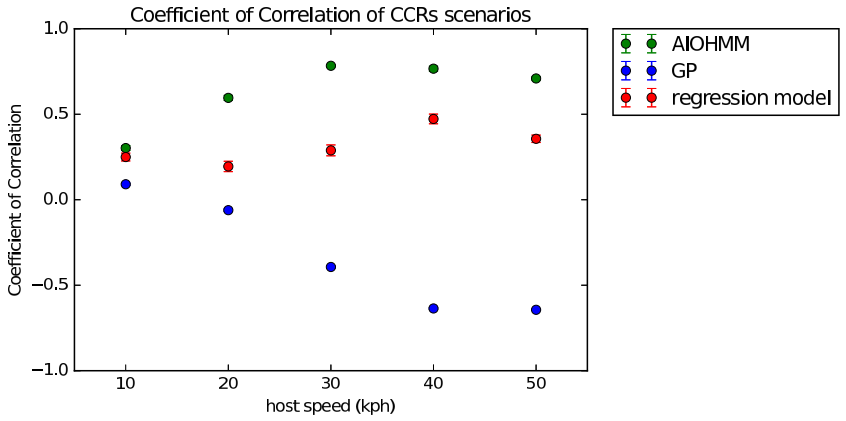

Figure 5.2: Coefficient of correlation of CCRs scenarios for all three models

In Figure 5.3, the probability distribution of all three models together with the probability distribution of real logs are plotted. We can observe that the histogram of Regression model is quite similar to the real logs. This can also be confirmed in Figure 5.4. In Figure 5.4, the Jensen-shannon distances (see Equation 3.4) of Regression model for most scenarios are much lower, which means that the probability distribution of Regression model is closer to the real distribution than other two models. Histograms of all the models for each CCRs scenario are shown in Figure A.1 to Figure A.5 in Appendix. Similar to performance evaluation using other metrics, GP model is the worst since it has the most different histogram and largest Jensen-shannon distance. When it comes to time series data, there is a drawback of using these two metrics since the time dimension is not considered during the evaluation, which means we still can get the same Jensen-shannon value and distribution even if the data are re-ranged.

However, it is important to mention that due to the time limitation, we only run the CCRs scenarios simulations, which are quite simple. What we can do as the future work is to simulate those three models under more complex scenarios, i.e. cut in or take over, to check whether the simulation results remain the same.

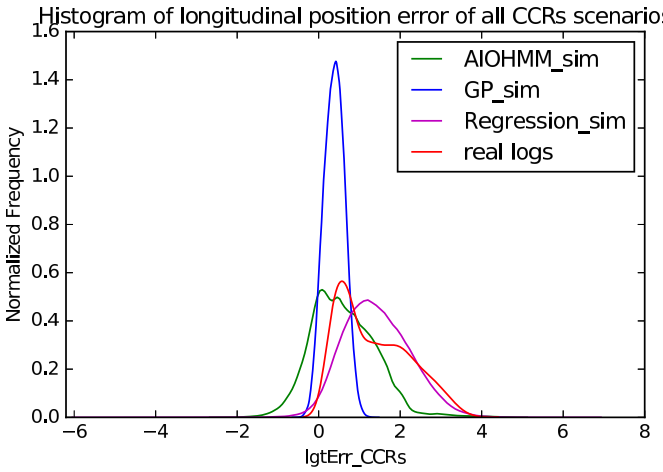

Figure 5.3: Histogram of longitudinal position error of CCRs scenarios for all three models and corresponding real logs

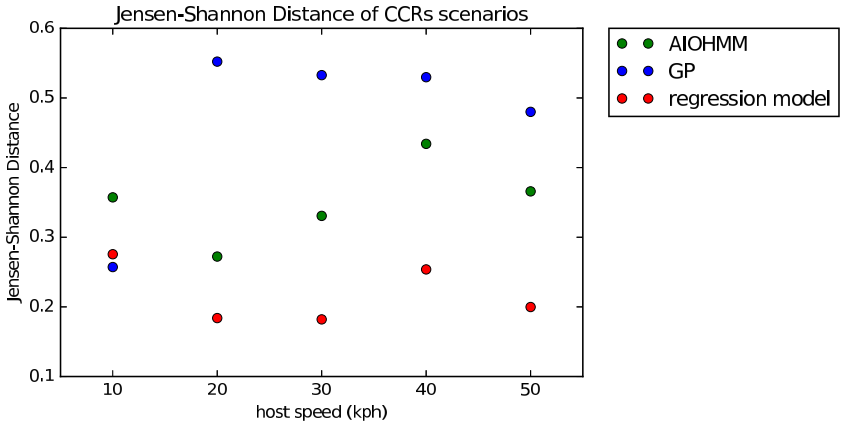

Figure 5.4: Jensen-Shannon Distance of CCRs scenarios for all three models

To conclude, this section presents the results of performance evaluation of sensor error models in sensor level. Simulated outputs from standalone error models are compared with true sensor errors. By applying different metrics, we are able to visualize and quantify the discrepancies between simulated data and real data. The smaller the discrepancy, the better model in terms of its accuracy. Basically, Regression model achieves the best performance when it comes to RMSE, histogram and Jensen-Shannon distance. However, if we consider the model fitness in time

dimension, AIOHMM model is better than Regression model. And among all three models, GP model is the worst in terms of all the metrics.

5.2 Function Level Validation

In this section, we will discuss about the simulation results in function level in different types of simulations. Depending on whether the system response is a single value or time history, various metrics will be employed as descried in Section 4.6.

5.2.1 Closed-loop Simulation Results in MIL Framework

Simulated braking distances using either non-ideal sensor models or ideal sensor model are compared with the ones recorded in the real host log. Since the ideal sensor model is deterministic, simulations of the same scenario with ideal sensor model will result in the same braking distance, which means its variance is zero. Since the test dataset for each scenario includes only single run, we cannot assess the uncertainties of the braking distance of each scenario in the test data. However, according to the engineers who collected the test dataset, the reason they only recorded one dataset for each scenario is that the variance of braking distance for each scenario is very small, which can almost be regarded as zero. Therefore, it is important that we care about the braking distance variance using non-ideal sensor models due to the randomness introduced by statistical error models.

Simulated braking distance against CCRs scenarios with GP model can be analyzed from Figure 5.5. For scenarios of 20/30/40 km/h, it can be seen that the medians of braking distance using GP model are closer to the real ones compared to the ones using ideal sensor model. However, even though GP model achieves more realistic braking distance for part of scenarios in terms of its median values, the corresponding variances are relatively large since the variances in test dataset are zero. The '+' indicates that there are even simulations results that can be regarded as outliers due to the high variance. Besides, we even get some negative braking distance values representing the vehicle collisions in the simulation, which never happen during the real field tests. Therefore, we can conclude that GP model can not be applied in current SPAS tool chain due to the high variance of the braking distance it has brought.

The box plot of simulated braking distance using AIOHMM model are illustrated in Figure 5.6. For all the scenarios, real braking distances are outside of the box of simulated ones. This means real braking distance is not within 50% middle simulated data. For scenarios of 30/40 km/h, real ones are even above the upper whisker. Moreover, the ideal sensor model yields to better simulation accuracy for all test scenarios, which can be visually inspected from the gap between real ones and simulated ones. Similar to the simulations using GP model, AIOHMM model results in high variance of braking distance as well. It does not increase the fidelity of simulation in the MIL framework.

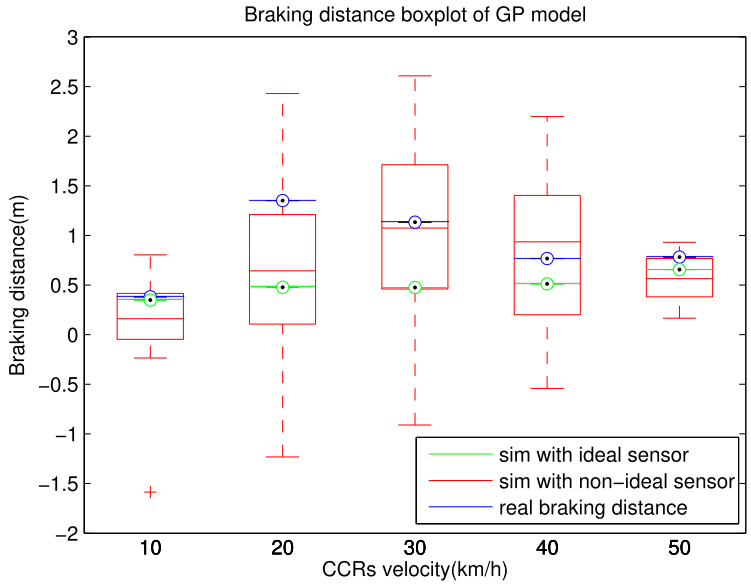

Figure 5.5: Braking distance boxplot of GP model

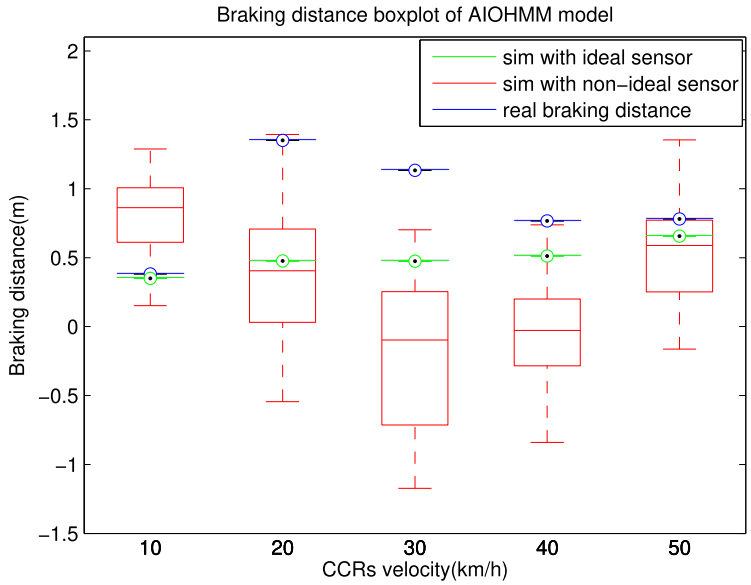

Figure 5.6: Braking distance boxplot of AIOHMM model

For the Regression model, simulated braking distances are all negative values, which means the host car and target car collides for all test scenarios and runs. This indicates that the Regression Model can not capture the behavior of the sensor error properly, even make the accuracy worse than the one using only ideal sensor model.

5.2.2 Open-loop Simulation Results in MIL Framework

In the open-loop simulation without including the brake model, braking distance is no longer available. Thus the methods for the comparison of the single value system response are not applicable here any more. The system response of open-loop simulation we focus on becomes the deceleration request of the AEB function. Meanwhile, measures intended for comparing time histories data are used in the function level validation.

As introduced in section 4.6, a comprehensive metric that classifies and combines three meaningful error components is deployed. Times histories from the first time stamp that the braking is triggered to the time stamp that host vehicle stopped are extracted for the further calculation of the ISO scores. All scores for one scenario are averaged and the mean scores are list in Table 5.1.

Table 5.1: Mean of ISO scores with respect to each scenario

	10km/h	**20km/h**	**30km/h**	**40km/h**	**50km/h**
ideal sensor	0.8360	0.6230	0.7030	0.5220	0.5310
AIOHMM	0.8870	0.5887	0.7449	0.6655	0.6692
GP	0.3364	0.4553	0.7347	0.7732	0.9021
Regression	0.2184	0.4566	0.6198	0.7140	0.8644

The averaged ISO scores using AIOHMM model are generally higher than those applying ideal sensor. Except for 20 km/h scenario, the discrepancies between simulated data and measured data are reduced, which improve the confidence of simulation.

The simulations using Regression model achieve different ISO scores for each scenario. For the CCRs scenarios with low speed, the averaged scores are relatively low compared to the simulations with ideal sensor model. Adding the regression error model even reduces the accuracy of simulations. For example, the mean score for 10 km/h is only 0.2184 since 60% of total runs even fail to trigger the AEB function. It is the similar case for 20 km/h and 30 km/h that some runs achieve good agreement of deceleration request with the real one, while others achieve relatively small deceleration request. For the simulations of 40 km/h, though the mean score is better than the one using ideal sensor, there are still 20% of total runs yielding to very low scores due to the low generated deceleration request for these runs. Two examples of the simulated deceleration request against real one are shown in Figure 5.7, one with low ISO score and the other one with high score. However, the fidelity for simulation with 50 km/h is highly improved by adding the regression error model. The scores of all runs exceed the ideal score. To conclude the performance of regression model, it increases the simulation accuracy of the CCRs scenarios with high speed, while it functions badly for low speed scenarios.

The performance of the GP model is quite similar to the Regression model. ISO scores for the low speed scenarios are not as good as those of ideal model, while it

is the opposite case for the high speed scenarios.

Compared with the outcome derived from the closed-loop simulation, there seems to be a conflict based on the system responses we select. Adding regression models in closed-loop simulations yields to collisions for all scenarios. But this model achieves high ISO grades for high speed scenarios. Similarly, AIOHMM model in the closed-loop simulation seems not as promising as in the open loop simulation. However, it makes sense for these models to result in different functions performance in various simulation environments. As we mentioned in Section 4.3, closed-loop simulations include all the other vehicles components, which may also introduce unknown influence for the final braking distance. They are more intended for the evaluation of the model applicability in the SPAS tool chain. Function level evaluation in open loop environment is more reliable to estimate the sensor error models themselves.

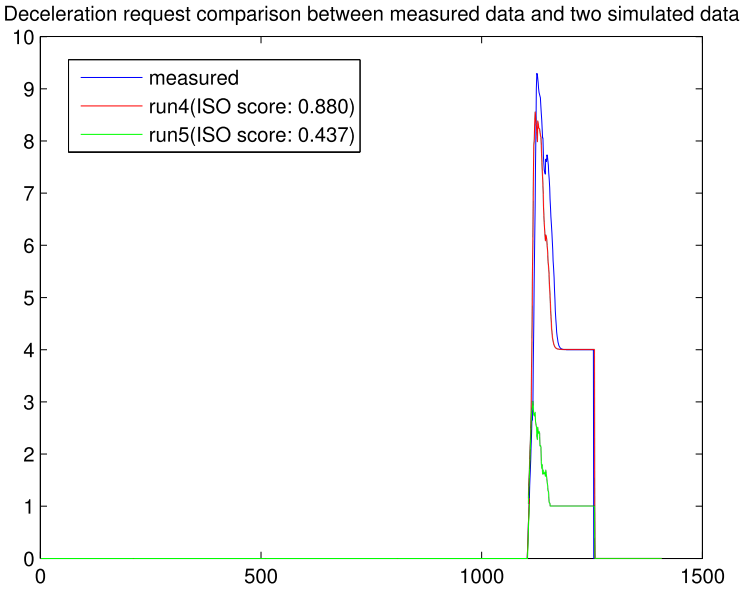

Figure 5.7: This figure illustrates the deceleration request of two simulated runs(40 km/h) using regression error model. Run4 agrees more with the measured data and results in a high score. Run5 is quite different from the measure data. Thus the score is low.

5.2.3 Closed-loop Simulation Results in HIL Framework

In our HIL closed-loop simulation, since the target object data is extracted from real log, host vehicle and target vehicle are not in the same coordinate system. Therefore, we cannot calculate the braking distance. Deceleration request is selected as the system response. After collecting simulation data in closed-loop HIL environment, function level evaluation of the sensor error models would be the same as in open-loop MIL environment. The discrepancies between simulated deceleration

request and measured ones are quantified by the same metrics as above. Due to the time limit, only the Regression model is deployed in the simulation and evaluated. However, the HIL simulation setup and the evaluation method can also be applied for the other two error models.

The regression error model exhibits the similar behaviour as in the MIL open loop simulation. When it comes to high speed scenarios, the ISO scores are higher than the ideal ones. For low speed scenarios, it does not achieve good function performance. Actually, the HIL closed-loop simulation setup is quite similar to the setup of MIL open loop simulation with only one physical brake added. Target object information other than longitudinal distance is both extracted from the real log. This explains why we get similar conclusions regarding the Regression model from the HIL closed-loop simulation and from the MIL open loop simulation.

Table 5.2: Overall ISO scores of the deceleration request in HIL framework

	10km/h	**20km/h**	**30km/h**	**40km/h**	**50km/h**
ideal sensor	0.7966	0.5529	0.7700	0.5533	0.5930
Regression	0.1540	0.2964	0.5766	0.8054	0.8377

6

Conclusion and Future Work

In this thesis we propose a method for the validation of sensor error models in sensor fusion level. We evaluate the standalone sensor error models by quantitatively measuring the discrepancy between model output and measured data based on several statistical comparison metrics. From the simulation results we obtained, distribution of the predicted data from Regression model are closer to the measured data in general. However, AIOHMM model can capture the characteristics of measured data over time dimension much more accurately than others. And GP model achieves the worst prediction performance.

Apart from the traditional sensor level validation that simply compares the measured data with the model output, we also suggest to evaluate the responses of the system where the sensor models are intended to be used, that is function level validation.

The simulation setup and tool chain for the function level validation are presented. Both advantages and disadvantages of different validation environments are explained. Depending on the type of validation setup, we select different system responses including braking distance and deceleration request. In terms of braking distance, we use box plot to visualize the distribution between different groups. For deceleration request comparison, ISO metric is used to quantify the discrepancies between two time histories. By using these metrics, we are able to evaluate the accuracy of the sensor error models in terms of their intended use. The method we proposed can also evaluate whether the sensor error models can improve the fidelity of system compared to ideal sensor model in both MIL and HIL environment .

There exist three limitations of current thesis work. First, only Euro NCAP CCRs scenarios are included in the function level test. Although these scenarios are the typical ones where the sensor is applied, these scenarios are still too 'one-dimension' since only the traffic situation with straight roads is considered. In order to get a better evaluation of the property of sensor error models, scenarios including roads with more curvature should be included in the function level test. Second, when set up the closed loop simulation in HIL environment, only the host vehicle states are fed to the ECU from virtual simulation environment, while we use the target object information from the real log. This is not like a complete closed-loop simulation where no real data is required. We can call our setup "partial closed-loop simulation" or "partial open-loop simulation". It would be better to set up a complete closed-loop simulation to evaluate the model applicability in our HIL

environment. Third, models are merely evaluated in terms of their accuracy in this thesis. However, other aspects of model evaluation are never mentioned here. For example, model complexity is also one of the important factors that should be considered when we determine whether we want to apply the model in our simulation tool chain. A complex model would require longer simulation time and higher computational demand. Evaluation of model complexity can be seen as future work.

Bibliography

[1] Delphi RACam : integrated radar and camera, 2016-01-28. http://www.systemplus.fr/reverse-costing-reports/delphi-integrated-radar-and-camera-system/.

[2] William L Oberkampf, Timothy G Trucano, and Charles Hirsch. Verification, validation, and predictive capability in computational engineering and physics. *Applied Mechanics Reviews*, 57(5):345–384, 2004.

[3] American society of mechanical engineers. *Council on Codes and Standards, Board of Performance Test Codes: Committee on Verification and Validation in Computational Solid Mechanics*, 2003.

[4] Arno Bergmann. Benefits and drawbacks of model-based design. *KMUTNB International Journal of Applied Science and Technology*, 7:15–19, 09 2014.

[5] ISO 26262:2012: Road vehicles — functional safety. *International Organization for Standardization, Geneva, Switzerland.*, 2012.

[6] Johan Florbäck. Anomaly detection in logged sensor data. Master's thesis, Chalmers University of Technology, 2015.

[7] Feature Selection, 2019-08-15. https://en.wikipedia.org/wiki/Feature_selection.

[8] Euro NCAP. European New Car Assessment Programme(Euro NCAP): Test Protocol-AEB Systems. Version 1.1, June 2015.

[9] RT-Range – an introduction to OxTS' comprehensive ADAS testing solution, 2016-08-15. https://www.oxts.com/technical-notes/rt-range-introduction/.

[10] Frank E. Ritter, Michael J. Schoelles, Karen S. Quigley, and Laura Cousino Klein. *Determining the Number of Simulation Runs: Treating Simulations as Theories by Not Sampling Their Behavior*, pages 97–116. Springer London, London, 2011.

[11] Tianfeng Chai and Roland R Draxler. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250, 2014.

[12] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[13] K. P. Burnham and D. R. Anderson. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. New York: Springer, 2ed edition ed., 2002.

[14] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

[15] Wolfgang Breitling and Centrex Consulting Corporationi. Histograms–myths and facts. In *Proc. Hotsos Symposium on Oracle System Performance*, 2005.

[16] Umakanta Nanda and Sushant Kumar Pattnaik. Universal asynchronous receiver and transmitter (uart). In *Advanced Computing and Communication Systems (ICACCS), 2016 3rd International Conference on*, volume 1, pages 1–5. IEEE, 2016.

[17] Frédéric Leens. An introduction to i 2 c and spi protocols. *IEEE Instrumentation & Measurement Magazine*, 12(1):8–13, 2009.

[18] Vector Informatik GmbH. User manual CANoe. Version 7.5, 2010.

[19] Akhilesh Kumar Maurya and Prashant Shridhar Bokare. Study of deceleration behaviour of different vehicle types. *International Journal for Traffic & Transport Engineering*, 2(3), 2012.

[20] ISO/TS 18571:2014: Road vehicles — objective rating metric for non-ambiguous signals. *International Organization for Standardization, Geneva, Switzerland.*, 2014.

[21] ISO/TR 16250:2013: Road vehicles — objective rating metrics for dynamic systems. *International Organization for Standardization, Geneva, Switzerland.*, 2013.

[22] H Sarin, Michael Kokkolaras, G Hulbert, Panos Papalambros, S Barbat, and R-J Yang. Comparing time histories for validation of simulation models: error measures and metrics. *Journal of dynamic systems, measurement, and control*, 132(6):061401, 2010.

[23] Christian Gehre, Heinrich Gades, and Philipp Wernicke. Objective rating of signals using test and simulation responses. In *Proceedings: International Technical Conference on the Enhanced Safety of Vehicles*, volume 2009. National Highway Traffic Safety Administration, 2009.

[24] Zhenfei Zhan, Yan Fu, and Ren-Jye Yang. Enhanced error assessment of response time histories (eearth) metric and calibration process. Technical report, SAE Technical Paper, 2011.

[25] Y Li, CL Wen, Z Xie, and XH Xu. Synchronization of batch trajectory based on multi-scale dynamic time warping. In *Machine Learning and Cybernetics, 2003 International Conference on*, volume 4, pages 2403–2408. IEEE, 2003.

A

Appendix 1

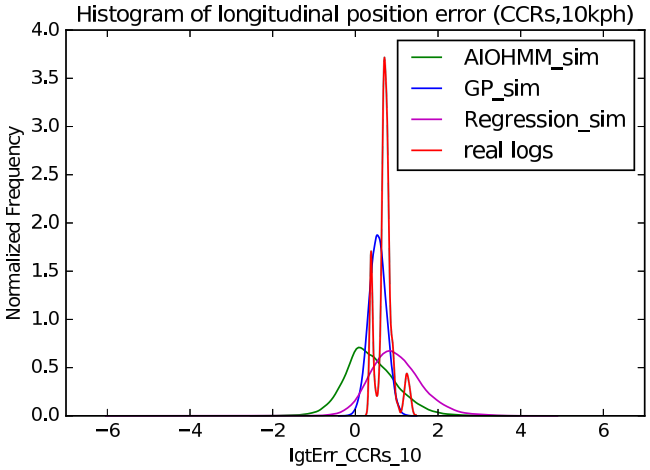

Figure A.1: Histogram of longitudinal position error for all three models and corresponding real logs when host speed is 10 km/h.

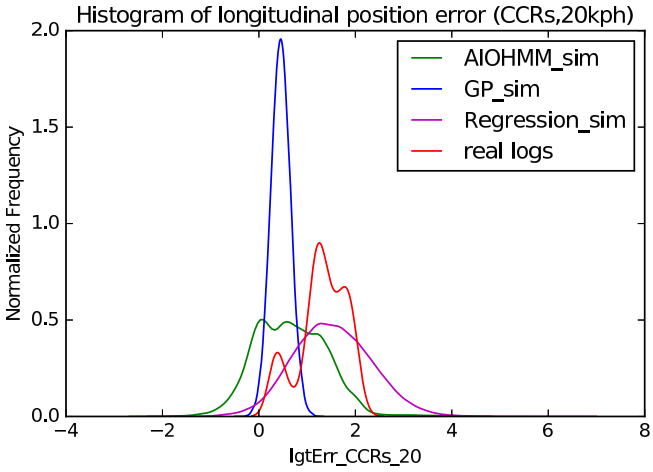

Figure A.2: Histogram of longitudinal position error for all three models and corresponding real logs when host speed is 20 km/h.

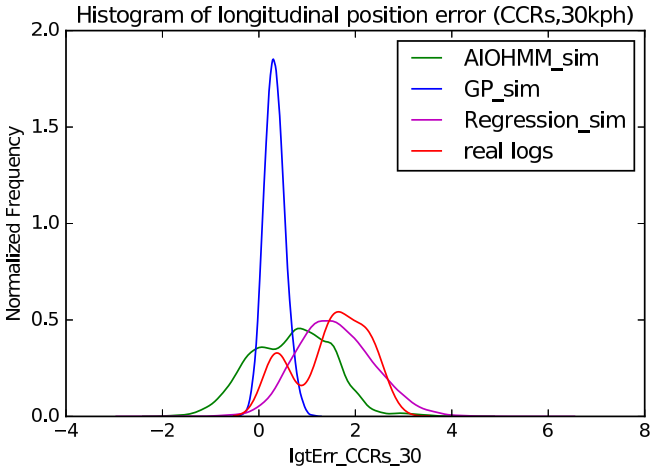

Figure A.3: Histogram of longitudinal position error for all three models and corresponding real logs when host speed is 30 km/h.

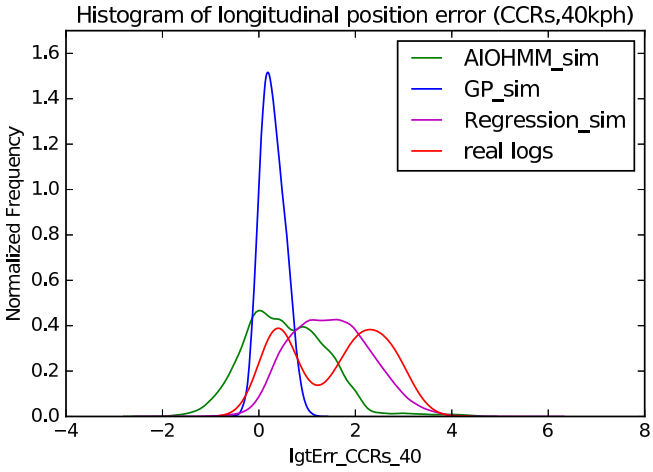

Figure A.4: Histogram of longitudinal position error for all three models and corresponding real logs when host speed is 40 km/h.

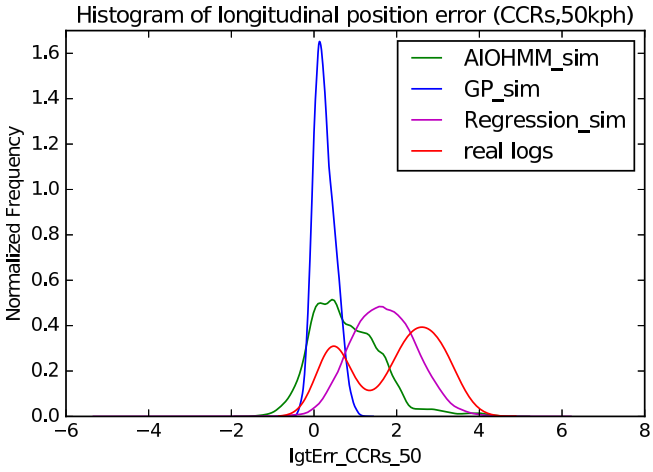

Figure A.5: Histogram of longitudinal position error for all three models and corresponding real logs when host speed is 50 km/h.