Development of an FMCW Lidar Signal Processing Model

Masterthesis No. 871/23
Editor: Kristof Hofrichter | 2530549
Supervisor: Lukas Elster M.Sc.

TECHNISCHE UNIVERSITÄT DARMSTADT

FZD FAHRZEUGTECHNIK TU DARMSTADT

Kristof Hofrichter
Matriculation No.: 2530549
Study program: Maschinenbau

Masterthesis No. 871/23
Topic: Development of an FMCW Lidar Signal Processing Model

Submitted: October 17, 2023

Technische Universität Darmstadt
Fachgebiet Fahrzeugtechnik
Prof. Dr.-Ing. Steven Peters
Otto-Berndt-Straße 2
64287 Darmstadt

Abstract

This work is concerned with the simulation of a frequency modulated continuous wave (FMCW) lidar (light detection and ranging) sensor in the context of automated vehicles. In order to save time and resources, automated driving functions are increasingly safeguarded in virtual environments, which requires respective simulation models of the vehicle's perception sensors. In the following, the simulation of the FMCW lidar sensor is split into the three components: environment simulation, signal propagation model and signal processing model. The latter represents the main focus of this work and includes all processing steps that are executed within the sensor housing. An externally provided ray tracer realizes the signal propagation model.

At the beginning, it is necessary to define the requirements for the FMCW lidar signal processing model. For this purpose, an existing approach is adapted and further developed. As a result, it is specified that the model must be able to accurately reproduce the beam pattern, the range measuring and the direct radial velocity measuring. Next up, a model development methodology is introduced which envisages an iterative step-by-step implementation of the three mentioned requirements with continuous verification and validation. Accordingly, within each iteration the current model is verified and validated after the implementation step.

Subsequently, the proposed model development methodology is utilized to realize the signal processing model. The beam pattern, the radial range measuring and the radial velocity measuring are implemented one after the other. For each of the three, experimental reference measurements are conducted to obtain real sensor data. After that, the signal processing model is used within the sensor simulation to re-simulate the real measurements in a virtual environment. To enable this, additional reference sensors such as a laser range finder are employed during the real measurements to precisely determine the range to the target. For the velocity reference measurements, an Automotive Dynamic Motion Analyzer (ADMA) captures the position and velocity of the moving target vehicle. The uncertainties of these reference sensors are also taken into account by the simulation. The resulting simulated data is directly compared to the real reference data to validate the model after each implementation. At the end, a partly validated model is presented that fulfills the basic functions of an FMCW lidar sensor.

The findings of this work show that the developed methodologies and approaches are suited for the development of a less complex perception sensor model. However, further improvements are necessary to enable more sophisticated and fully validated models.

Contents

List of Symbols and Indices

Latin formula symbols:

Symbol	**Unit**	**Description**
c	m/s	Speed of light
E/X	-	Monochromatic lightwave
f	Hz	Frequency
$\mathcal{F}$	-	Set of CDFs or EDFs, p-box
F	-	CDF/EDF
i	A	Electric current
P	W	Optical power
r	m	Radial range
$\dot{r}$	m/s	Radial velocity
R	A/W	Photodiode responsivity
t	s	Time
v	m/s	Velocity

Greek formula symbols:

Symbol	**Unit**	**Description**
κ	Hz/s	Slope of frequency ramp
λ	nm	Wave length
τ	s	Round-trip delay
ϕ	deg	Azimuth angle
ω	1/s	Angular frequency
γ_t	deg	Lidar target rotation angle
ζ	$[\zeta]$	Measurand
θ	deg	Elevation angle

Indices:

Symbol	**Description**
air	Air
bias	Model bias
c	Corrected
CAVM	Corrected area validation metric / Model scattering error
down	Down-ramp
desired	Desired value
DP	Detection point

Symbol	Description
Doppler	Doppler
line	Horizontal scan line
laser	Laser
LO	Local oscillator
nom	Nominal value
noise	Noise
P	Pixel
ref	Reference (Measured with reference device)
real	Real (Measured with real lidar sensor)
rt	Output by the raytracer
RX	Received
stdv	Standard deviation
sim	Simulated by the sensor model
up	Up-ramp

List of Abbreviations

ADAS	Advanced Driver Assistance Systems
ADMA	Automotive Dynamic Motion Analyzer
APSS	Active Perception Sensor System
AVM	Area Validation Metric
BSI	British Standards Institution
CAVM	Corrected Area Validation Metric
CDF	Cumulative Distribution Function
CEPRA	Cause, Effect, and Phenomenon Relevance Analysis
CMP	Credible Modeling Process
CSP	Credible Simulation Process
DDT	Dynamic Driving Task
DVM	Double Validation Metric
EDF	Emperical Cumulative Distribution Function
EOM	Electro-Optic Modulator
FMCW	Frequency Modulated Continuous Wave
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FN	False Negative
FoV	Field-of-View
FP	False Positive
FZD	Fahrzeugtechnik Darmstadt
GNSS	Global Navigation Satellite System
GT	Ground Truth
GUI	Graphical User Interface
ID	Identification Number
IMU	Inertial Measurement Unit
lidar	Light Detection and Ranging
LO	Local Oscillator
MEMS	Microelectromechanical
ODD	Operational Design Domain
OPA	Optical Phased Array
OSI	Open Simulation Interface
PerCollECT	Perception Sensor Collaborative Effect and Cause Tree
radar	Radio Detection and Ranging

ROS2	Robot Operating System 2
RX	Received
SiL	Software-in-the-Loop
SUT	System Under Test
TL	Tunable Laser
ToF	Time-of-Flight
TUDa	Technical University of Darmstadt
TX	Transmitted
ViL	Vehicle-in-the-Loop
XML	Extensible Markup Language

List of Figures

List of Tables

1 Introduction

Amongst a diverse range of others, improving the safety of public road traffic is one primary motivation for the push towards automated driving.[1] As human drivers are responsible for the majority of today's accidents[1,2], the automation systems aim to support or completely replace human drivers. With the release of their automated driving system Drive Pilot[3] in 2022, Mercedes-Benz has taken the next step as it is the first SAE-Level 3[4a] system that has an internationally valid system approval.[5] SAE-Level 3 implies, that under certain conditions the system is able to perform the dynamic driving task (DDT), which includes "all of the real-time operational and tactical functions required to operate a vehicle in on-road traffic"[4b] with a human driver, who must take over the DDT when requested to do so, as a fallback.[4c] The Drive Pilot function can be activated on certain parts of the German highway system at speeds up to 60 km/h in high traffic conditions, provided that, good weather conditions and daylight are present.[3] To ensure the safety of the automated driving system, Mercedes-Benz relies on a wide range of different sensors for the vehicles environment perception system with lidar (light detection and ranging), radar (radio detection and ranging) and camera being particularly important.[3] Environment perception is one of the primary tasks of any automated driving system as it provides the crucial information for subsequent stages like decision making and vehicle control.[6] Since radar, lidar, and camera each possess their own strengths and weaknesses, today's manufacturers usually rely on at least two of the three sensors for their (partly) automated vehicles[7] and for instance, utilize sensor fusion for an accurate, robust, and reliable obstacle detection.[8]

Automotive radars are small, lightweight, robust and allow a high detection range even in difficult environmental conditions.[9a] Besides distances, relative radial velocities of objects are directly measurable using the Doppler effect.[10] On the downside, a detailed mapping of the environment is impractical due to a lower resolution, which moreover leads to little information about the detected object.[9a] In contrast, cameras provide a detailed, high-resolution image of the environment while lacking performance in adverse weather conditions and distance measurement.[9a] The 3D imaging capabilities of lidars are characterized by good accuracy and resolution as well as a high detection range.[9a] The high resolution image of the environment further enables machine learning based object classification and recognition.[9b] Drawbacks of lidars are a deteriorated performance in adverse weather conditions[9b] along with interference effects caused by external light sources and sunlight.[10]

[1] Wachenfeld, W.; Winner, H.: The Release of Autonomous Vehicles (2016), p. 425.

[2] Thomas, P. et al.: Identifying the causes of road crashes in Europe (2013).

[3] Mercedes-Benz: DRIVE PILOT (2023).

[4] SAE International: SAE-J3016 (2021). a: -; b: p. 9; c: pp. 25-27

[5] Mercedes-Benz: Vorreiter bei automatisierten Fahr- und Sicherheitstechnologien (2022).

[6] Velasco-Hernandez, G. et al.: Autonomous Driving Architectures, Perception and Data Fusion (2020).

[7] Marti, E. et al.: A Review of Sensor Technologies for Perception in Automated Driving (2019), p. 104.

[8] Yeong, D. J. et al.: Sensor and Sensor Fusion Technology in Autonomous Vehicles (2021), p. 29.

[9] Mohammed, A. S. et al.: The Perception System of Intelligent Ground Vehicles (2020). a: p. 12; b: p. 14

[10] Roriz, R. et al.: Automotive LiDAR Technology (2022), pp. 6283-6285.

While these drawbacks mostly concern so called pulsed time-of-flight (ToF) lidars, frequency modulated continuous wave lidars are said to be more robust against harsh environmental conditions[11] and immune to interference from other light sources and sunlight.[12] In contrast to pulsed ToF lidars that modulate the intensity of the light, FMCW lidars modulate the frequency of the emitted signal in time and use an interferometric detection scheme to demodulate the received signal.[13] Additional advantages of FMCW lidars are the capability of direct velocity measurement of the target by using the Doppler effect, high depth accuracy and the lower optical peak power due to continuously emitting photons.[12] These properties make FMCW lidars highly interesting for a potential application in advanced driver assistance systems (ADAS) and other automated driving functions.

With regard to the aforementioned goal of increased safety through the introduction of automated vehicles, the question arises, as to how it can be proven that the automated vehicle is actually safer than a human driver. Theoretical considerations made by Wachenfeld and Winner[14] show that a couple of billion test kilometers are necessary to prove that an automated vehicle is twice as safe as human drivers in current vehicles. The necessary test distance further increases considering that the driving has to be repeated for each variation of the system, as is the case, for example, when new FMCW lidar sensors are introduced.[14] One way to reduce the test effort is replacing real elements of the vehicle or the environment with artificial or virtual ones. Software-in-the-loop (SiL) simulations, utilize vehicle dynamics, traffic flow and perception sensor system simulation besides many others to fully represent all aspects of the vehicle and its environment virtually. Whereas for vehicle-in-the-loop (ViL) simulations, the vehicle is real but the environment is artificial or virtual.[15] Both approaches have in common that perception sensor system models are required to close the loop and therefore these models are of great interest for the development of ADAS and automated vehicles.

The work at hand contributes to the simulation approach by providing an FMCW lidar signal processing model which makes it possible to virtually test and evaluate the performance of FMCW lidars in the context of ADAS and automated driving.

1.1 Ascertainment of Assignment

This work is concerned with the development of an FMCW lidar signal processing model. After a literature research on the fundamentals of FMCW lidars and perception sensor modeling, the first step is to define the requirements and identify the relevant system parameters that are necessary for modeling an FMCW lidar signal processing model. Since the aspired model is based on a real FMCW lidar sensor, specific values are assigned to the previously identified parameters. For this purpose, values are taken from data sheets and operating instructions, on the one hand, and determined experimentally, on the other. For all reference measurements performed within the scope of this work, the uncertainties of the reference sensors must also be taken into account. Subsequently, the obtained requirements and system parameters are used to adjust a pulsed ToF lidar signal processing model to the FMCW principle. The developed

[11] Lee, J. et al.: Frequency Modulation Control of an FMCW LiDAR (2023), p. 1.

[12] Rogers, C. et al.: A universal 3D imaging sensor on a silicon photonics platform (2021), p. 257.

[13] Behroozpour, B. et al.: Lidar System Architectures and Circuits (2017), p. 139.

[14] Wachenfeld, W.; Winner, H.: The Release of Autonomous Vehicles (2016), pp. 442-446.

[15] Amersbach, C. T.: Diss., Functional Decomposition (2020), p. 20.

model is a phenomenological model which means that it consists of physical and stochastic components. The primary objective is the development of the signal processing model, whereas, the signal propagation is modeled with the help of a ray tracer provided by Persival GmbH.

Finally, the reference measurements are utilized to validate the developed FMCW lidar model on an exemplary and sample basis. To this effect, selected experimental setups are virtually re-simulated using the model and the results are compared with the real world reference measurements afterwards.

1.2 Methodology and Structure

In Chapter 2 the essential fundamentals are explained. After a brief definition of terms, the basics of an FMCW lidar and ray tracing are introduced. Subsequently, the state of the art on perception sensor simulation is briefly addressed and the validation metric used in this work is clarified.

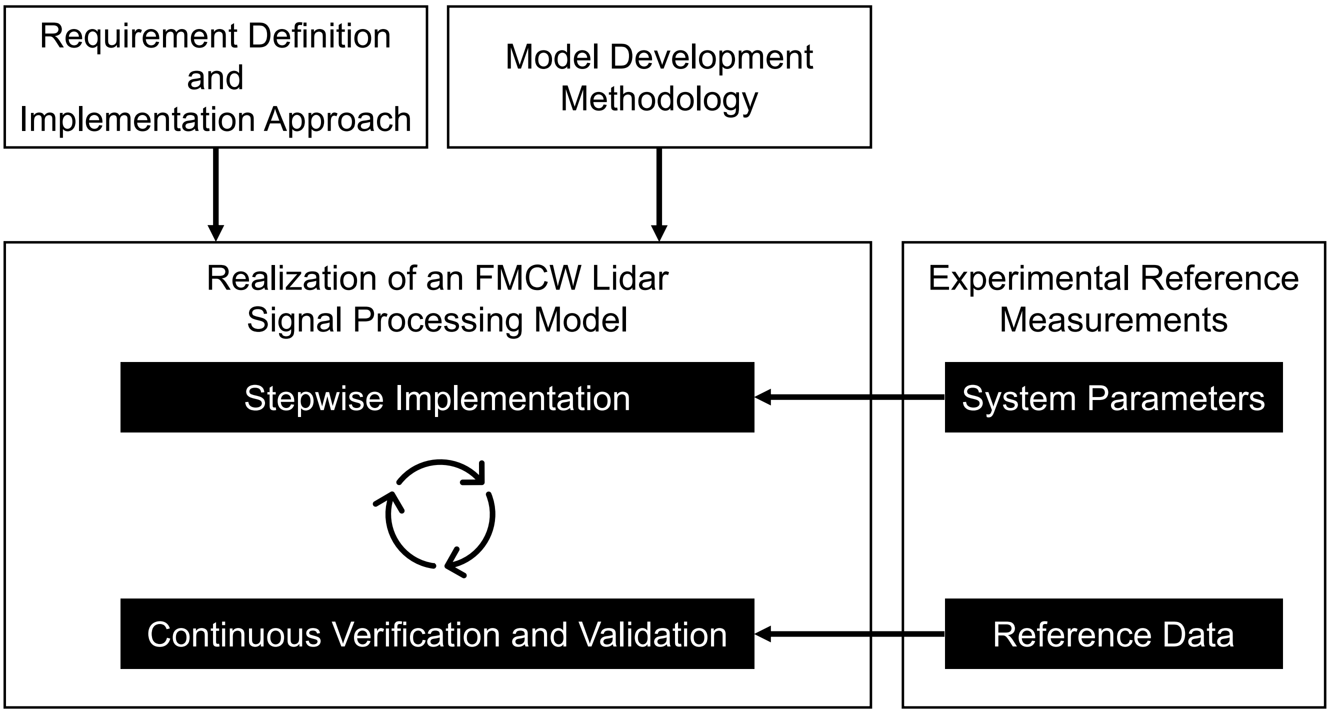

Requirement Definition and Implementation Approach
Model Development Methodology
Realization of an FMCW Lidar Signal Processing Model
Stepwise Implementation
Continuous Verification and Validation
Experimental Reference Measurements
System Parameters
Reference Data

Figure 1-1: Methodology and structure.

In Chapter 3, the basic structure of the complete FMCW lidar sensor simulation is first presented and it is clarified how the signal processing model is defined in the scope of this work. Figure 1-1 illustrates the subsequent steps which are the core of this work. In order to develop a FMCW lidar signal processing model, it is necessary to define the requirements and select or develop an implementation approach. Furthermore, the model development methodology is needed, according to which the model is practically realized. The requirement definition, the implementation approach and the model development methodology are also introduced in Chapter 3 and serve as the basis for the following realization of the FMCW lidar signal processing model which is addressed in Chapter 4. Within the realization, the three design parameters beam pattern, radial range and relative radial velocity are iteratively implemented into the model step-by-step, as envisaged by the model development methodology. After each iteration, the model is verified and validated. For the implementation, information in the form of system parameters are needed. These system parameters are determined with the help of data sheets and operating instructions or are derived from experimental reference measurements. The experimentally obtained real lidar data are further employed for the validation of the signal processing model. In accordance with the three design parameters to be implemented, measurements are conducted for the beam pattern, the range and the velocity. For the validation of the model, the experimental measurements are re-simulated with the developed model and the resulting data is compared to the real reference data. Finally, in Chapter 5, the developed methodologies and the realization of the signal processing model are assessed by clarifying problems and weaknesses and future research objectives are mentioned.

2 Fundamentals

In the following, some important terms are defined in advance in order to avoid ambiguities. Afterwards, the fundamentals of FMCW lidars and ray tracing are explained. Lastly, the state of the art on perception sensor simulation is discussed.

2.1 Terminology

The first three terms are beam pattern, pixel and detection point. For the definition of a detection point, the DIN SAE SPEC 91471:2023-05[16a] standard is utilized. In this standard, the term detection describes a ”measured signal of the lidar sensor, whether representing ground-truth or artifacts or just noise”[16b]. A signal is defined as a ”single distance information at a certain azimuth and elevation which may as well contain intensity, width, noise floor and other parameters”[16b]. In the context of this work, this definition is adopted with the modification that detection is extended to detection point to emphasize that the detection is part of a lidar point cloud. A point cloud is defined according to the norm as a ”set of detections in one-, two-, or three-dimensional coordinate space”[16b].

In this work, the term beam pattern refers to the pattern according to which the lidar sensor scans the field-of-view (FoV) with the laser. A pixel is defined as a small discrete section of the beam pattern in which a single detection point may or may not occur. If the lidar sensor is capable of detecting multiple returns, more than one detection point may be contained in one pixel. The azimuth and elevation positions of the pixels do not have to be temporally constant.

Furthermore, the four terms experimental setup, measurement, recording and frame are to be defined. The term frame is described according to the standard as a ”periodical scan of the whole FoV”[16b]. The experimental setup is defined as the setup used to obtain data with the lidar sensor. In the context of this work, a recording refers to a collection of a certain number of consecutive frames captured by the lidar sensor. A measurement describes a set of recordings that are all taken with the same experimental setup. As an example, two measurements are conducted with five recordings each implies, that two different experimental setups are used and for each of the two, five recordings are captured. Two different experimental setups might be, for example, that a lidar target is set up once at a range of 10 m and once at a distance of 20 m.

Finally, the terms real reference measurement, real reference sensor and real reference data must be distinguished from term reference measuring device. The term real reference measurement implies a measurement with the real lidar sensor, which is used to record comparative data for validation. The lidar is therefore referred to as real reference sensor and outputs real reference data. Other measuring devices that are used to transfer the real experimental setup into the simulation are designated as reference measuring devices. The laser range finder and the ADMA employed in Chapter 4 are examples for reference measuring devices.

[16] DIN Deutsches Institut für Normung e. V.: DIN SAE SPEC 91471:2023-05 (2023). a: -; b: pp. 7-9

2.2 FMCW Lidar Sensor

Lidar is a technology that uses laser light to measure the range of a target. Typical pulsed ToF lidars transmit a short light pulse that is reflected back to the sensor when it hits a target and detected by a receiver. By measuring the time difference between transmitting and receiving of the pulse, the round-trip delay of the pulse to the target τ is calculated. Together with the speed of light in the medium air c_{air}, the range r to the target is determined by the equation $r = 0.5 \cdot c_{\text{air}} \cdot \tau$.[17] Since the intensity of the laser is modulated in a ToF lidar, it is also referred to as an intensity-modulation, direct-detection lidar.[18]

Lidars that use coherent detection, such as frequency modulated continuous wave (FMCW) lidars, are based on the modulation of the lasers phase or frequency domain. In contrast to the ToF lidar, the FMCW lidar transmits light continuously while linearly modulating the frequency in a triangular shape. At the same time, the intensity remains constant for the most part.[18]

In the following, the basic principle of the FMCW lidar is explained first. Subsequently, its architecture is described and the individual components are addressed.

2.2.1 Basic Principle

Coherent detection relies on the optical mixing of the received optical signal, that has been reflected by a target, with the optical signal of a local oscillator (LO). The signal of the LO is typically branched of the transmitting laser source (see Figure 2-2).[18] For a static target, the received signal is shifted horizontally by the round-trip delay τ in relation to the LO signal, as shown at the top of Figure 2-1. Thus, the resulting difference frequency Δf (often referred to as beat frequency) of the two signals is proportional to the range r of the target. However, in case of a moving target, the received signal is subjected to an additional vertical frequency shift $\Delta f_{\text{Doppler}}$ caused by the Doppler effect as depicted at the bottom of Figure 2-1. This frequency shift is proportional to the wave length of the laser λ_{laser} and the radial velocity of the target $\dot{r}$ and follows the expression $\Delta f_{\text{Doppler}} \approx 2\dot{r}/\lambda_{\text{laser}}$. This additional shift, caused by the Doppler effect, makes it possible to measure not only the range but also the radial velocity of the target at the same time. For this, however, it is first necessary to identify which part of the difference frequency Δf is caused by the range and which by the velocity of the target.[19]

[17] Behroozpour, B. et al.: Lidar System Architectures and Circuits (2017), pp. 135-136.

[18] Kim, T.: Diss., Realization of Integrated Coherent LiDAR (2019), pp. 8-9.

[19] Piggott, A. Y.: Understanding the physics of coherent LiDAR (2022), pp. 2-4.

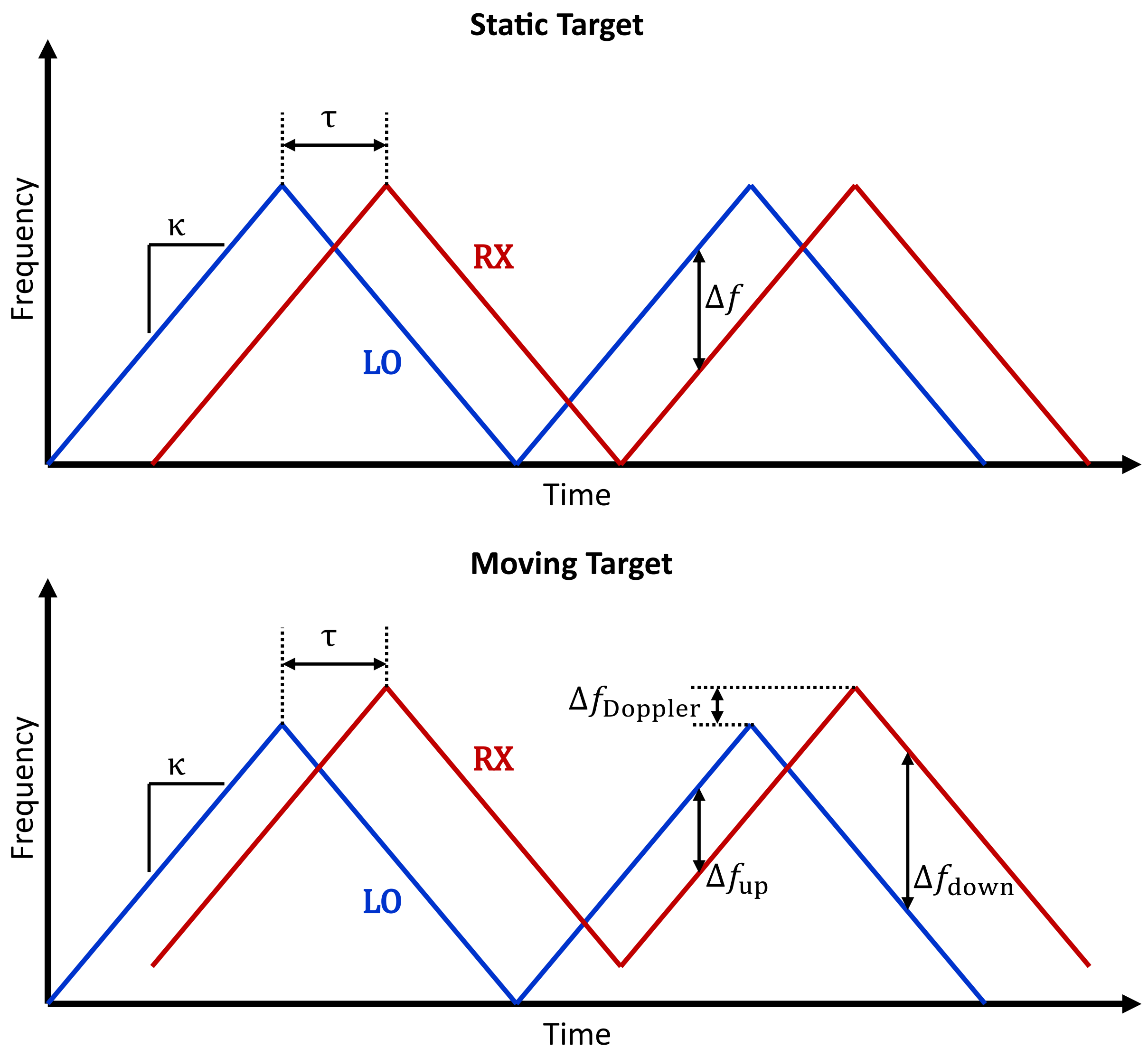


Figure 2-1: Both plots show the optical signal of the local oscillator (LO) in contrast to the optical received signal (RX). For a static target (top) the received signal is shifted horizontally. A moving target (bottom) results in an additional vertical shift. Own representation based on: [20a].

To be able to resolve this ambiguity, the triangular shape shown in Figure 2-1 with an up-ramp followed by a down-ramp is applied. By measuring the frequency difference during the up-ramp Δf_{up} and down-ramp Δf_{down}, the range and velocity are simultaneously determined.[20b] With the slope of the frequency ramp κ the range is defined as[20b]

$$r = \frac{c_{\text{air}}(\Delta f_{\text{up}} + \Delta f_{\text{down}})}{4\kappa}. \tag{2-1}$$

The equation for the velocity reads[20b]

$$\dot{r} = \frac{\lambda_{\text{laser}}(\Delta f_{\text{up}} - \Delta f_{\text{down}})}{4}. \tag{2-2}$$

[20] Piggott, A. Y.: Understanding the physics of coherent LiDAR (2022). a: p. 4, Fig. 2; b: pp. 2-4

2.2.2 Basic Architecture

The basic architecture of the FMCW lidar is illustrated in Figure 2-2 and includes the components: laser source, splitter, beam steering unit, coupler and balanced detector.

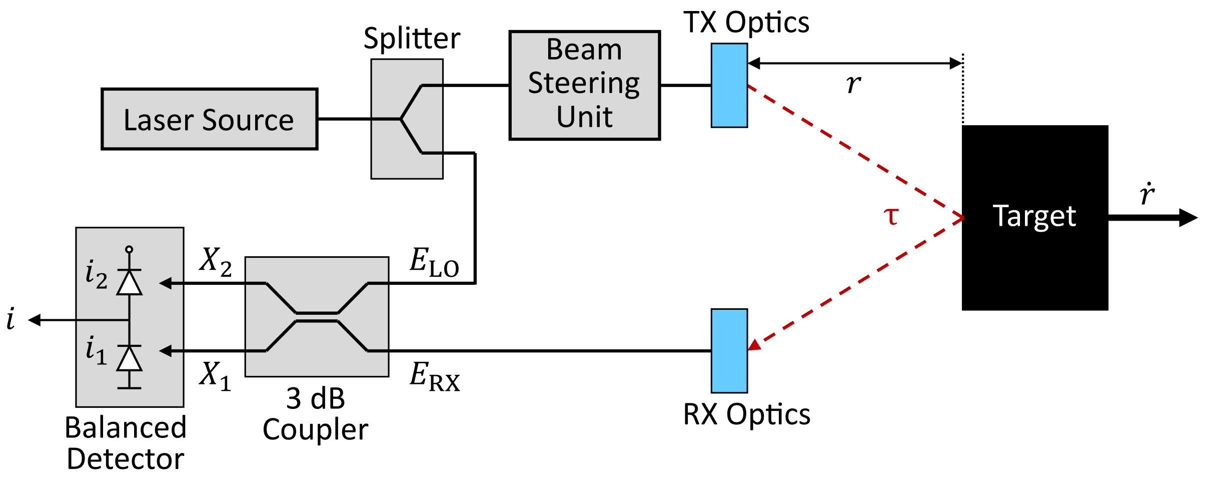


Figure 2-2: Basic Architecture of an FMCW lidar. Own representation based on: [21,22a,23].

In the upcoming chapters, the components are introduced and explained one after the other. The transmitter (TX) and receiver (RX) optics serve the purpose of separating the sensor from the environment and are not discussed any further.

2.2.3 Laser Source and Splitter

As deduced from the principle of the FMCW lidar, the laser source must be able to temporally change the frequency of the emitted light. Two possibilities to realize such a laser source are the tunable laser (TL) and the electro-optic modulator (EOM). For the TL, the frequency is modulated directly in the optical domain. In contrast, the EOM uses a laser source that has a fixed frequency. The fixed-frequency optical signal is mixed with an electrical signal within the EOM which results in an optical output signal. By varying the frequency of the electrical signal, the output's signal frequency is tunable. In theory, the result of both techniques is the same, but there are practical differences. Due to the larger modulation bandwidth of the TL, it is more suitable for applications where a very high resolution is required. However, lasers that are widely tunable, often experience higher phase noise. The phase noise reduces the coherence length which in return lowers the maximum operating range of the FMCW lidar. Therefore, for long-range applications where a lower resolution is acceptable, an FMCW lidar with an EOM is more suitable.[22b]

FMCW lidars often employ a laser source with the wave length of 1550 nm[24] which offers an advantage regarding eye safety regulations since light of wave lengths beyond 1400 nm gets absorbed by the front layers of the eye. This allows for higher power levels compared to other popular wave lengths like 905 nm

[21] Piggott, A. Y.: Understanding the physics of coherent LiDAR (2022), p. 2, Fig. 1.

[22] Behroozpour, B. et al.: Lidar System Architectures and Circuits (2017). a: p. 140, Fig. 5; b: pp. 140-141

[23] Barry, J. R.; Lee, E. A.: Performance of coherent optical receivers (1990), p. 1376, Fig. 11.

[24] Rablau, C.: LIDAR - Introducing optics to broader engineering and non-engineering audiences (2019), p. 12.

and thus greater ranges.[25] The downside is that the emitters and detectors of the 1550 nm wave length are more expensive due to the need for costly materials such as Indium Phosphide and Gallium Arsenide. Additionally, light of this wave length is more sensitive to environmental influences like rain, fog and snow. Although, this disadvantage is compensated for by the higher possible laser power.[25] Another important property of the laser source is the linearity of the frequency ramps, as this significantly influences the accuracy of the FMCW lidar.[26] As described above, the beam splitter is needed to branch off light from the laser source, that is used as the local oscillator.

2.2.4 Coupler and Balanced Receiver

As mentioned in Chapter 2.2.1, the main concept of coherent detection is the mixing of the LO signal with the received signal. To calculate the range and velocity, the difference in frequency of the two signals Δf needs to be extracted. This is done with a coherent balanced optical receiver that consists of a coupler followed by a balanced detector with two photodiodes (see Figure 2-2).[27] The commonly used 3 dB coupler adds the LO lightwave to the received lightwave[28a] and divides the input power of both lightwaves in a 50:50 ratio to both outputs.[29] The monochromatic received lightwave is represented by

$$E_{\mathrm{RX}}(t) = \sqrt{P_{\mathrm{RX}}} e^{j\omega_{\mathrm{RX}} t}, \tag{2-3}$$

and the LO lightwave by

$$E_{\mathrm{LO}}(t) = \sqrt{P_{\mathrm{LO}}} e^{j\omega_{\mathrm{LO}} t}, \tag{2-4}$$

where P is the average power of the lightwave that is defined by $P = |E(t)|^2$.[28b] The angular frequency ω is calculated by $\omega = 2\pi f$, where f is the frequency of the light. Besides the consideration of light as a wave, the assumptions are made that the phase and polarisation of the two waves are identical.[28a] Furthermore, any noise is neglected in the following consideration. The current output from a photodiode is expressed as

$$i(t) = RP, \tag{2-5}$$

where R is the responsivity of the photodiode.[28b] If the two inputs of a lossless 3 dB coupler are given by $E_{\mathrm{RX}}(t)$ and $E_{\mathrm{LO}}(t)$, the output lightwaves $X_1(t)$ and $X_2(t)$ are defined by[28a]

$$X_1(t) = (E_{\mathrm{RX}}(t) + E_{\mathrm{LO}}(t)) \frac{1}{\sqrt{2}} \tag{2-6}$$

and

$$X_2(t) = (E_{\mathrm{RX}}(t) - E_{\mathrm{LO}}(t)) \frac{1}{\sqrt{2}}. \tag{2-7}$$

[25] Roriz, R. et al.: Automotive LiDAR Technology (2022), p. 6284.

[26] Rablau, C.: LIDAR - Introducing optics to broader engineering and non-engineering audiences (2019), p. 12.

[27] Piggott, A. Y.: Understanding the physics of coherent LiDAR (2022), p. 5.

[28] Barry, J. R.; Lee, E. A.: Performance of coherent optical receivers (1990). a: pp. 1375-1376; b: pp. 1370-1372

[29] Mitschke, F.: Fiber Optics (2010), p. 132.

The output current $i(t)$ of the balanced detector is calculated by the difference of the currents of the two photodiodes $i_1(t)$ and $i_2(t)$. Together with Equation 2-5 this yields[30]

$$\begin{aligned} i(t) &= i_1(t) - i_2(t) \\ &= R(|X_1(t)|^2 - |X_2(t)|^2). \end{aligned} \tag{2-8}$$

Finally, with the equations 2-8, 2-3, 2-4, 2-6 and 2-7 the output current $i(t)$ of the balanced detector is expressed as[30]

$$i(t) = 2R\sqrt{P_{\mathrm{RX}}P_{\mathrm{LO}}}\cos[(\omega_{\mathrm{RX}} - \omega_{\mathrm{LO}})t]. \tag{2-9}$$

The output current of the coherent balanced optical receiver, therefore, oscillates in a sinusoidal waveform with the frequency $\omega_{\mathrm{RX}} - \omega_{\mathrm{LO}}$ that is equal to the frequency difference Δf of the two mixed lightwaves E_{RX} and E_{LO}. This phenomenon has various names such as light beating, optical mixing, photomixing, and optical heterodyning, to name a few.[31] After amplifying the output current, it is digitized with an analog to digital converter and fed into the digital signal processor. The frequency (and thus Δf) is extracted by taking the Fourier transform of the digitized sinusoidal waveform.[32]

Since the received signal is usually orders of magnitude weaker than the LO signal[33a], mixing has the advantage that the received signal is significantly amplified. This enables the use of relatively noisy detectors. Furthermore, a coherent detector is less sensitive to ambient light, since it is only able to detect light with a frequency similar to that of the LO.[33b]

2.2.5 Beam Steering Unit

To obtain a 3D image of the environment in form of the typical lidar point clouds, the focused laser beam must be directed to all regions of the desired FoV. This is done by a beam steering unit that redirects the generated laser beam and, thereby, scans the FoV in a certain pattern.[34] As already introduced, this certain pattern will be referred to as beam pattern and the discretized sections of the pattern are called pixels in the further course. There is a variety of different beam steering techniques which Roriz et al.[35] differentiate into the three categories: rotor-based mechanical lidar, scanning solid-state lidar and full solid-state lidar. Rotor-based mechanical lidars are widely present in the automotive industry in all kinds of variants and provide an up to 360° horizontal FoV. These lidars are often realized by mechanically rotating the scanning part or mirrors. Due to the rotating parts and the resulting added inertia, rotor-based lidars are usually heavier, more expensive and consume more energy.[35]

As a result of the absence of rotating components, scanning solid-state lidars represent a less expensive variant . However, this usually reduces the FoV. Two popular technologies of this lidar type are Microelectromechanical (MEMS) and optical phased array (OPA) systems. A MEMS lidar uses small

[30] Barry, J. R.; Lee, E. A.: Performance of coherent optical receivers (1990), pp. 1376-1377.

[31] Saleh, B. E. A.; Teich, M. C.: Fundamentals of photonics (1991), p. 70.

[32] Hashemi, H.: A Review of Silicon Photonics LiDAR (2022), p. 2.

[33] Piggott, A. Y.: Understanding the physics of coherent LiDAR (2022). a: p. 7; b: pp. 2-3

[34] Behroozpour, B. et al.: Lidar System Architectures and Circuits (2017), p. 136.

[35] Roriz, R. et al.: Automotive LiDAR Technology (2022), pp. 6285-6287.

electromechanical mirrors that change their tilt angles and, thereby, redirect the laser beam. To be able to scan in a 2D pattern, either a single mirror with two tilting axes is used ore two mirrors with one axes each. One of the disadvantages is that vehicle vibrations and shocks can have a negative impact on the MEMS mirror's performance. OPA systems modulate the phase of multiple laser emitters that are arranged in an array configuration. By applying different phases, the laser beam can be optically steered in different directions. The big advantage of this approach is that there are no moving parts at all which also allows for an integration on a chip.[36]

An example for a full solid-state lidar is the so called flash lidar, that illuminates the entire environment with a flash light and collects the back-scattered light with photo detectors. Since these do not direct a focused beam, but rather illuminate the entire environment at once, this approach will not be discussed further. In addition to the three types mentioned, there are other new approaches that, for example, combine aspects of the different types and approaches.[36]

2.3 Ray Tracing

Ray tracing is a rendering technique that is used to create photorealistic images of a 3D scene by following the path of a ray of light through the scene. As the ray traverses the scene, it may interact with objects of the environment.[37] The principle of ray tracing is illustrated in Figure 2-3. At first, a ray is generated and shot into the scene from the camera's position through a pixel of the image plane. Next, it is checked whether objects in the scene are intersected by the ray and if so, which one is closest to the camera's position. The process of finding the closest object along a ray is called ray casting. From the point of intersection, a new ray is casted towards the light source to be able to determine if the hit point is shadowed.[38a]

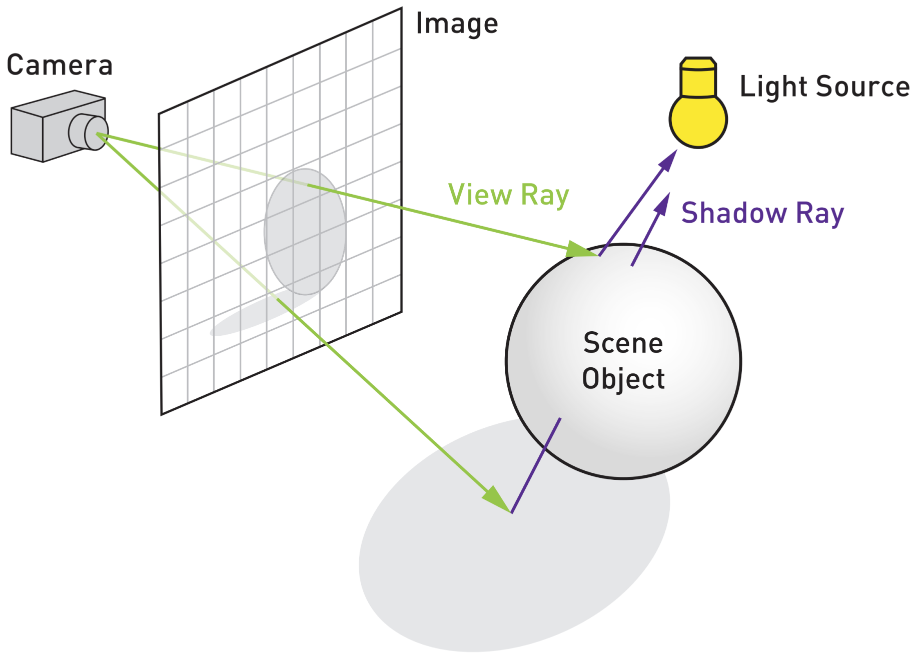


Figure 2-3: Basic principle of ray tracing.[38b]

[36] Roriz, R. et al.: Automotive LiDAR Technology (2022), pp. 6285-6287.

[37] Pharr, M. et al.: Physically based rendering (2017), p. 4.

[38] Haines, E.; Shirley, P.: Ray Tracing Terminology (2019). a: pp. 8-9; b: p. 9, Fig. 1-1

Furthermore, the reflective and refractive properties of the intersected object are also taken into account by shooting additional rays from the intersection point into the scene. If one of these new rays hits an object, a new intersection point is available where another set of rays is generated. This process is usually repeated until a maximum number of bounces is reached. By tracking back the resulting tree of rays the color of the pixel is determined. The material properties of the hit objects specify in which direction the new rays are shot into the scene from the intersection point (see Figure 2-4).[39a]

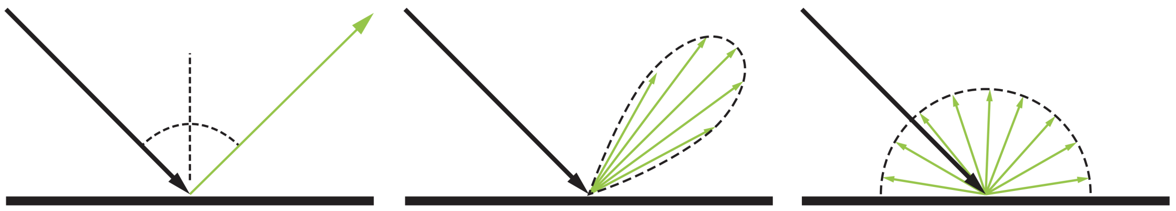


Figure 2-4: Different reflection types (mirror, glossy, diffuse).[39b]

Ray tracing utilizes the ray casting process to recursively gather information about the color of each pixel.[39a] In addition to the color information, there is also the possibility to calculate the range of the intersection points in relation to the camera position.[40] For this reason, ray tracing is used for the simulation of lidar sensors. Moreover, the ray tracer used in this work is able to output the velocity of the hit object, which is useful for the simulation of a FMCW lidar. Instead of the image plane shown in Figure 2-3, a pixel arrangement is applied for the lidar sensor simulation, that replicates the beam pattern of the real reference sensor.

2.4 State of the Art

In the following chapter, the state of the art on the topic of perception sensor simulation is addressed first. Afterwards, a short overview of the verification and validation of these sensors is given. Lastly, the topic of simulation credibility assessment is covered.

2.4.1 Perception Sensor Simulation

First, three possibilities for categorizing sensor models summarized by Rosenberger[41] are presented. Accordingly, he categorizes the sensor models by input and output data, by modeling approach, and by fidelity. These categorizations serve to give an overview of the different possibilities of sensor models and are also used to classify the FMCW lidar signal processing model developed in this work.

The first categorization possibility is based on the respective data inputs and outputs of the different sensor models. For a standardized naming, Linnhoff et al.[42] introduced a naming scheme that specifies a model as *<input>-based <technology> <output> model*. For the definition of the models input and output, the

[39] Haines, E.; Shirley, P.: Ray Tracing Terminology (2019). a: pp. 8-11; b: p. 11, Fig. 1-4

[40] NVIDIA Corporation: NVIDIA DEVELOPER Ray Tracing (2023).

[41] Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022), pp. 19-27.

[42] Linnhoff, C. et al.: Towards Serious Sensor Simulation for Safety Validation (2021).

interfaces from the Open Simulation Interface[43] (OSI) and the ISO 23150[44] international standard are employed.[45a]

In the second category, the models are classified according to the underlying modeling approach. Rosenberger differentiates between idealized models, stochastic models, physical models, and phenomenological models, which is also apparent in Table 2-1. The ground truth (GT) models listed in the table simply perform a coordinate transformation of all objects into the sensor coordinate system and are, therefore, not discussed further. Idealized models do not include physical effects of the signal and no information is lost from the original GT environment.[45a]

Models that are based on stochastic processes derived from real sensor data are called stochastic models or data-driven models. These models are often used to represent the noise of the intensity or the detection's position, for example. In order to be able to realize these models, real measurement data are required. For this reason, the complexity of the stochastic model depends strongly on the parameter space to be covered. The more complex and larger the parameter space is, the more the required amount of data increases, which leads to the result that at a certain point, for example, physical models are preferable.[45a]

Physical models render the simulated scene based on the sensor's wave length, resolution, and sensitivity and often focus on modeling the signal propagation. Different approaches are available to realize the rendering in a physical model like for example the Z-buffer method. This is an efficient method to calculate geometric occlusions of objects. An alternative approach is the previously described ray tracing, which is widely adopted in the simulation of perception sensors. The output of physical models is not only limited to detections, but also other interfaces are possible. As phenomenological models Rosenberger characterizes those, that integrate both physical and stochastic components.[45a]

Table 2-1: Categorization of active perception sensor system (APSS) models by modeling approach.[45b]

	GT "models"	**Idealized APSS models**	**Phenomenological APSS models**	
			Stochastic	**Physical**
Principle	Transformation of global GT in sensor perspective GT	Perfect FoV (only values the sensor can actually measure)	Phenomena from data (probabilistics & statistics)	Cause-effect chains leading to phenomena (e.g. signal attenuation, reflection, absorption, transmission, etc.)
Possible specifications	None (only transformation)	Position, orientation, idealized FoV	False detections (FP/FN), noise, pollution, manipulation, ..	Wave lengths, material properties, surfaces, signal processing, ...
Sensor accuracy	Not modeled	Not modeled	Realistic stochastic	Realistic single measurements
Complexity	None	Very small	Depends on parameter space	Very high to infinite

43 ASAM e.V.: ASAM OSI® (Open Simulation Interface) (2023).

44 International Organization for Standardization: ISO 23150:2021(E) (2021).

45 Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022). a: pp. 19-27; b: p. 21, Tab. 2-1

As the third category, Rosenberger cites the classification according to the sensor model's fidelity introduced by Schlager et al.[46a]. They distinguish between low, medium and high fidelity sensor models. Low fidelity models are characterised by the fact that they are based on geometric aspects of the objects in the environment and use object lists as input and output.[46b] Sensor models that take into account the FoV, the detection probability and physical aspects are referred to as medium fidelity models. The input of these models are still object lists, but the output may also be raw data.[46c] High fidelity sensor models are based on rendering techniques like, for example, ray tracing. Instead of object lists they take a 3D environment as an input and output raw data.[46d] Apart from the categorization, Schlager et al. give an overview of a large number of different perception sensor models for the simulation of camera, radar and lidar. These are sorted according to the introduced categorization and the advantages and disadvantages of each model are outlined.[46a]

2.4.2 Verification and Validation (V&V) of Perception Sensor Simulation

At the beginning, metrics for the verification and validation (V&V) of sensor simulations are addressed and the metric that is applied in this work is explained. Subsequently, a validation method is outlined, which later serves for the evaluation of the methodology used in this work.

Validation Metrics

The following section on metrics for sensor model verification and validation is based on the summary of Elster et al.[47a] both in terms of content and structure. According to Elster et al., in addition to the experimental setup and design used to acquire reference data, the choice of metric for comparing reference and simulated data is an important factor for the validation.[47b] Rosenberger[48a] compares and assesses a variety of different validation metrics and concludes that the area validation metric (AVM) is suited for the V&V of sensor simulations.[48b] The AVM introduced by Ferson et al.[49] is intuitive as it is based on computing the area between two cumulative distribution functions (CDFs) or empirical cumulative distribution functions (EDFs).[48b] Furthermore, the result of the AVM is in the unit of the measurand[48b] and the AVM has the ability to cover epistemic and aleatory uncertainties.[47b] Aleatory uncertainties are statistical deviations caused by the inherent randomness of the reference measurements. Information shortage concerning the model structure, world knowledge and measurement errors is the epistemic uncertainty.[47b,50]

The AVM is determined by calculating the integral of the absolute difference between two CDFs F and $\widetilde{F}$, where F represents real and $\widetilde{F}$ simulated data. The resulting equation reads

$$d_{\mathrm{AVM}}(F, \widetilde{F}) = \int_{-\infty}^{\infty} |F(\zeta) - \widetilde{F}(\zeta)| \, d\zeta, \tag{2-10}$$

[46] Schlager, B. et al.: State-of-the-Art Sensor Models for Virtual Testing (2020). a: -; b: p. 240; c: pp. 242-243; d: p. 247

[47] Elster, L. et al.: Introducing the Double Validation Metric (2023). a: pp. 4-10; b: pp. 4-5

[48] Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022). a: -; b: p. 102

[49] Ferson, S. et al.: Model validation and predictive capability (2008).

[50] Benke, K. K. et al.: Error propagation in computer models (2018).

where ζ is the measurand.[51] Given that the cumulated probability $F(\zeta)$ is limited to [0,1] and unitless with m quantiles, the formula is rewritten to

$$d_{\text{AVM}}(F, \widetilde{F}) = \frac{1}{m} \sum_{i=1}^{m} |\zeta(F_i) - \widetilde{\zeta}(F_i)|, \tag{2-11}$$

which defines the mean error of all m quantiles of the CDF.[51] Since EDFs describe the aleatory uncertainties, these are taken into account by the AVM, as already mentioned. To enable that the AVM also covers the epistemic uncertainties, multiple EDFs, so called p-boxes are utilized, instead of two single EDFs. The width of a p-box is defined by the two outer EDFs and describes the epistemic uncertainty at each quantile.[52a] For p-boxes the AVM is calculated by

$$d_{\text{AVM}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}}) = d^{-} + d^{+}, \tag{2-12}$$

where $\widetilde{\boldsymbol{\mathcal{F}}}$ is the simulated and $\boldsymbol{\mathcal{F}}$ the real p-box.[52a] Area d^+ represents the regions where $\widetilde{\boldsymbol{\mathcal{F}}}$ is greater than $\boldsymbol{\mathcal{F}}$, while area d^- delineates regions where it is lower, as depicted in Figure 2-5.[52a]

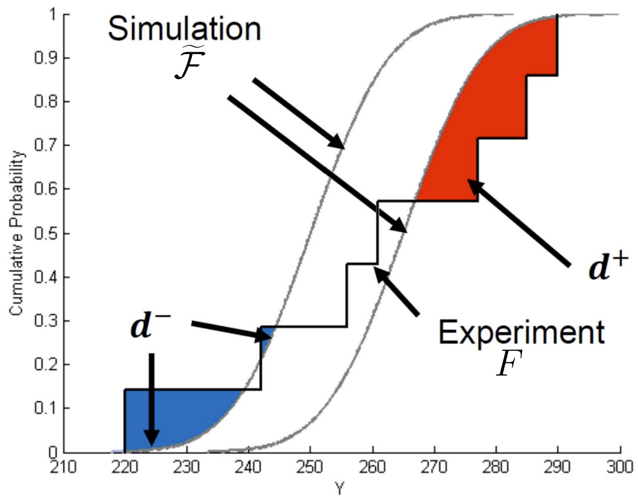


Figure 2-5: Calculation of AVM with p-box. In this case, $\boldsymbol{\mathcal{F}}$ is an infinitely small p-box.[53,52b]

For the validation measurements, it is important to minimise the uncertainties which, in practice, cannot be fully attained. For this reason, the uncertainties are also taken into account in the simulation.[52a] This is practically realised, for example, by measuring the range of a lidar target with an additional sensor such as a laser range finder. Assuming the laser range finder has an uncertainty of ± 1 cm, a first simulation is performed with the target at the measured range plus the uncertainty of 1 cm and a second simulation with the range minus the uncertainty. The two resulting EDFs then form the outer boundaries of the simulated p-box. However, the simulated p-box is not limited to two EDFs and it is possible to add more within the boundaries of the p-box.[52a]

[51] Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022), p. 105.

[52] Elster, L. et al.: Introducing the Double Validation Metric (2023). a: pp. 5-6; b: p. 6, Fig. 4

[53] Voyles, I. T.; Roy, C. J.: Model Validation in the Presence of Aleatory and Epistemic Uncertainties (2015), p. 4, Fig. 1.

As another requirement for the metric, Rosenberger mentions the ability to distinguish between model bias and model scattering error.[54a] While the model bias, depicted in Figure 2-6, is an approximation of the mean deviation, the model scattering error is the difference in the shape of the distribution functions.[55]

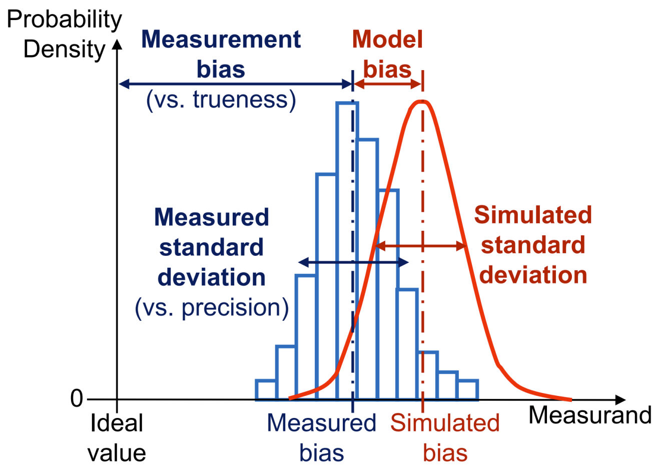

Probability Density
Measurement bias (vs. trueness)
Model bias
Measured standard deviation (vs. precision)
Simulated standard deviation
0
Ideal value
Measured bias
Simulated bias
Measurand

Figure 2-6: Bias and standard deviation of real and simulated measurement.[54b]

To obtain both, the model bias and the model scattering error, Rosenberger introduced the double validation metric (DVM) which is also applied later on in this work.[54c] First, the bias, as introduced by Voyles et al.[56], is calculated by

$$d_{\text{bias}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}}) = d^- - d^+, \tag{2-13}$$

which is similar to the AVM, but instead of the sum, the difference is taken. As a result, the symmetrically distributed area portions of the AVM are excluded. These portions define the model scattering error which means that only the model bias remains.[55] In order to receive the model scattering error in the next step, Rosenberger introduced a novel metric called corrected AVM (CAVM).[54c] For this, the simulated p-box $\widetilde{\boldsymbol{\mathcal{F}}}$ is corrected according to the previously calculated bias by[56]

$$\widetilde{\boldsymbol{\mathcal{F}}}_{\text{c}}(\zeta) = \widetilde{\boldsymbol{\mathcal{F}}}(\zeta - d_{\text{bias}}). \tag{2-14}$$

This results in a bias-free simulated p-box, which only contains the scattering error.[55] The CAVM is then determined by calculating the AVM with the corrected simulated p-box by

$$d_{\text{CAVM}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}}) = d_{\text{AVM}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}}_{\text{c}}) = d_{\text{c}}^- + d_{\text{c}}^+, \tag{2-15}$$

and is used to describe the model scattering error.[54c] The DVM is subsequently formed by combining the calculated model bias d_{bias} and model scattering error d_{CAVM}, resulting in[54c]

$$d_{\text{DVM}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}}) = (d_{\text{bias}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}}), d_{\text{CAVM}}(\boldsymbol{\mathcal{F}}, \widetilde{\boldsymbol{\mathcal{F}}})). \tag{2-16}$$

[54] Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022). a: p. 72; b: p. 11, Fig. 1-6; c: pp. 118-119

[55] Elster, L. et al.: Introducing the Double Validation Metric (2023), pp. 7-8.

[56] Voyles, I. T.; Roy, C. J.: Model Validation in the Presence of Aleatory and Epistemic Uncertainties (2015).

Accordingly, the DVM is obtained by first determining the bias of two p-boxes or EDFs and then calculating the model scattering error in a second step with CAVM.[57] This two step process is illustrated in Figure 2-7.

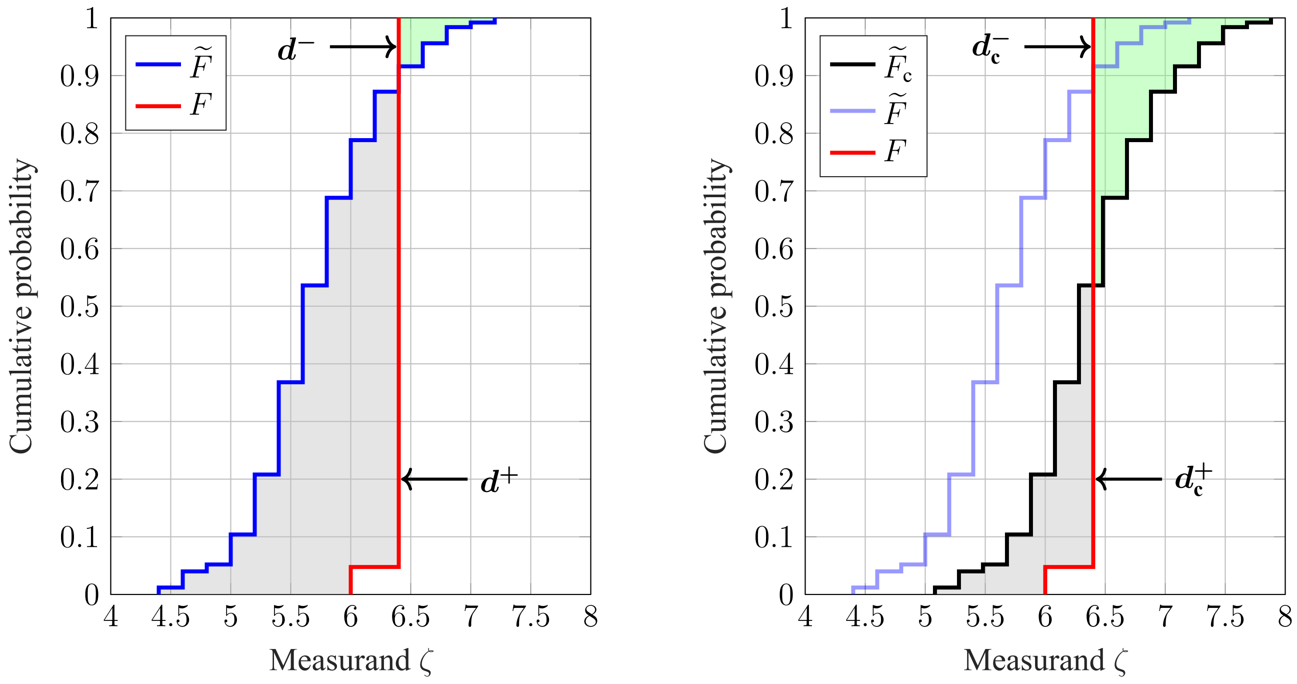


Figure 2-7: Calculation of the bias and the scattering error for two single EDFs with the DVM. First, d^- and d^+ are determined to compute d_{bias}. With the bias, the simulated EDF $\widetilde{F}$ is corrected to $\widetilde{F}_c$. Subsequently, d_{CAVM} is obtained by calculating d_c^- and d_c^+. Own representation based on: [58].

However, the DVM has limitations that are also related to the use of p-boxes. Elster et al. point out several edge cases that are problematic for the DVM.[57] To demonstrate this, two edge cases are addressed in the following. The first edge case, depicted on the left-hand side of Figure 2-8, occurs when a relatively wide simulated p-box is compared with a narrower real p-box and, in addition, the right boundary of the simulated p-box coincides with the left boundary of the real one. This leads to a relatively small bias compared to the width of the p-boxes, as indicated by the gray area between them. As long as the two p-boxes are right to each other, the width of the boxes do not affect the bias.[57] The second edge case appears when a single outlier in a set of real EDFs is close to the simulated p-box (see left-hand side of Figure 2-8). This constellation again results in a relatively low bias and the single outlier outweighs all other EDFs.[57]

[57] Elster, L. et al.: Introducing the Double Validation Metric (2023), pp. 7-10.

[58] Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022), p. 118, Fig. 9-2.

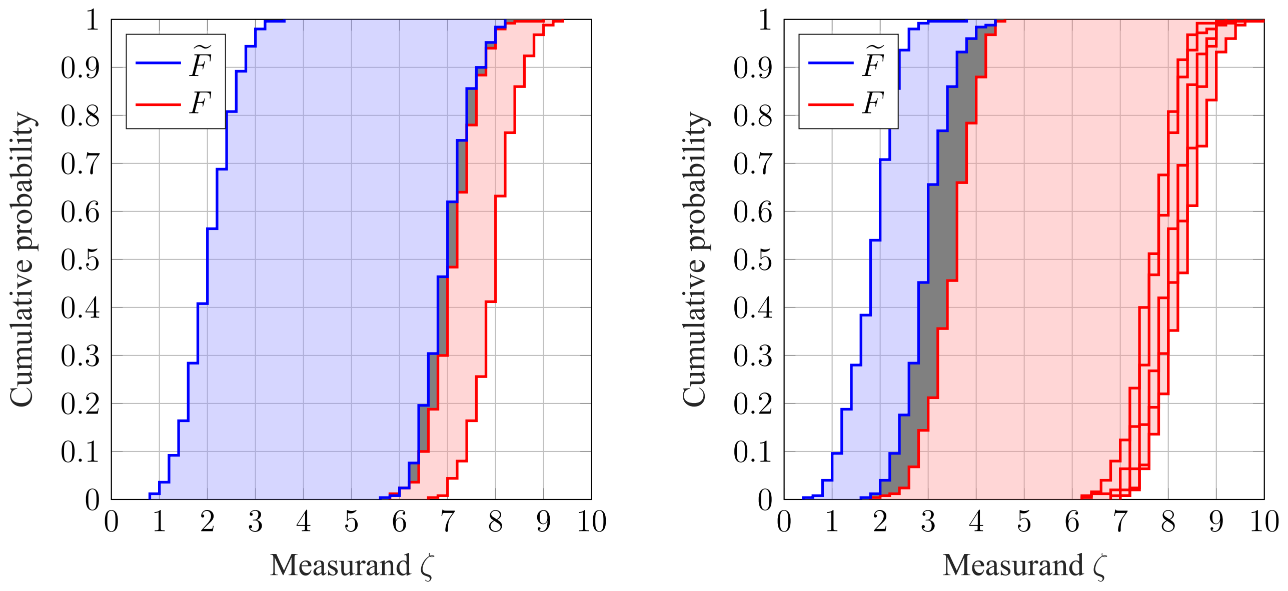


Figure 2-8: Edge cases for the DVM calculation with p-boxes. Own representation based on: [59].

For these reasons, no p-boxes are used for the DVM calculation in this work. Rather than computing a single model bias and model scattering error for two p-boxes (multiple EDFs), each simulated EDF is compared to each real EDF, one by one. For example, having five real and three simulated EDFs leads to 15 comparisons in total and, thereby, to 15 biases and scattering errors. These are summarized and visualized in two respective boxplots.[60]

Validation Method

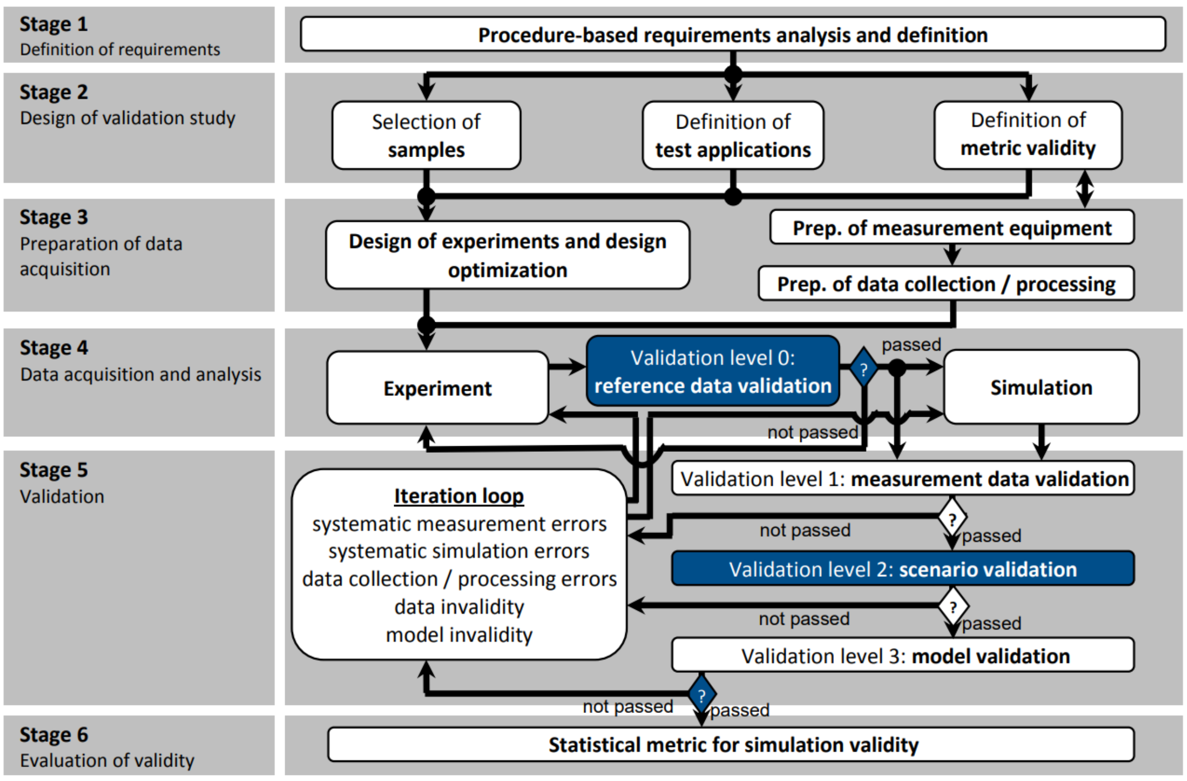
The following is a brief description of the method for an objective quality assessment of simulation models by statistical validation introduced by Viehof[61]. This method, initially developed for vehicle dynamics simulation, is adopted with minor modifications by Rosenberger et al.[62] for the validation of perception sensor simulation. Figure 2-9 illustrates the modified methodology of Rosenberger et al. As this method is only briefly addressed in the later assessment in Chapter 5, it is referred to the mentioned literature for more detailed information.

[59] Elster, L. et al.: Introducing the Double Validation Metric (2023), p. 9, Fig. 9 and 10.

[60] This idea originates from a consultation with M. Sc. Lukas Elster (FZD) and Dr.-Ing. Clemens Linnhoff (Persival GmbH).

[61] Viehof, M.: Diss., Objektive Qualitätsbewertung von Fahrdynamiksimulationen (2018).

[62] Rosenberger, P. et al.: Towards a Accepted Validation Methodology for Sensor Models (2019).



Stage 1
Definition of requirements
Stage 2
Design of validation study
Stage 3
Preparation of data acquisition
Stage 4
Data acquisition and analysis
Stage 5
Validation
Stage 6
Evaluation of validity
Procedure-based requirements analysis and definition
Selection of samples
Definition of test applications
Definition of metric validity
Design of experiments and design optimization
Prep. of measurement equipment
Prep. of data collection / processing
Experiment
Validation level 0: reference data validation
?
passed
Simulation
not passed
Iteration loop
systematic measurement errors
systematic simulation errors
data collection / processing errors
data invalidity
model invalidity
Validation level 1: measurement data validation
not passed
?
passed
Validation level 2: scenario validation
not passed
?
passed
Validation level 3: model validation
not passed
?
passed
Statistical metric for simulation validity

Figure 2-9: Method for objective quality assessment of simulation models by statistical validation.[63],[64]

2.4.3 Simulation Credibility Assessment

Finally, the simulation credibility assessment extensively discussed by Ahmann et al.[65a] is concisely covered. Although this topic is not explicitly addressed in the course of this work, it is of relevance, as care is taken to ensure compatibility with the methodologies developed in Chapter 3. Due to the increasing complexity of modern products, simulations are becoming more relevant for decision-making within the product development process.[65b] For this reason, it is important to ensure the credibility of these simulations, to lower the risk of unreliable decisions. Another important aspect is the traceability of information throughout the development processes, as they often take place in a distributed form and simulations are shared across organizational borders.[65b]

For this purpose, Heinkel and Steinkirchner[66] proposed the credible simulation process framework to integrate the credibility of simulations into the product development process.[65b] This framework is illustrated in Figure 2-10.

[63] Rosenberger, P. et al.: Towards a Accepted Validation Methodology for Sensor Models (2019), p. 10, Fig. 3.

[64] Viehof, M.: Diss., Objektive Qualitätsbewertung von Fahrdynamiksimulationen (2018), p. 47, Fig. 4-3.

[65] Ahmann, M. et al.: Towards Continuous Simulation Credibility Assessment (2022). a: -; b: pp. 171-173

[66] Heinkel, H.-M.; Steinkirchner, K.: Credible Simulation Process Framework (2023).

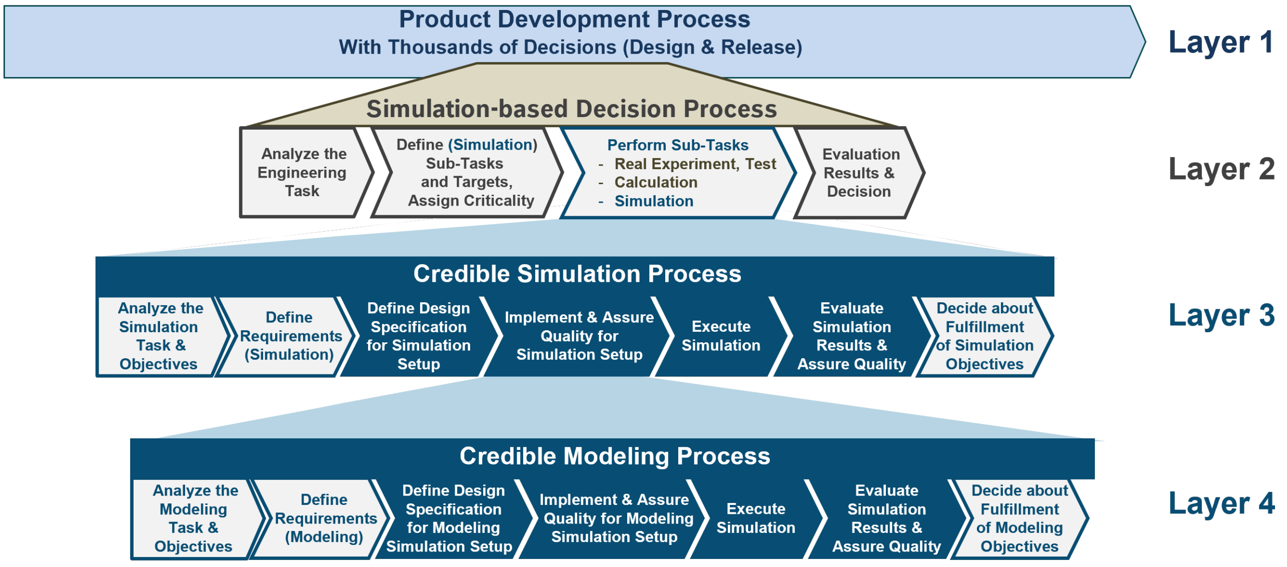

Product Development Process
With Thousands of Decisions (Design & Release)
Layer 1
Simulation-based Decision Process
Analyze the Engineering Task
Define (Simulation) Sub-Tasks and Targets, Assign Criticality
Perform Sub-Tasks - Real Experiment, Test - Calculation - Simulation
Evaluation Results & Decision
Layer 2
Credible Simulation Process
Analyze the Simulation Task & Objectives
Define Requirements (Simulation)
Define Design Specification for Simulation Setup
Implement & Assure Quality for Simulation Setup
Execute Simulation
Evaluate Simulation Results & Assure Quality
Decide about Fulfillment of Simulation Objectives
Layer 3
Credible Modeling Process
Analyze the Modeling Task & Objectives
Define Requirements (Modeling)
Define Design Specification for Modeling Simulation Setup
Implement & Assure Quality for Modeling Simulation Setup
Execute Simulation
Evaluate Simulation Results & Assure Quality
Decide about Fulfillment of Modeling Objectives
Layer 4

Figure 2-10: Structure of Credible Simulation Process.[67]

The product development process (Layer 1) includes various decisions, some of which are based on simulations. These decisions are part of the simulation-based decision process (Layer 2) and incorporate a simulation task. For each of these tasks a simulation request is sent to the credible simulation process (CSP) (Layer 3) transferring information. The CSP consists of different phases, as shown in Figure 2-10, that are iteratively executed. The models that are implemented into the simulation in the CSP are developed within the credible modeling process (CMP) (Layer 4). As before, there is a transfer of information, this time with modeling requests and the phases in the CMP are iteratively performed as in the CSP. During the CSP and CMP phases, metadata is collected to ensure traceability.[68] For more detailed and continuing information on simulation credibility assessment, it is referred to the mentioned publication of Ahmann et al.

[67] Heinkel, H.-M.; Steinkirchner, K.: Credible Simulation Process Framework (2023).

[68] Ahmann, M. et al.: Towards Continuous Simulation Credibility Assessment (2022), pp. 171-173.

3 FMCW Lidar Sensor Modeling

The objective of the upcoming chapter is to introduce the methodology that is later used to model the FMCW lidar sensor. First, the basic structure of the sensor simulation is presented, which also clarifies the term signal processing model. The following section outlines the process, through which the model's requirements are derived. These requirements are subsequently utilized as the foundation for the model development methodology, which is discussed in the final segment of this chapter.

3.1 Basic Structure of FMCW Lidar Sensor Simulation

The simulation of the FMCW lidar sensor is primarily subdivided into three components: environment simulation, signal propagation model and signal processing model (in the further course also referred to as sensor model). For one, these components are inspired by Cao[69], who breaks the sensor simulation down into environment model, wave propagation model and sensor performance model. In addition, some elements are built upon the work of Rosenberger et al.[70] in which a structure of a lidar sensor simulation is introduced. The generic decomposition of a sensor processing chain presented by Linnhoff et al.[71] is utilized as well. They divide the processing chain into the blocks emission, signal propagation, reception, signal pre-processing, detection identification and feature identification.

To begin with, the description of the environment simulation is adopted from Rosenberger at al. and thus includes, among other things, the roads, objects, traffic and dynamics that are necessary to model a real environment. Thereby, the environment simulation provides ground truth data that serves as the basis for the following signal propagation model.[70] The signal, traveling through the environment and possibly interacting with objects and particles, is part of the signal propagation model.[71] In the scope of this work, the signal propagation model covers the area outside the sensor housing and is realized by a ray tracing approach. As soon as the signal enters the sensor housing, the signal processing model takes over and simulates the processing steps from signal receiving to the output of the sensor model which, in this case, is a point cloud. These processing steps are termed receiver-side signal processing in the further course of this work. Additionally, the processing steps that are executed before or during the emission of the signal are included as well and will be referred to as transmitter-side signal processing.

How this simulation structure is to be practically implemented is depicted in Figure 3-1. Certain information, provided by a database, are needed to be able to simulate an environment. In order to achieve a high degree of compatibility, the ASAM standards OpenDRIVE[72] and OpenSCENARIO[73] are applied to define the environment in the database. Both formats are based on the extensible markup language (XML) syntax. While the OpenDRIVE format describes the static road network as well as objects and features along the roads[72], the OpenSCENARIO format defines the dynamic content of the environment, like vehicle maneuvers, other traffic participants, but also traffic and environment conditions[73]. In addition,

[69] Cao, P.: Modeling Active Perception Sensors for Real-Time Virtual Validation (2018).

[70] Rosenberger, P. et al.: Sequential lidar sensor system simulation (2020).

[71] Linnhoff, C. et al.: Towards Serious Sensor Simulation for Safety Validation (2021).

[72] ASAM e.V.: ASAM OpenDRIVE® (2021).

[73] ASAM e.V.: ASAM OpenSCENARIO® (2022).

the OpenMATERIAL[74] format is used for a physically correct implementation of material properties. The information contained by the database is utilized by a OpenSCENARIO player, for example esmini[75], and the resulting description of the environment is passed to the signal propagation model. For a standardized exchange of information that allows for a broad applicability, the components of the simulation use the Open Simulation Interface[76] (OSI) format to exchange data.

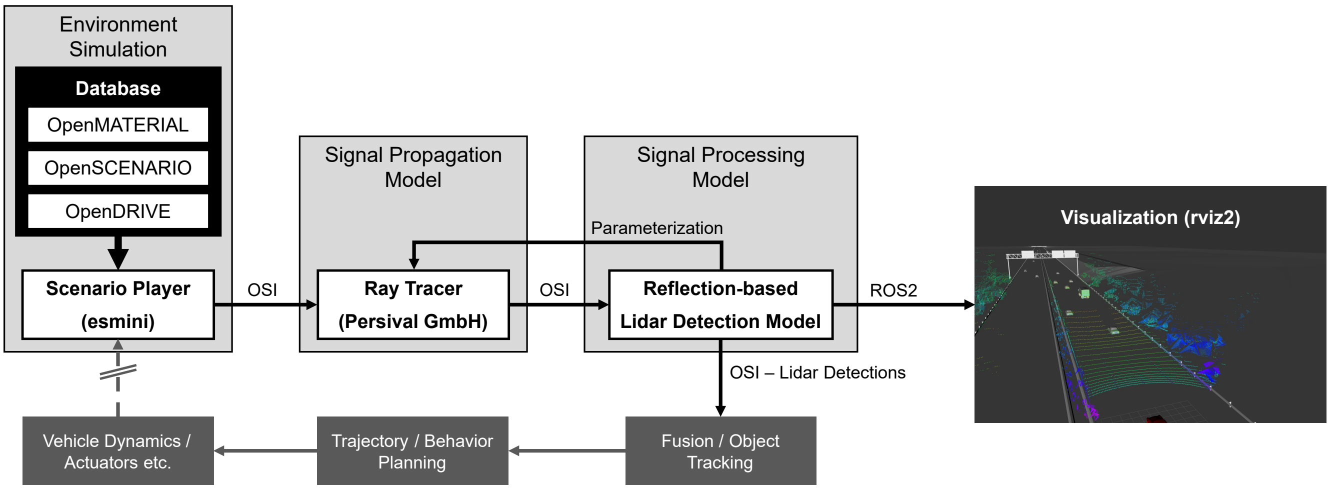


Figure 3-1: Basic structure of FMCW lidar sensor simulation.[77]

In this work, the signal propagation model is a ray tracer provided by Persival GmbH, which performs the ray tracing process on the input environment data and outputs the resulting reflections. These reflections serve as the input of the signal processing model which follows the naming convention introduced in Chapter 2: *<input>-based <technology> <output> model*. The (receiver-side) signal processing model includes all stages involved in deriving the point cloud output from the reflection input. To ensure that the ray tracer delivers a suitable result, it has to be parameterized first. This parameterization is part of the transmitter-side signal processing and executed before the first ray tracing is performed.

The signal propagation model and the signal processing model are packaged as a single Functional Mock-up Unit (FMU) which is a module that exchanges data, in this case OSI data, via the Functional Mock-up Interface[78] (FMI) for co-simulation.[79] OpenMCx[80] is used as the co-simulation framework. Since the environment scenarios simulated in this work are very basic, there is no need to create a complete database with an OpenDRIVE file etc. Instead, the environment is programmatically defined in a simple OSI publisher, that sends the data to a network-proxy FMU which is taken from OpenMSL[81]. This FMU serves as a connection between the OSI publisher and the signal propagation/processing model FMU.

[74] Ludwig Friedmann: OpenMATERIAL (2023).

[75] GitHub - esmini: Environment Simulator Minimalistic (esmini) (2023).

[76] ASAM e.V.: ASAM OSI® (Open Simulation Interface) (2023).

[77] The structure originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

[78] Modelica Association: Functional Mock-up Interface (FMI) (2023).

[79] Modelica Association: Functional Mock-up Interface Specification (2023).

[80] Eclipse Foundation: Eclipse OpenMCx (2023).

[81] Automotive Solution Center for Simulation e.V.: OpenMSL SL-5-3 OSMP Network Proxy (2023).

To be able to visualize the detection point output of the sensor model in real time, a third FMU converts the OSI data received from the signal propagation/processing model FMU into messages of the Robot Operating System 2[82] (ROS2) framework. The ROS2 visualizer rviz2 displays the point clouds in a graphical user interface (GUI). To save the output data of the sensor model for later offline processing, the third FMU is replaced by an OpenMSL trace file writer FMU[83], that instead of converting the OSI data, writes the data into so called OSI trace files. Later on, these trace files are converted into the more common CSV format, enabling the import of the sensor model output into Matlab for evaluation purposes.

The gray blocks in Figure 3-1 indicate what a complete simulation of an automated vehicle might look like. Since the focus of this work lies on the development of a signal processing model, none of the other blocks are dealt with in depth. The only exception is the ray tracer which is more involved because of the parameterization. Given that the ray tracing is based on a physical principle and the signal processing involves statistical components, as will be apparent in Chapter 4, the developed model is to be classified to a phenomenological modeling approach. It should be noted that the entire simulation behaves deterministically, and therefore, always delivers the same output for the same input. This behavior serves the reproducibility.

3.2 Requirement Definition and Implementation Approach

Before the development of the signal processing model starts, the requirements need to be defined. For this purpose, the requirements for the whole sensor simulation are derived first. On the one hand, the requirements are fundamental, as they specify what exactly needs to be implemented and on the other hand, they also function as the benchmark which the developed signal processing model or sensor simulation is tested against. An important factor for the sensor simulation requirements is the Operational Design Domain (ODD) of the system (e. g. an automated vehicle) in which the sensor is used. The ODD defines the operating conditions under which the system is designed to function.[84] For example, a sensor, used in automated vehicles on public roads, has to meet different requirements than a sensor for robots operating in closed factory buildings. To ensure that this basic information is available from the start, the ODD is defined at the beginning. Subsequently, the methodology, including the results, of the requirement definition is presented.

3.2.1 Definition of Operational Design Domain

For the following methodology that describes how the sensor simulation requirements are derived, the ODD represents an important foundation of information. Furthermore, the ODD is useful to define boundary conditions and requirements for the real validation reference measurements. The selected ODD is closely related to the one of the Mercedes-Benz Drive Pilot[85] which is limited to slow highway travel with high traffic volume. Thus, it is assumed that, in the context of this work, the simulated sensor is part of an automated traffic jam system that is similar to the Drive Pilot.

[82] Open Source Robotics Foundation: ROS - Robot Operating System (2023).

[83] Automotive Solution Center for Simulation e.V.: OpenMSL SL-5-6 OSI Trace File Writer (2023).

[84] SAE International: SAE-J3016 (2021), p. 17.

[85] Mercedes-Benz: DRIVE PILOT (2023).

The description of the ODD is based on the PAS 1883:2020[86a] standard published by the British Standards Institution (BSI). In this standard, the ODD is fundamentally divided into the three attributes scenery, environmental conditions and dynamic elements. Each of the attributes is further classified into more sub-attributes as depicted in Figure 3-2. Only the attributes that are relevant within the context of this work are described below.

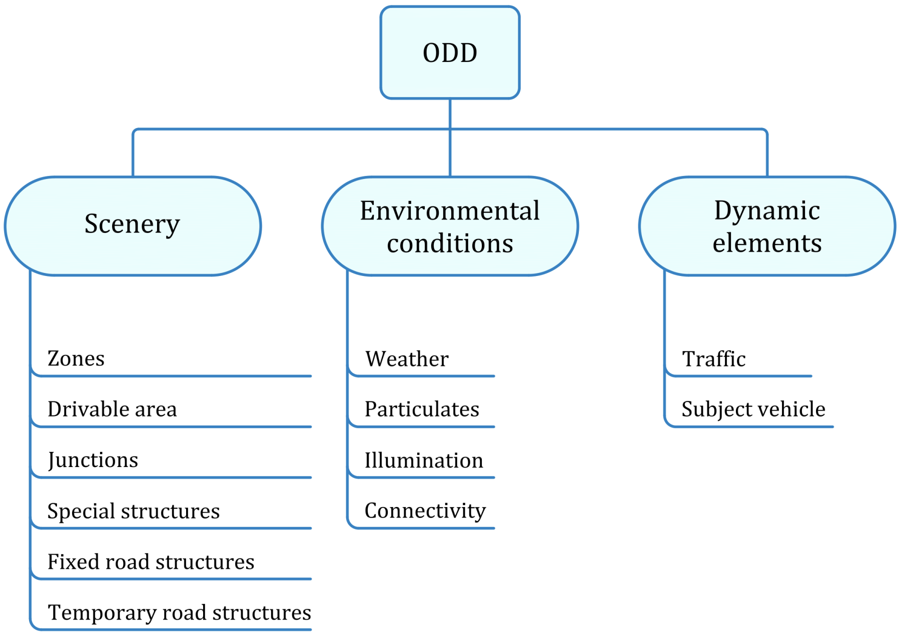


Figure 3-2: ODD attributes of BSI PAS 1883:2020.[86b]

In the following, the relevant attributes are defined utilizing the textual description format of the PAS 1883:2020 standard.

Scenery

Drivable area type:

- For drivable area type, we require [motorways].
- We do not allow [radial roads, distributor roads, minor roads, parking, shared space].

Drivable area lane specification:

- For number of lanes we require at least [two] lanes.
- For lane type we allow [traffic lane, emergency lane].
- We do not allow [bus lane, cycle lane, tram lane].

Drivable area surface:

- For drivable area surface type, we allow [asphalt, concrete].

[86] British Standards Institution: PAS 1883:2020 (2020). a: -; b: p. 5, Fig. 1

- For drivable area surface type, we do not allow [cobblestone, gravel, granite setts].
- For drivable area surface conditions, we do not allow [icy, snow, standing water, wet road, surface contamination].

Special structures:

- For special structures, we do not allow [tunnels, toll plaza].

Temporary road structures:

- For temporary road structures, we do not allow [construction site detours, road works, temporary emergency signage].

Environmental conditions

Rainfall:

- For rainfall, we require no more than [0 mm/h].

Snowfall:

- For snowfall, we require no more than [0 mm/h].

Illumination:

- For illumination, we allow [day, cloudiness, artificial illumination].
- We do not allow [night, low-ambient lighting condition].

Dynamic elements

Traffic:

- For agent types, we allow [vehicles, two-wheelers].
- We do not allow [vulnerable road users, animals, non-motorized agents, bicycles, parked vehicles].
- For volume of traffic, we require [very high traffic volume, traffic jam].
- For speed of traffic, we allow [up to 60 km/h].
- For traffic, we require [a vehicle ahead of subject vehicle].

Subject vehicle:

- For speed of subject vehicle, we allow [up to 60 km/h].

3.2.2 Methodology

The methodology, that is applied to derive the requirements for the FMCW lidar sensor simulation, is inspired by a step-wise approach proposed by Rosenberger et al.[87]. This approach includes the following five simplified steps:

1. The sensor simulation's output is defined with regard to the system under test (SUT) which uses this output as its input.
2. A catalogue of possible effects is needed that helps to identify and collect the effects that can be observed at the defined real sensor's output.
3. The relevance of the possible effects needs to be determined by analysing the impact of the effects on the system under test.
4. The approaches used to implement the effects are to be defined (for example physical or stochastic).
5. The accuracies of the selected effects are to be specified.

These five steps are used to derive a modified and extended methodology that is utilized to define the requirements for the FMCW lidar sensor simulation within this work. This new methodology is explained and executed in the following and its structure is illustrated in Figure 3-3. In general, this methodology is not only designed for the requirement definition of FMCW lidar sensor simulations, but is also applicable to other sensor simulations. The first adjustment is the division into internal and external sensor simulation requirements.[88]

External Sensor Simulation Requirements

External requirements are those requirements that are already defined in advance by external factors and are difficult to alter. This includes, for example, requirements resulting from the software or hardware with which the simulation is performed. Another set of requirements originates from the available interfaces of the real reference sensor or the real reference measurement data. To be able to validate certain sensor simulation interfaces or outputs, the real reference sensor or data must also provide the exact same interfaces. Accordingly, the availability of interfaces has a great limiting influence on the choice of possible effects and the implementation approach. This problem is discussed in more detail later on. Depending on the interfaces, the output of the sensor simulation is also specified.

[87] Rosenberger, P. et al.: Towards a Accepted Validation Methodology for Sensor Models (2019), p. 5.

[88] This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

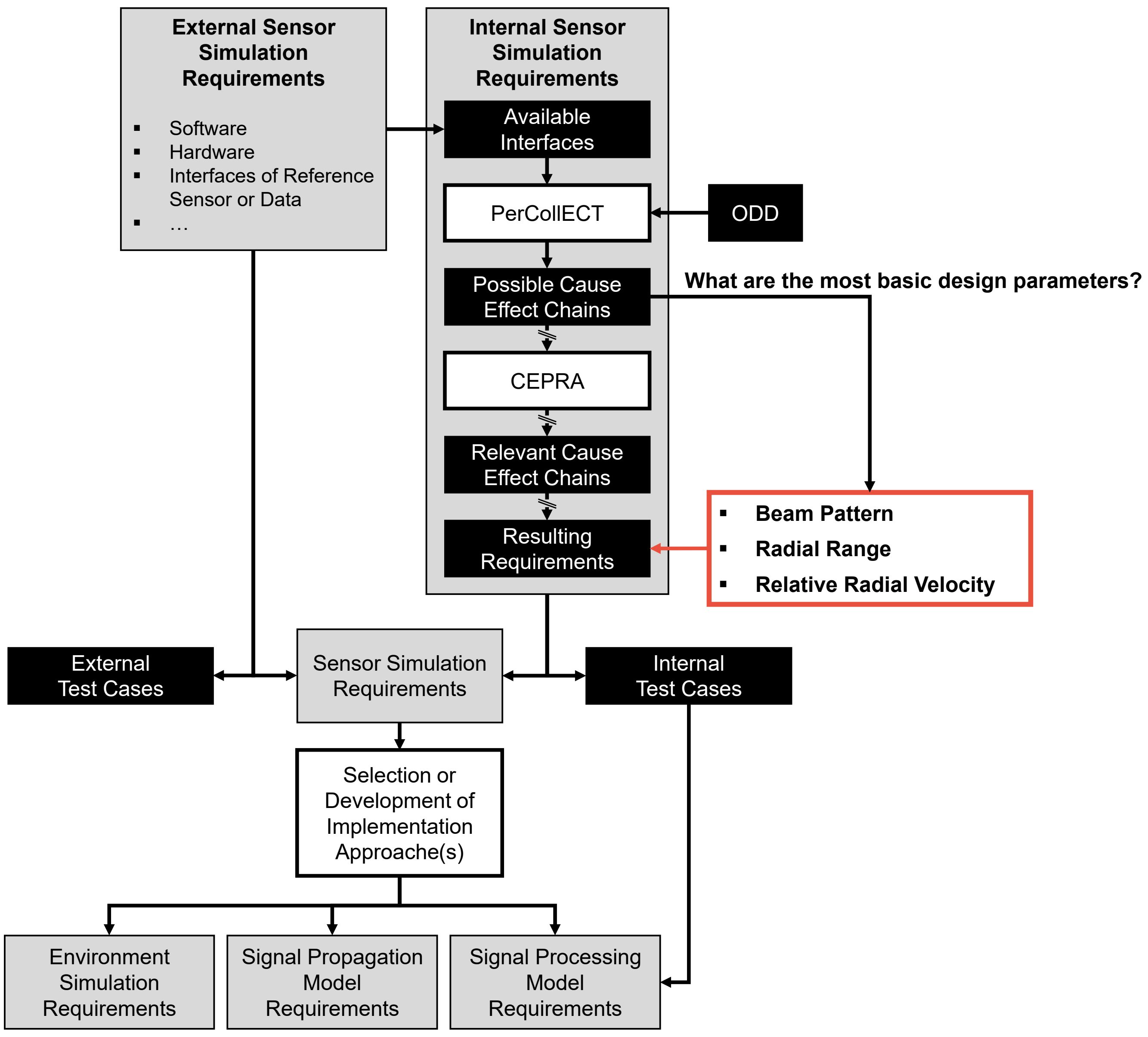

External Sensor Simulation Requirements
Software
Hardware
Interfaces of Reference Sensor or Data
…
Internal Sensor Simulation Requirements
Available Interfaces
PerCollECT
ODD
Possible Cause Effect Chains
What are the most basic design parameters?
CEPRA
Relevant Cause Effect Chains
Resulting Requirements
Beam Pattern
Radial Range
Relative Radial Velocity
External Test Cases
Sensor Simulation Requirements
Internal Test Cases
Selection or Development of Implementation Approache(s)
Environment Simulation Requirements
Signal Propagation Model Requirements
Signal Processing Model Requirements

Figure 3-3: Methodology for sensor simulation requirement definition.

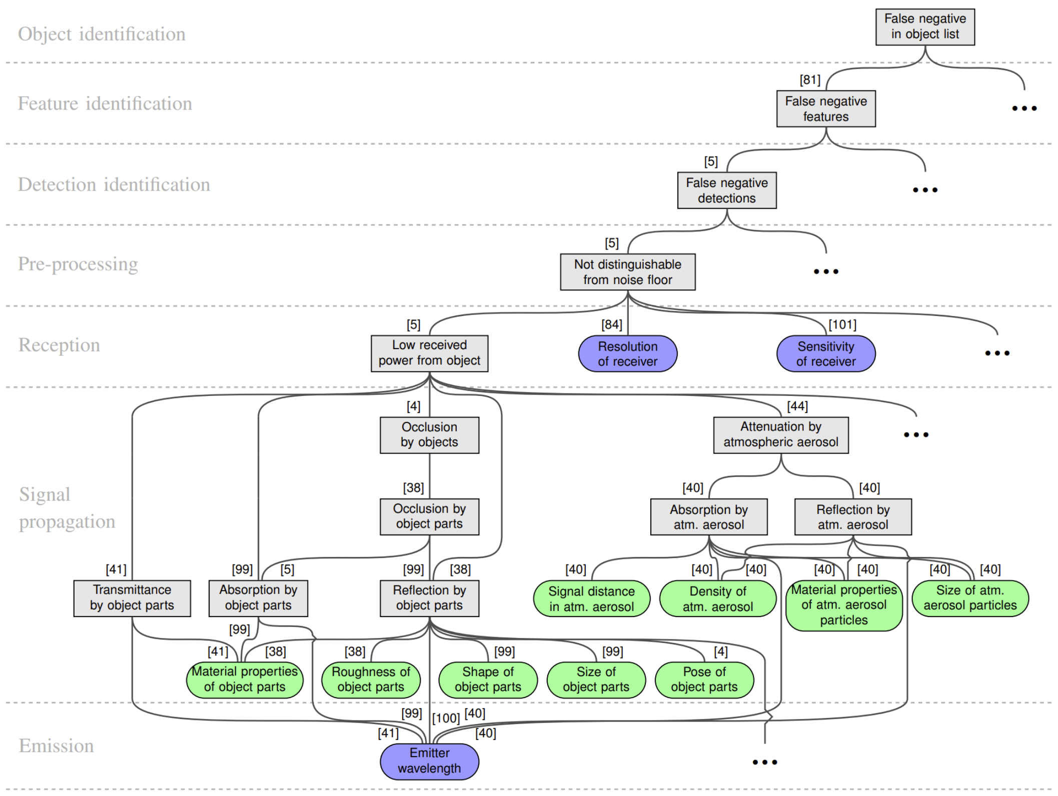
The following external requirements arise within the scope of this work. Since the only available interface of the real reference FMCW lidar sensor is on point cloud level, the output of the sensor simulation are detections in the form of point clouds. Furthermore, a SUT is also referenced which processes the data provided by the FMCW lidar sensor simulation. For the purpose of demonstration, it is assumed that the SUT is an object detector based on machine learning which takes the sensor's point clouds as an input and outputs corresponding object lists. Additional requirements result from the assignment including the structure of the sensor simulation given in Chapter 3.1. Note that external requirements such as computing time are not considered in the scope of this work.

Internal Sensor Simulation Requirements

The internal requirements define which properties of the real sensor must be replicated by the sensor model simulation. In contrast to the external requirements, internal ones are not given and are determined within the following procedure. This procedure is closely related to the five-step approach of Rosenberger et al. and mainly address their steps one, two and three.

The first step of Rosenberger et al., that involves the definition of the simulation output, is already addressed by the external requirements. Therefore, the available validation interfaces, including the simulation output, are taken from there and are utilized as a starting point (see internal sensor simulation requirements in Figure 3-3).

A possible solution for the second and the third step is provided by the work of Linnhoff et al.[89a] that introduces a two-stage method for model specification. The first stage proposed by Linnhoff et al. is a collection of cause-effect chains for perception sensors called Perception Sensor Collaborative Effect and Cause Tree[90] (PerCollECT). PerCollECT organizes the cause-effect chains in a tree representation that connects three different types of boxes as can be seen in Figure 3-4. Effects are marked by the grey boxes, green boxes symbolize causes that are independent of the specific sensor and the blue boxes indicate design parameters of the sensor.[89a] With PerCollECT, a catalogue is available that provides possible cause-effect chains.



Object identification
False negative in object list
Feature identification
[81]
False negative features
Detection identification
[5]
False negative detections
Pre-processing
[5]
Not distinguishable from noise floor
Reception
[5]
Low received power from object
[84]
Resolution of receiver
[101]
Sensitivity of receiver
Signal propagation
[4]
Occlusion by objects
[44]
Attenuation by atmospheric aerosol
[38]
Occlusion by object parts
[40]
Absorption by atm. aerosol
[40]
Reflection by atm. aerosol
[41]
Transmittance by object parts
[99]
[5]
Absorption by object parts
[99]
[38]
Reflection by object parts
[40]
Signal distance in atm. aerosol
[40]
[40]
Density of atm. aerosol
[40]
[40]
Material properties of atm. aerosol particles
[40]
[40]
Size of atm. aerosol particles
[99]
[41]
[38]
Material properties of object parts
[38]
Roughness of object parts
[99]
Shape of object parts
[99]
Size of object parts
[4]
Pose of object parts
Emission
[99]
[100]
[40]
[41]
[40]
Emitter wavelength

Figure 3-4: Exemplary excerpt of PerCollECT.[89b]

The second stage introduced by Linnhoff et al. is a method to evaluate the relevance of the possible cause-effect chains called Cause, Effect, and Phenomenon Relevance Analysis (CEPRA). A defined ODD for the sensor and the SUT is required as a prerequisite for CEPRA. If this is satisfied, two experts are consulted to determine the relevance of different cause-effect chains. At first, an expert on the sensor

[89] Linnhoff, C. et al.: Towards Serious Sensor Simulation for Safety Validation (2021). a: -; b: p. 4, Fig. 1

[90] Linnhoff, C. et al.: PerCollECT - LidarLimbs (2022).

evaluates how frequently the effect or cause-effect chain occurs within the ODD and rates the chance of occurrence with a value between 1 (Cannot occur) and 10 (Extremely high). Alongside, the impact of the cause-effect chain on the SUT is assessed by a second expert who is well-versed with the SUT. Again, the impact is rated by the expert by assigning values between 1 (Very low) and 10 (Very high). The sum of these two scores is used as a relevance score for the specific cause-effect chain.[91a] Figure 3-5 shows an excerpt of a CEPRA.

Pheno-menon (P)	**Effect chain (EC) of phenomenon**	**Causes of effect chains**	**P&EC occurrence in ODD*** (*O*, filled by sensor expert)		**P&EC impact on SUT in ODD*** (*I*, filled by SUT expert)		**Relevance of P&EC**
			$[1, 10]$	Rationale	$[1, 10]$	Rationale	$O + I$
False negative in object list	→ FN features → FN detections → Not dist. from noise floor → Low rec. power from object → Occlusion by objects → Occlusion by object parts → Reflection by object parts	• Materials of reflect. obj. parts • Roughness of reflect. obj. parts • Shapes of reflect. obj. parts • Size of reflect. obj. parts • Emitter wavel.**	*4*	*FN objects caused by occluding reflecting objects occurs rarely in a front radar on a highway, because of multi-path propagation.*	*6*	*FN obj. occurring because of occlusion in a front radar have a moderate impact because mainly only direct neighbor objs. considered.*	***10***
	→ FN features → FN detections → Not dist. from noise floor → Low rec. power from object → Reflection by object parts		*2*	*FN objects caused by compl. away-reflecting obj. cannot be ruled out, but are not expected on highway.*	*9*	*FN objects occurring in a front radar have a very high impact on a highway pilot.*	***11***
	→ FN features → FN detections → Not dist. from noise floor → Low rec. power from object → Attenuation by atm. aerosol → Absorption by atm. aerosol	• Signal dist. in atm. aerosol • ... • Emitter wavel.**	*3*	*FN objects caused by completely absorbing atmospheric aerosol occur only in harsh weather in a front radar on a highway.*	*5*	*FN objects occuring in harsh weather conditions may be covered by safety concept with a moderate impact on the highway pilot.*	***8***
	•••						
•••							

Figure 3-5: Exemplary excerpt of a CEPRA.[91b]

The cause-effect chains identified by this two-stage method could then be used as requirements for the sensor model.[91a] However, due to the complexity and size of PerCollECT and the necessary expert consultation for CEPRA, it is evident that it is not possible to perform this method within the scope of this work. For this reason, an abbreviated variant of the two-stage method introduced by Linnhoff et al. is utilized in the following. This variant is also represented in Figure 3-3.

Initially, on the basis of the available validation interfaces and with the help of the ODD, the number of possible cause-effect chains available in PerCollECT is narrowed down as much as possible. This is done by excluding all cause-effect chains that cannot be observed at any of the interfaces, as these would not be validatable. Accordingly, it is only possible to simulate design parameters, effects or sensor-independent causes that are observable at a sensor interface. Furthermore, all effects that are not covered by the ODD are neglected as well. For example, if it is defined that the system is not designed to operate in rain, all effects related to rain are excluded.

After obtaining a reduced number of cause-effect chains, the CEPRA is replaced by the following approach. Since the signal processing model in this work is newly developed, the focus lies on the most basic aspects first. For this reason, only the design parameters (blue boxes) of the narrowed down cause-effect chains are taken into account. This step is also related to the fact that a lot of effects are based on these design parameters. For example, it is not reasonable to implement effects like multiple returns per pixel without first realizing the divergence of the outgoing laser beam. Subsequently, instead of performing the CEPRA, the question of what are the most basic design parameters is addressed. However, the answer to this question is subjective and may not be universally valid, but it is sufficient in the context of this work to

[91] Linnhoff, C. et al.: Towards Serious Sensor Simulation for Safety Validation (2021). a: -; b: p. 5, Tab. 2

develop a basic model. The design parameters determined in this way are used as the internal requirements for the sensor simulation.

As already previewed in Figure 3-3, the three design parameters beam pattern, radial range and relative radial velocity are selected for implementation, verification and validation in the scope of this work. First, the beam pattern is chosen because it is one of the main characteristics of a lidar sensor. Thereby, the beam pattern decisively determines the structure of the point cloud that is output by the sensor. Especially if, as assumed in this work, an object detector is connected downstream as a system under test, the structure of the point cloud plays an important role. This is due to the fact that an object detector trained on a certain beam pattern suffers a loss in performance when applied to point clouds of a different beam pattern.[92] Therefore, if the sensor model is to be used in combination with the object detector trained on real data, a correct beam pattern is essential. The same applies if the model is intended for generating training data for the detector. Since the beam pattern only defines the azimuth and elevation angles of the detection in the point cloud, the radial range is chosen as the second design parameter. Thus, it is possible to fully define the position of the detection points. Lastly, the relative radial velocity is selected as the third design parameter as it represents a special ability of the FMCW lidar sensor. In addition, all three selected design parameters are observable at the only available interface, the point cloud output. After both the internal and external sensor simulation requirements are present, the next step is to define the test cases.

Test Cases

The test cases are derived from the internal and external sensor simulation requirements and fulfill the purpose of quantifying the requirements as well as making them testable by defining pass-fail criteria. Due to the division into internal and external sensor simulation requirements, internal and external test cases result respectively. Since the performance of the sensor simulation is of secondary importance in the scope of this work, the external test cases are not discussed further.

The definition of the internal test cases (further just referred to as test cases), therefore, serves the same purpose as the fifth step of Rosenberger et al., as they both determine the accuracies of the selected design parameters, effects or sensor-independent causes in some way. In general, a distinction is made between test cases for the verification and the validation of selected design parameters etc. Verification test cases compare the results of the simulation with nominal values to determine whether, for example, a design parameter is implemented as intended. In contrast, validation tests are used to check if the intended implementation of the design parameter also reflects the behavior of the real sensor by comparing the simulated data with real reference data. For each selected design parameter, effect or sensor-independent cause, at least one verification and one validation test case are derived.

In addition to the verification or validation metric that is applied, each test case also specifies which nominal values, simulated data, and real reference data are required to check the test case. Furthermore, the test cases also detail whether and how the data have to be pre-processed for the use of the metric. The defined pass-fail criteria always refer to the selected metric which may be different for each test case.

The test cases of a particular design parameter, for example, are always to be tested at the interface where the design parameter is observable. Accordingly, the test cases are also indirectly dependent on the available interfaces via the selected design parameters, effects or sensor-independent causes. For example,

[92] Tsai, D. et al.: See Eye to Eye: A Lidar-Agnostic 3D Detection Framework (2022).

an effect that is implemented into the sensor simulation, but not observable in the reference data due to missing interfaces, cannot be tested with a validation test case. As a consequence the effect is not selected and implemented in the first place.

Considering that the signal processing model, as defined in Chapter 3.1, includes all the steps performed within the sensor housing, the potential interfaces of the real sensor are also attributed to this model. Due to the fact that each design parameter, effect or sensor-independent cause must be observable at an interface of the real sensor, all test cases, derived from the internal sensor simulation requirements, are checked within the signal processing model (see Figure 3-3). However, the selection of the implementation approach may result in further test cases for the environment simulation, signal propagation model or signal processing model which will be discussed in the upcoming section of the implementation approach.

It is possible that parts of the environment simulation or signal propagation model are developed independently of a sensor. In this case the test cases for sensor-independent causes cannot be validated using a signal processing model. However, further considerations on this are not subject of this work.

Finally, it is discussed how the test cases are derived in practice. Ideally, there are existing sources that specify the accuracy of the design parameters, effects or sensor-independent causes to be implemented, from which the pass-fail criteria are determined. Another possibility is to observe, for example, a specific design parameter on two identical real sensors and compare the results. Through the comparison, it is possible to conclude which deviation of the simulation output is to be tolerated as a minimum.[93] Furthermore, the pass-fail criteria could be determined with an iterative sensitivity analysis.[93] For this purpose, the pass-fail criteria are assumed or estimated and a first version of the sensor simulation is developed. Subsequently, certain parameters are varied in the simulation and the impact of these changes on the SUT is assessed in the context of the sensitivity analysis. Afterwards, the results are used to improve the previous estimates and assumptions.

The definition of the test cases, used for the FMCW lidar sensor model of this work, follows in Chapter 4. Since no other real sensors and no resources to perform a sensitivity analysis are available, most of the test cases in this work are based on assumptions and estimates.

Implementation Approach

After the internal and external sensor simulation requirements are derived, the question arises how the selected design parameters, effects and sensor-independent causes are practically implemented into the sensor simulation. This issue is solved by the implementation approach that, in the context of this work, describes the practical solution with which one or multiple design parameters, effects and sensor-independent causes are realized in the sensor simulation. The selection of the implementation approach is comparable to the fourth step of Rosenberger et al.[94].

In contrast, the modeling approach is interpreted to be more like a classification of different implementation approaches. This is illustrated by the following example: Part of the implementation approach, applied later on in this work (Chapter 4), is to sample each pixel of the beam pattern with exactly one ray by the ray tracer. This approach is sufficient to implement the three selected basic design parameters: beam pattern,

[93] This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

[94] Rosenberger, P. et al.: Towards a Accepted Validation Methodology for Sensor Models (2019), p. 5.

radial range and relative radial velocity. A different implementation approach is chosen by Rosenberger et al.[95], who super-sample each pixel with several rays for their ToF lidar sensor model. This approach allows e. g. to simulate the echo pulse width as an output. Even though the two examples are considered to be different implementation approaches, both are classified as physical modeling approaches.

The implementation approach determines in which way and in which part(s) of the sensor simulation the selected design parameters, effects and sensor-independent causes are implemented. Certain effects could be realized in the environment simulation, the signal propagation model, the signal processing model or in several of them. Furthermore, the implementation approach does not have to be limited to a single design parameter, effect or sensor-independent cause but can also be involved in the implementation of multiple ones. Therefore, when developing and selecting the implementation approach(es), it makes sense to have an overview of all design parameters etc. derived in the requirement definition.

Certain information about the sensor may be required for the practical implementation of design parameters, effects or sensor-independent causes. This information is referred to as system parameters in the following. The power, wave length and beam divergence of the laser, as well as the beam pattern of the sensor are all examples of system parameters. System parameters are obtained from various sources such as data sheets or operating instructions, but are also determined experimentally. How the system parameters for the design parameters: beam pattern, radial range and relative radial velocity are determined is described in Chapter 4.

The requirements for the environment simulation, the signal propagation model and the signal processing model result from the choice of implementation approach(es). For illustration purposes, the effect that raindrops cause additional lidar detections in the atmosphere is considered.[96] This effect could, for example, be physically realised in the environment simulation by simulating the individual raindrops with a fluid dynamics simulation.[97] Another possible implementation approach could be a statistical simulation of the additional detections within the signal propagation or processing model.[98] Thus, the requirements for the three parts of the sensor simulation are dependent on the implementation approach. These requirements may also lead to additional test cases for the respective parts of the simulation.

However, the previously defined test cases that result directly from the internal sensor simulation requirements remain independent of the implementation approach and are always tested within the signal processing model. In the event that the effect of additional rainfall-related detections is realized in the environment simulation, it remains essential to assess the implementation of this effect at the sensor interface where it is observable using test cases. Since all possible implementation approaches are required to implement all selected design parameters, effects and sensor-independent causes in some way, the respective interfaces for the test cases are always located within the signal processing model regardless of the implementation approaches chosen.

When it comes to the fidelity of the implementation approach and to the question at which point of the sensor simulation the approach should be applied, the following has to be considered. In addition to the internal sensor simulation requirements already mentioned, the external requirements must be taken

[95] Rosenberger, P. et al.: Sequential lidar sensor system simulation (2020).

[96] Linnhoff, C. et al.: Measuring the Influence of Environmental Conditions on Lidar Sensors (2022).

[97] This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

[98] Linnhoff, C.: Diss., Analysis of Environmental Influences for Simulation (2023), p. 98ff.

into account. It is particularly important, for example, to match the computational effort required for the implementation approach with the available computing power of the hardware. This is necessary to keep the computing time within the requirements. Even if a high fidelity implementation approach provides better results, it is essential to ensure that the calculations for this approach are feasible. Additionally, it is not reasonable to choose a high fidelity implementation approach that simulates low level interfaces if there is no possibility to validate these due to missing interfaces of the real reference sensor. Accordingly, only interfaces that are also used for the test cases of the selected design parameters, effects and sensor-independent causes should be considered in the implementation approach. For these test cases it is already ensured that the interfaces are available.

In the context of this work, the only available interface is the point cloud output of the real FMCW lidar sensor. As a result, a very simple implementation approach is chosen, in which each pixel of the beam pattern is only sampled with a single ray. This implementation approach provides a sufficient basis to be able to implement all three design parameters with just a few extensions that are addressed in Chapter 4. Since all three parameters are to be implemented in the signal processing model, there are no additional requirements for the environment simulation. On the other hand, the ray tracer, i. e. the signal propagation model, is required to be able to output not only the range of an object but also its velocity. Since the signal propagation model lies outside of the focus of this work, the resulting additional test cases are neglected and the signal propagation model is considered to be fully validated.

3.3 Stepwise Implementation with Continuous V&V Model Development Methodology

This chapter presents the methodology that is used to develop the signal processing model (sensor model). The methodology involves a step-by-step implementation with continuous verification and validation.[99] The basic idea behind this is, that the design parameters, effects and sensor-independent causes resulting from the previous requirement definition are taken separately and are gradually implemented into the sensor model. After each implementation step, a complete verification and validation of the sensor model is performed before the next design parameter, effect or sensor-independent cause is implemented. Through this procedure, the sensor model is extended step-by-step and continuously verified and validated. This methodology allows the model to be highly modular, customizable and extensible. Further advantages are that a validated model is available at each stage of the development process and it is possible to automate the verification and validation process. In the following, the methodology is first presented and afterwards future extensions are outlined.

3.3.1 Methodology

The methodology for developing the signal processing model, introduced below, assumes that the internal sensor simulation requirements are available in the form of design parameters, effects or sensor-independent causes. In addition, the corresponding test cases, implementation approaches and system parameters are expected to be known.

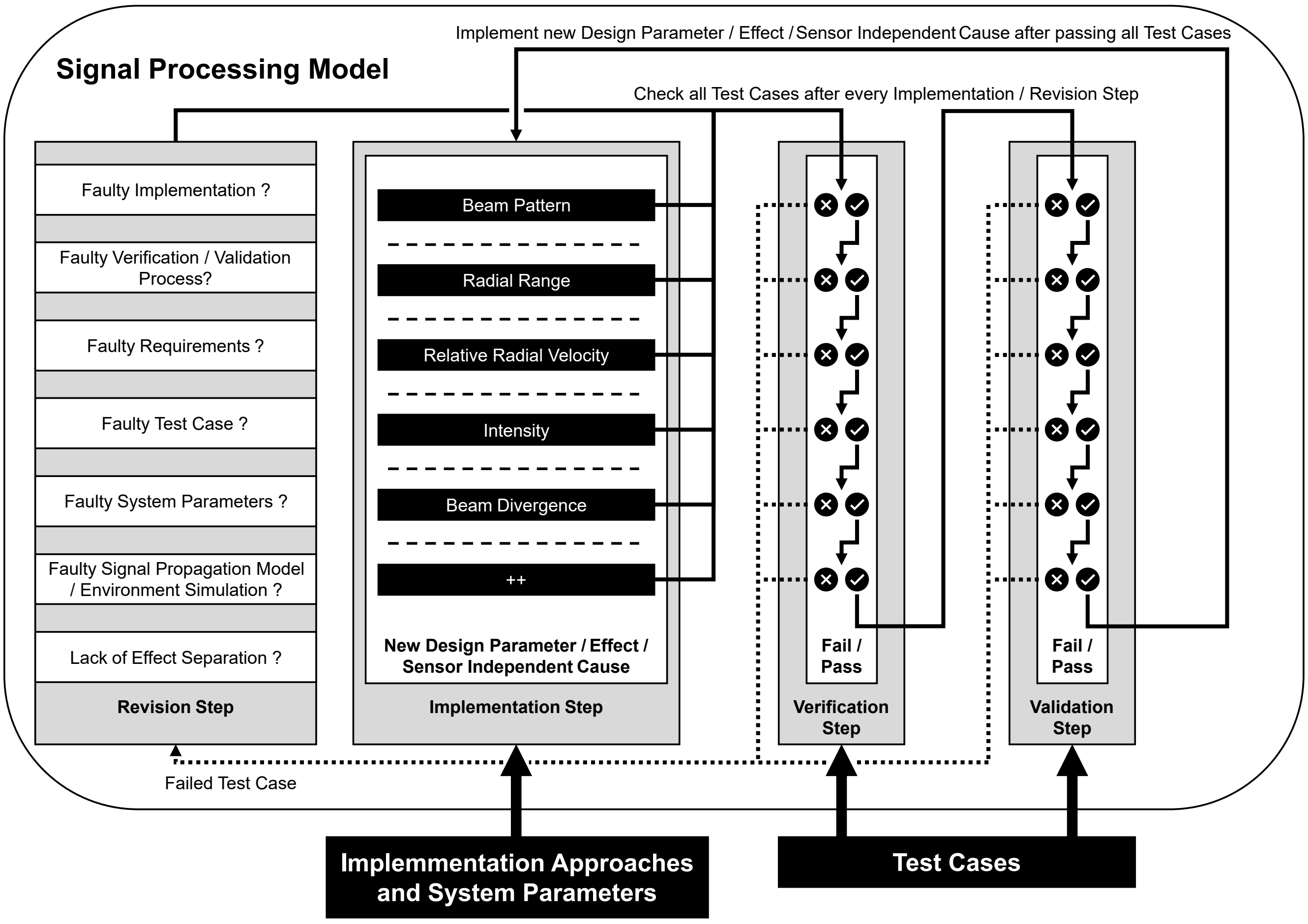
Figure 3-6 reveals that the methodology is divided into the four main steps: implementation step, verification step, validation step and revision step. The methodology proposes an iterative step-by-step development process that starts at the **implementation step**. Within this step, the first design parameter,

[99] This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

effect or sensor-independent cause, in this case the sensor's beam pattern, is implemented into the signal processing model. For this purpose, the respective implementation approach and, if necessary, system parameters are used. It is important to mention that, within each implementation step, only one design parameter, effect or sensor-independent cause is implemented at a time.

After the first, in this example, design parameter is implemented, the **verification step** follows. In this step, the verification test cases are checked one after another. The test cases are gradually added with each iteration according to the design parameter, effect or sensor-independent cause that was implemented in the preceding implementation step. In each verification step, all existing test cases are checked again from the beginning, even if they have already been passed in the previous iteration. This is to ensure that subsequent implementations do not interfere with the previous ones. Since there are no previous test cases in the first iteration, only the test cases for the beam pattern need to be checked.

Once all test cases have been passed, the **validation step** follows. This step is equivalent to the verification step, with the distinction that validation test cases are assessed. After passing these test cases as well, the next iteration is initiated which again starts with an implementation step. Thus, the next design parameter, effect or sensor-independent cause is added to the model, whereupon the verification and validation steps follow.



Signal Processing Model
Implement new Design Parameter / Effect / Sensor Independent Cause after passing all Test Cases
Check all Test Cases after every Implementation / Revision Step
Faulty Implementation ?
Faulty Verification / Validation Process?
Faulty Requirements ?
Faulty Test Case ?
Faulty System Parameters ?
Faulty Signal Propagation Model / Environment Simulation ?
Lack of Effect Separation ?
Revision Step
Beam Pattern
Radial Range
Relative Radial Velocity
Intensity
Beam Divergence
++
New Design Parameter / Effect / Sensor Independent Cause
Implementation Step
Fail / Pass
Verification Step
Fail / Pass
Validation Step
Failed Test Case
Implemmentation Approaches and System Parameters
Test Cases

Figure 3-6: Model development methodology for the signal processing model.

However, as soon as one test case is not passed in either the verification step or the validation step, a **revision step** is initiated. Within this step, an investigation is conducted to identify the reason for the failure of the test case and to eliminate it. Figure 3-6 lists some possible questions to ask to determine the cause. Nevertheless, this list of questions is by no means to be considered complete. After the problem is deemed fixed, the verification and validation steps are repeated to confirm this. If a test case fails again, the revision step is restarted. This procedure is reiterated until all test cases have been passed and subsequently a new implementation step begins. This method leads to an iterative development of the signal processing model which is continuously verified and validated after each modification.

3.3.2 Future Extension and Automation of Model Development Methodology

In the course of this work, the presented model development methodology is only applied for the development of the signal processing model. In the future, it is conceivable to extend the methodology to the entire sensor simulation and, thus, use it to develop the environment simulation and the signal propagation model. Figure 3-7 depicts the structure of a possible extension.

In this method the selected design parameters, effects or sensor-independent causes are again implemented iteratively, and one iteration spans all three parts of the simulation. If a selected design parameter, for example, is successfully added to the environment simulation, it is subsequently implemented into the other two parts as well. Once the design parameter is successfully integrated into the signal processing model, a new iteration begins. In the case of failed test cases, the entire sensor simulation would have to be investigated for the fault during the revision step.

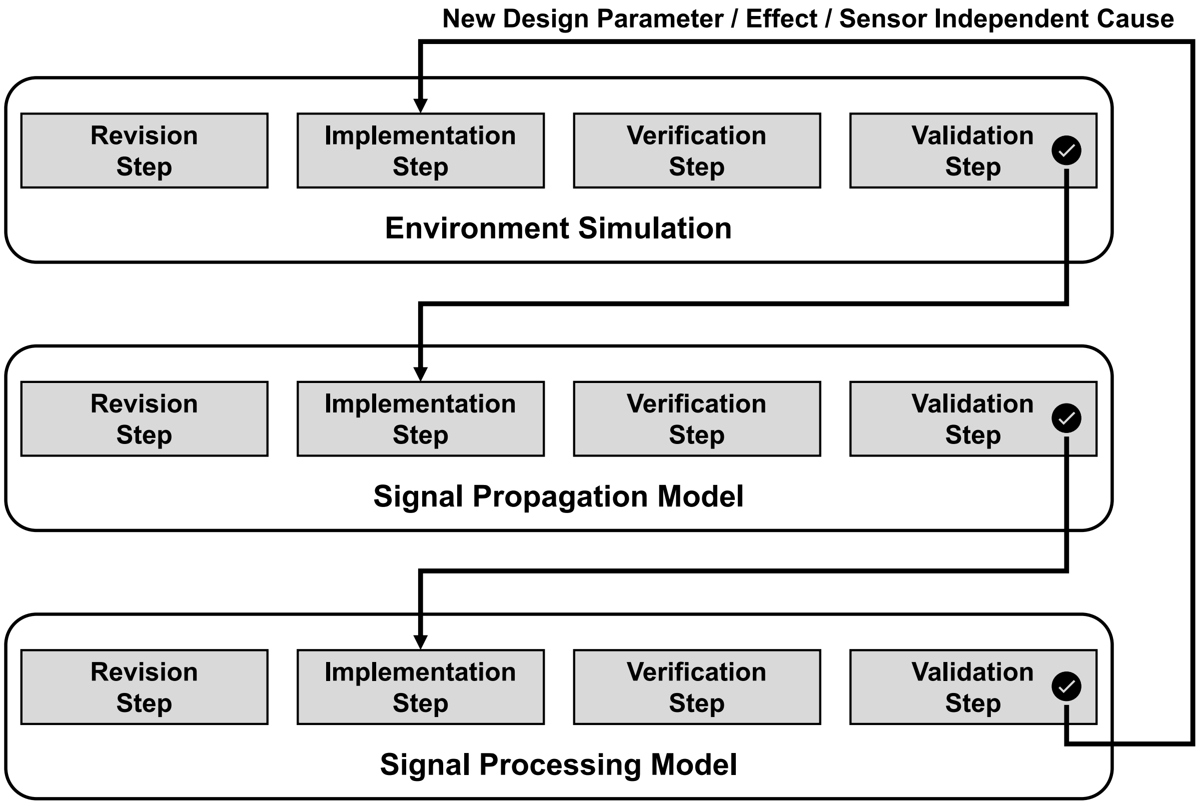


Figure 3-7: Future extension of the model development methodology. All unlabeled arrows do not pass any information but simply indicate the sequence of execution.

Another aspect that should be addressed in the future is the automated evaluation of the verification and validation test cases. Given that the number of test cases increases with every newly implemented design parameter, effect or sensor-independent cause and all existing test cases are re-tested in each iteration, a considerable amount of effort is required for more complex signal processing models. Therefore, there is interest in automating the review of the test cases.

This could possibly be realized by appending an evaluation block to the structure shown in Figure 3-1, that compares the simulation output with reference data or nominal values using the test cases.[100] In this evaluation block, all test cases with their corresponding data preparation, metrics, and pass-fail criteria are to be programmed. Furthermore, all simulation scenarios that are needed to generate the simulated data, as well as the reference data and nominal values must be available in a database.

Adding the test cases to the automated evaluation results in an additional effort for each iteration. However, the effort involved will be beneficial later on as in each iteration of the development process of the signal processing model, the previous test cases are automatically re-tested without additional work.

[100]This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

4 Exemplary Realization of an FMCW Lidar Signal Processing Model

In the following the presented model development methodology is carried out exemplarily for the three in Chapter 3.2 selected design parameter beam pattern, radial range and relative radial velocity. The implementation approach mentioned in the same chapter is used for this purpose. The processing of the real and simulated sensor data as well as the calculation of the verification and validation results is performed with Matlab.

4.1 Beam Pattern

This chapter addresses the implementation, verification and validation of the sensor model's beam pattern. First, the methodology is briefly showcased along with the requirements and test cases. Next, the system parameters that define the model's beam pattern are derived with the help of experimental reference measurements. Finally, three iterations of the implementation, verification and validation steps are performed.

4.1.1 Methodology and Detailed Requirements with Test Cases

Regarding the beam pattern, it is assumed that a sensitivity analysis leads to the conclusion that the positions of the pixels are to be treated as temporally constant. Consequently, effects like noise or intentional and unintentional variations of the azimuth or elevation angle are neglected. Since the sensitivity analysis cannot be carried out within the scope of this work, the result is taken as given in order to be able to demonstrate the methodology. The requirements are thereby reduced in a way that the pixels are to be implemented at fixed positions within the beam pattern.

Considering that the beam patterns of existing lidar sensors are usually not completely homogeneous over the whole vertical and horizontal FoV, it is not sufficient to evaluate only a single pixel of the beam pattern. The lidar used in this work for example, organizes the pixels in multiple horizontal scan lines. For this reason, the position of each pixel is taken into account, resulting in the following parameters to be evaluated:

- Azimuth angle of every pixel $\phi_{\mathrm{P,all}}$
- Elevation angle of every pixel $\theta_{\mathrm{P,all}}$

The fact, that the individual pixels of the real sensor are not clearly assignable with the help of an identification number (ID), poses a major challenge for evaluation. With an ID, the positions of all pixels in the sensor model's beam pattern would be compared with their nominal value (verification) or their respective pixels in the beam pattern of the real sensor (validation). This problem is solved by not considering the individual pixels but whole horizontal scan lines instead. These scan lines are clearly identifiable. With it, this approach brings further challenges, which becomes clear in the following definition of the test cases.

Due to the fact that so far no effects are implemented into the sensor model leading to missing detections, it is possible to iterate through every detection point of a simulated scan line. Provided that the number of pixels per scan line is implemented correctly, allows an unambiguous assignment of the detection points to their respective pixels. For example, simply the fifth detection point of the simulated beam pattern is taken and compared with the fifth pixel's nominal position. Consequently, the first verification test case checks whether the number of pixels per scan line matches the nominal value. A first passed test case leads to the second test case which iterates through all detection points of a scan line and compares the points' azimuth angles ϕ_{DP} and elevation angles θ_{DP} with their pixel's nominal values $\phi_{\mathrm{P,nom}}$ and $\theta_{\mathrm{P,nom}}$. The whole procedure repeats for every scan line and 150 frames. Table 4-1 summarizes the two verification test cases.

Table 4-1: Test cases for beam pattern verification.

Verification Test Case	Tested Parameters
1	Number of Detection Points per Line $n_{\mathrm{DP,line}}$
2	Azimuth / Elevation Angles $\phi_{\mathrm{DP,all}}$ / $\theta_{\mathrm{DP,all}}$

The validation process compares real reference measurement recordings with simulated ones. Since in real recordings, detection points are sometimes missing due to other effects, it is not possible to apply the approach of the verification, which leads to a different approach. The first validation test case is concerned with the elevation angle of the pixels. For this purpose, the detection points of the same scan line are first aggregated over 150 frames. Since a pixel by pixel comparison is not viable, the aggregated detection points of scan line are subdivided into azimuth bins with a fixed width of 0.2°. This is applied to both, the reference and the simulated recording, thereby enabling a bin-wise comparison of the elevation angles of the detection points. The elevation angles of each bin-pair are evaluated with the help of the DVM. The width of 0.2° ensures that enough detection points per bin are available for the calculation of the DVM and thereby minimizes the influence of missing detection points. The whole procedure is repeated for every scan line and multiple reference recordings.

Two test cases are used to validate the azimuth angle of each pixel. In a first step, the number of detection points per scan line is checked by the second validation test case. The DVM of the azimuth resolution $\Delta\phi$ is calculated and reviewed in the third test case. In combination, these two test cases allow for a validation of the pixels' relative azimuth angles. A definitive solution for validating the absolute azimuth angles of the pixels without a pixel ID could not be found. Therefore, only the relative azimuth angle is validated.

The bias of the azimuth resolution helps to determine whether the pixels are accurately positioned in relation to each other in the azimuth direction. If the pixels are correctly distributed over the azimuth FoV is indicated by the model's scattering error. Assuming, the pixels of the real scan line are distributed equally, a simulated line with just two pixels at the correct distance to each other could lead to a low bias and scattering error. Therefore, the number of points per scan line is also taken into account. Table 4-2 gives an overview of the validation test cases.

Table 4-2: Test cases for beam pattern validation.

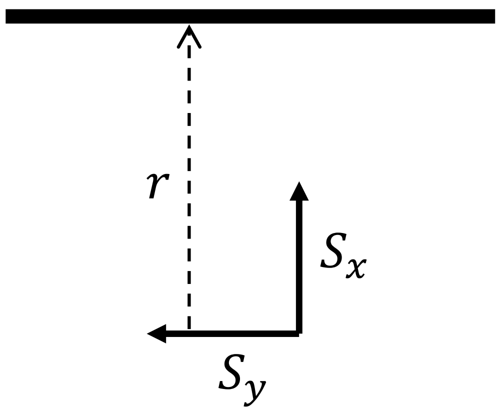
Validation Test Case	Tested Parameters
1	Elevation Angle of every Detection Point $\theta_{\mathrm{DP,all}}$ (With binning)
2	Number of Detection Points per Scan Line $n_{\mathrm{DP,line}}$
3	Azimuth Resolution of Detection Points $\Delta\phi_{\mathrm{DP}}$

In order to demonstrate the iterative approach of the proposed model development methodology shown in Chapter 3, the beam pattern serves as an example. Thus, a total of three iterations are performed, each with a slightly improved implementation together with the respective verification and validation. Specifically, this means that first, a very basic version of the beam pattern is implemented, which is then refined and evolved during the two successive iterations.

4.1.2 Experimental Reference Measurements and Determination of System Parameters

This chapter discusses the experimental setup used to acquire real reference data. Furthermore, it is explained how the system parameters, that are necessary to describe the sensor model's beam pattern, are derived from the reference data.

The most important requirement for the experimental setup is a large homogeneous surface of good reflectivity which ensures that every pixel of the FoV has a high detection probability. This helps minimizing the influence of other unrelated effects. Therefore, the goal of the measurement is to obtain as many detection points per frame as possible. For this reason the reference measurements are carried out in a lecture hall at the Technical University of Darmstadt (TUDa), as it provides a white wall that is usually used as a screen for video projectors. Figure 4-1 depicts the experimental setup (left) and the lecture hall (right). The sensor is positioned at the origin of the coordinate system S and placed at a height of 0.749 m. The sensor is facing the wall of the lecture hall in the direction of the x-axis at a range of r = 3.056 m and the y-axis of the sensor is aligned parallel to the wall. The range is measured with a Bosch GLM 120 C Professional laser range finder.



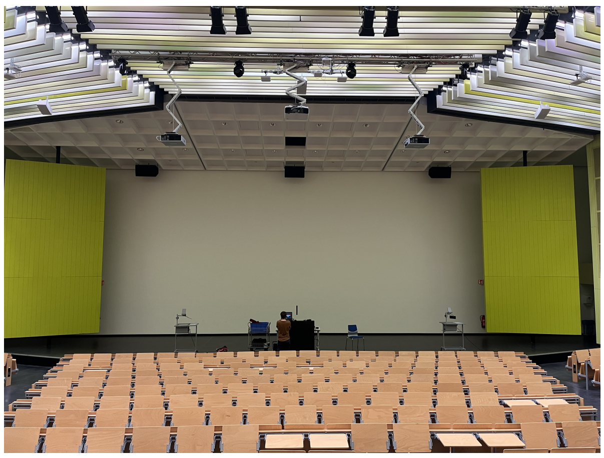


Figure 4-1: The sketch (left) shows the experimental setup of the beam pattern reference measurements. The origin of the coordinate system S indicates the position of the sensor that is placed at a height of 0.749 m above the ground. The range to the wall of the lecture hall (right) is r = 3.056 m.

Since a change of the experimental setup is not necessary, just one measurement with ten recordings of 150 frames each is conducted. The first five recordings are utilized to determine the system parameters that define the sensor model's beam pattern, the remaining five serve the validation. Usually, the uncertainties of the reference measurement devices have to be taken into account for the re-simulation of the experimental setup. This is not possible, since the positions of the detection points are not measured by an external device. Due to the fact that so far no laser intensities or the divergence of the beam are replicated by the sensor model, a deviation of the sensor's position in the simulation does not affect the simulated beam pattern.

In order to be able to implement a realistic beam pattern into the model, system parameters that describe the beam pattern are derived from the first five reference recordings. The elevation angles of the individual scan lines are the first parameters that are determined. For this purpose, all frames of the five recordings are aggregated into a single point cloud. Then, the elevation angles of all points on a single scan line are averaged, which is used to describe the vertical position of the line. Next, the azimuth resolution is deduced by calculating the distances of neighboring points of a single scan line and frame. Repeating this for all scan lines and frames leads to a collection of values for the azimuth resolution. Before determining the average value, possible outliers caused by missing detections are filtered out by deleting all values exceeding a certain threshold. This threshold is estimated with the help of the CDF depicted in Figure 4-2, which gives a impression of what are outliers. The horizontal FoV given by the data sheet, defining the maximum and minimum azimuth angle, is the last parameter needed to describe the beam pattern. The determination of additional parameters, that are necessary to implement a more sophisticated beam pattern, is described in the implementation section of each iteration.

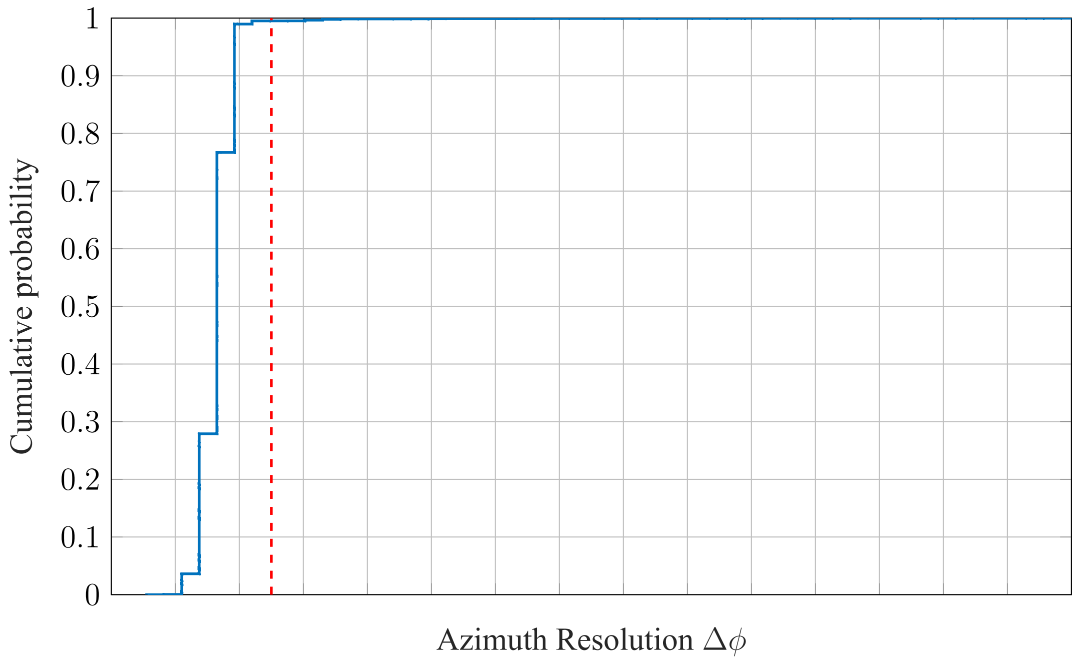


Figure 4-2: CDF of experimentally determined values of the azimuth resolution. Every value is determined by calculating the difference in the azimuth angles of neighboring detection points. The red dotted line indicates the threshold that is used to filter outliers. For reasons of confidentiality, the axis tick labels are missing.

4.1.3 First Iteration of Implementation and V&V of Beam Pattern

In this section, the first iteration of the beam pattern is implemented and the verification and validation processes are executed.

Implementation and Simulated Measurements

The implementation of the beam pattern mainly concerns the transmitter-side signal processing as it determines in which directions the virtual laser is emitted. This is realized by parameterizing the ray tracer. While initializing the model, the position of each pixel in the beam pattern is defined once, and as the beam pattern is considered as temporally constant, it remains unchanged. Based on these fixed positions, the ray tracer samples each pixel with exactly one ray. After the ray tracing process is performed, the range of each ray is assigned to the corresponding pixel. As a result, each detection is clearly defined by an azimuth angle, an elevation angle and the respective range.

In order to save resources such as computing power or development time, it is beneficial to keep the beam pattern as simple as possible. Consequently, for the first iteration, the scan lines of the model are assumed to be perfectly horizontal aiming to pass the test cases with minimum implementation and development effort. Since some of the scan lines of the real sensor are slightly curved on the left-hand side of the FoV, this assumption leads to local deviations of the elevation angle as illustrated in Figure 4-3. The following validation shows whether this simplification is justifiable or not. Due to the basic implementation approach, no additional parameters for the definition of the beam pattern are required.

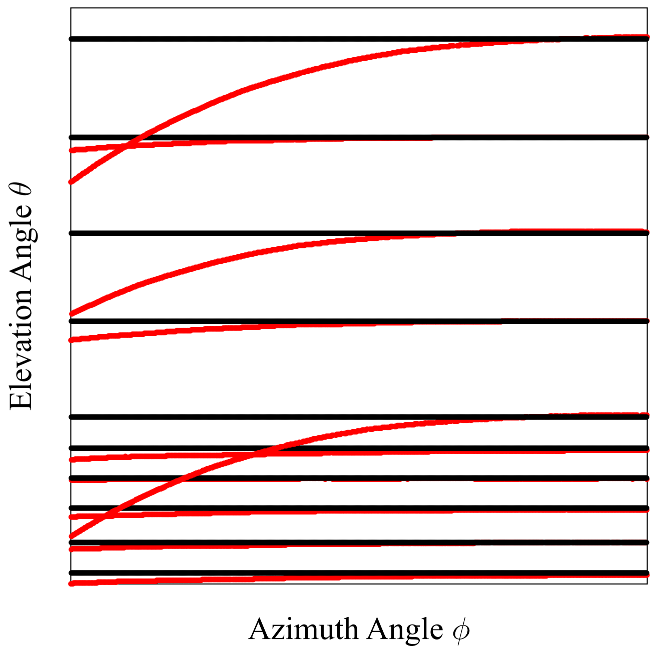


Figure 4-3: Section of the implemented beam pattern (black) compared with the real beam pattern (red). For reasons of confidentiality, the axis tick labels are missing.

To be able to execute the verification and the validation, simulated data is needed which is generated with the help of the developed sensor model after the implementation of the beam pattern. For this purpose, the sensor model is deployed in a virtual scene that replicates the experimental setup of the real reference measurement. Due to the deterministic characteristics of the simulation, the circumstance that no uncertainties are to be considered and the completely static experimental setup, just one measurement with a single recording is necessary. Identical to the ten reference recordings, the simulated recording contains 150 frames.

Verification

Initially, the simulated recording is used to check whether the beam pattern is implemented as intended.

The **first verification test case** verifies the correct number of pixels per scan line. Since the current state of the model ensures that every pixel contains exactly one detection point, the number of detection points per simulated scan line $n_{\mathrm{DP,line,sim}}$ is compared with the aspired number of pixels $n_{\mathrm{P,line,nom}}$. This comparison is conducted for every scan line and repeated for the 150 frames of the simulated recording. For the test cases to be considered as passed, the deviation has to be zero for every scan line and frame. In case of the first beam pattern iteration this condition is fulfilled, which indicates that the first test case is passed.

As the correct number of pixels per line is given, each detection point is compared with its corresponding pixel's position within the **second verification test case**. This is achieved by iterating through every detection point of the simulated recording and calculating the deviation from the respective pixel's azimuth and elevation angle. Figure 4-4 presents the outcome of this procedure. If the beam pattern is implemented correctly, there should only be deviations due to numerical inaccuracies in the calculation of the pixel positions. For this reason, only a small deviation of $0.001°$ is tolerated. Since, this threshold is not exceeded by neither the azimuth nor the elevation angle of any pixel, the second verification test case is considered to be passed.

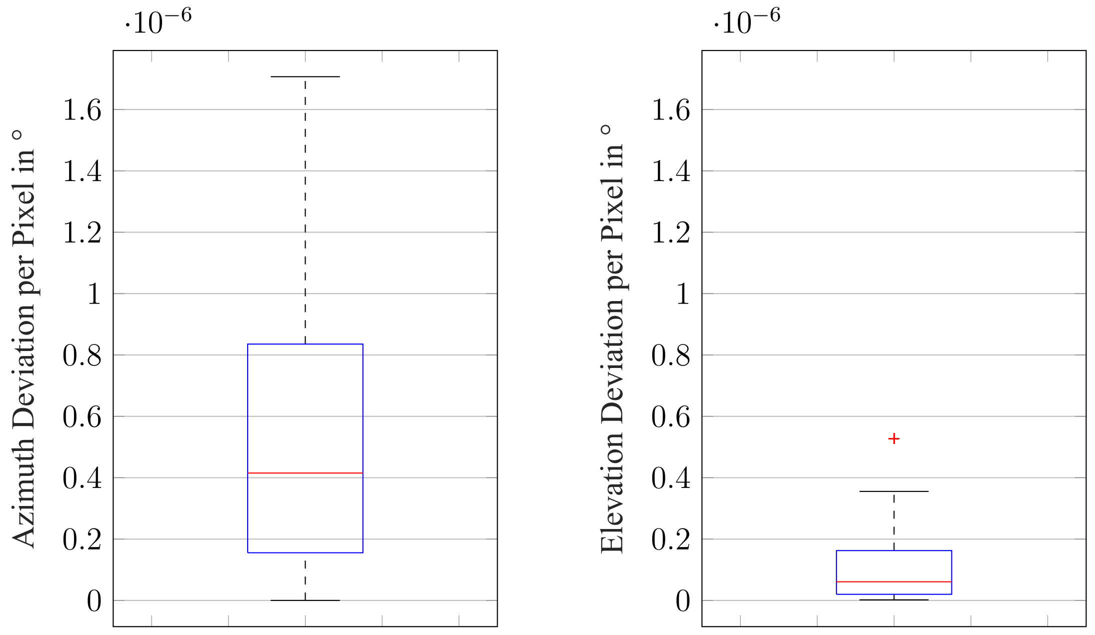


Figure 4-4: Verification result of beam pattern azimuth and elevation angle. Every detection point's azimuth angle ϕ_{DP} and elevation angle θ_{DP} are compared to their corresponding pixel's nominal values $\phi_{\mathrm{P,nom}}$ and $\theta_{\mathrm{P,nom}}$. This is repeated for one simulated recording of 150 frames.

Validation

In the next step, the verified sensor model is validated by comparing the simulated recording to the reference recordings of the real sensor.

The **first validation test case** is concerned with the elevation angle of each pixel. Because of the fact, that no pixel IDs are available, the binning approach introduced in Chapter 4.1.1 is applied. In every bin, the simulated elevation angles are compared to the reference elevation angles and the DVM is calculated. Furthermore, each of the five reference recordings is matched with the single simulated recording one by one leading to five comparisons in total. The summarized result of all bins is depicted in Figure 4-5. As for the pass-fail criterion of this test case, the assumption is made that a detection point shall not be confused with another point of another scan line below or above. Therefore, a threshold value of $\pm 0.075°$ is chosen for d_{bias} and d_{CAVM}, which is equal to the minimum distance of two horizontal scan lines of the real sensor divided by two. The boxplot of the bias shows that this threshold is clearly crossed, which means that the test case is failed. It is noticeable that the values above the threshold are all outliers and the median is close to zero, which indicates that the elevation angles of just a few bins are off. This leads to the assumption that the outliers are caused by the curved scan lines on the left-hand side of the real sensor's FoV (see Figure 4-3). Hence, the aim of the next iteration is to improve the simulated beam pattern at the left edge of the FoV.

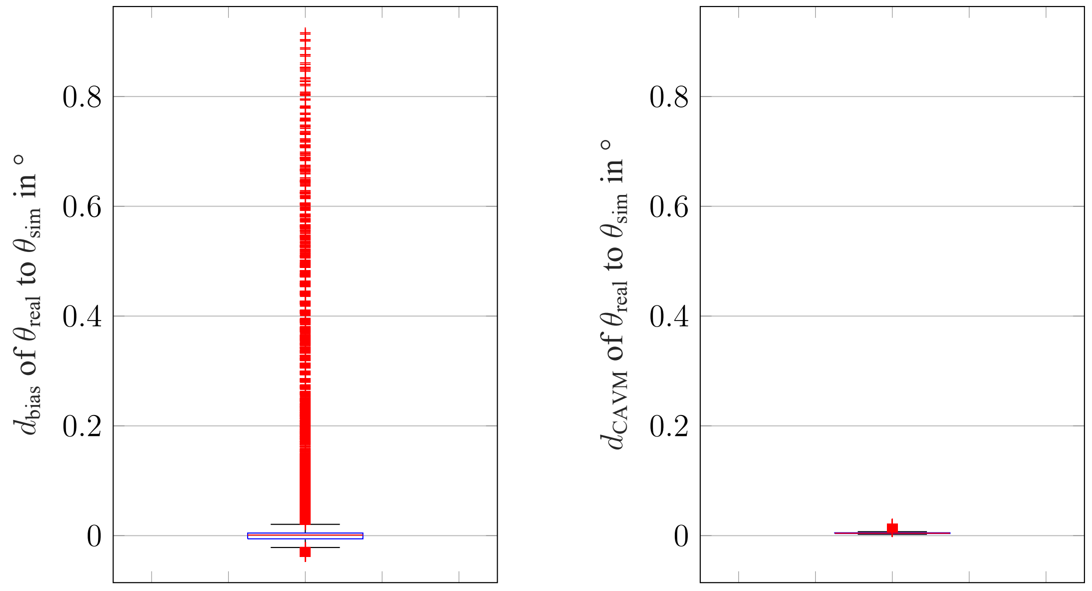


Figure 4-5: Validation result of the beam pattern elevation angle. The boxplots show the biases d_{bias} and scattering errors d_{CAVM} of the elevation angle θ of every azimuth bin. Detection points of five recordings with 150 frames each are compared to a single simulated recording of 150 frames.

For the sake of completeness and to exclude further weak spots, the other two validation test cases are also checked with the current beam pattern. As described in Chapter 4.1.1, the **second validation test case** tests the number of detection points per scan line, which is needed because of no available pixel IDs. First, the number of points of a single scan line of the real reference recording $n_{\text{DP,line,real}}$ is determined and divided by the number of points of the respective scan line of the simulated recording $n_{\text{DP,line,sim}}$. This step is repeated for every line and every frame and again, the five reference recordings are each matched against the single simulated recording. Figure 4-6 summarizes the result. Two conditions are to be met in order to pass the test case. The first condition stipulates that the quotient must not be greater than one and is reasoned by the fact that the simulated scan line contains the maximum number of points per scan line. A quotient exceeding one would mean that the simulated beam pattern is lacking pixels. It is also ensured, that multiple return detections are excluded from the scan lines of the reference recordings. The second condition states that outliers must not be less than 0.9. This allows for missing detection points in the reference recordings that are caused by unrelated effects. Since, both of these conditions are fulfilled, the test case is passed.

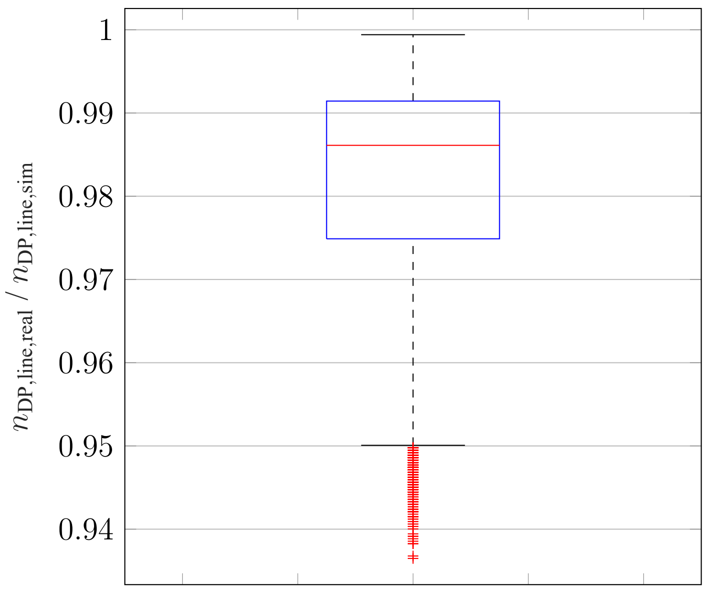


Figure 4-6: Validation result of number of points per scan line. The number of points of a real scan line are divided by the number of points of a simulated scan line. Each frame is processed individually and no frames are aggregated. One simulated recording is compared against five real recordings and each recording contains 150 frames.

Finally, the **third validation test case** addresses the azimuth resolution with the aim of validating the relative pixel position in azimuth direction. To begin with, a single scan line of a single frame is selected and the differences in azimuth angles of adjacent detection points are calculated, leading to $n_{\mathrm{DP,line}} - 1$ resolution values. This is performed on both, the reference recordings and the simulated recording. At this point the first opportunity arises to determine the DVM, as two sets of azimuth resolution values are present. However, one could also repeat the step of calculating the resolution values for multiple frames, still focusing on the same single line. This approach leads to two larger sets of resolution values that are available for the DVM calculation. Even larger sets emerge when resolution values from multiple lines are aggregated, in addition to multiple frames. Thus, there are three different processing stages at which the DVM can be determined. These stages will be further referred to as follows:

- Frame Stage (Single Frame and Single Line)
- Multi-Frame Stage (Multiple Frames and Single Line)
- Recording Stage (Multiple Frames and Multiple Lines)

In total, the same amount of resolution values is available for every approach. Nonetheless, the calculation for the frame stage results in the most values of biases and scattering errors given that the DVM is determined more often on smaller data sets in an early processing stage. Whereas with later stages, the size of the datasets increases resulting in a decreasing amount of biases and scattering errors. It is not yet clear which is the best stage to determine the DVM. For this reason, every approach is conducted to be able to compare the different results. The outcome is presented in Figure 4-7 and shows that the biases and scattering errors are different for every stage. However, it is observed that the maximum absolute bias and scattering error are derived from the frame stage approach. Therefore, the result for the frame stage is used to check against the pass-fail criterion to be able to cover the worst case.

For the pass-fail threshold of this test case it is assumed that a sensitivity analysis is carried out which comes to the conclusion that the azimuth resolution has no significant influence on the SUT. That is why a threshold of $\pm 0.01°$ for maximum d_{bias} and d_{CAVM} is chosen. Since this threshold is not exceeded, the test case is considered as passed.

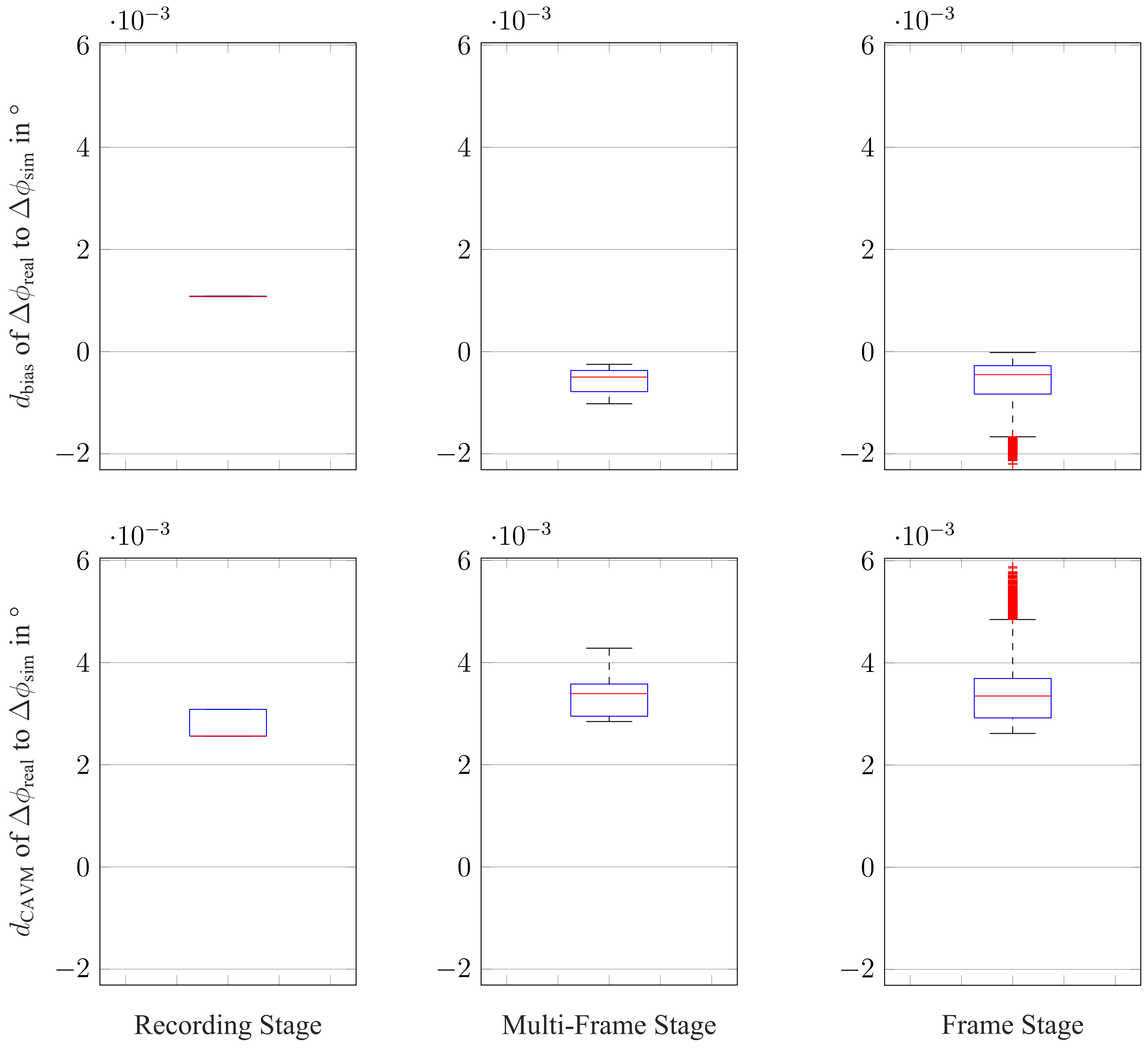


Figure 4-7: Validation result of the beam pattern azimuth resolution $\Delta\phi$. The top row shows the bias d_{bias} for the frame, multi-frame and recording stage. The bottom row depicts the scattering error d_{CAVM} respectively. It is observable that both d_{bias} and d_{CAVM} increase from recording to frame stage. One simulated recording is compared against five real recordings and each recording contains 150 frames.

4.1.4 Second Iteration of Implementation and V&V of Beam Pattern

The following chapter briefly summarizes the second iteration of the beam pattern's implementation, verification and validation.

Implementation and Simulated Measurements

For this iteration, the curved section of the horizontal scan lines on the left edge of the beam pattern is implemented. This modification addresses the issue that leads to the failed validation in the first iteration. To keep the beam pattern as simple as possible it is assumed that all curved scan lines have the same radius of curvature. A polynomial of degree four is used to describe the curvature. Just like the elevation angles of the scan lines, the parameters of the polynomial are determined with the five first reference recordings. For this purpose, the frames of one recording are aggregated, then the polynomial function is fitted to the curved lines and the parameters of every line are averaged. This step is repeated for all five recordings and in the end the emerging parameters are averaged again. These parameters are subsequently used to implement the curvature into the beam pattern which is illustrated in Figure 4-8. As in the previous iteration, one measurement with one recording is captured.

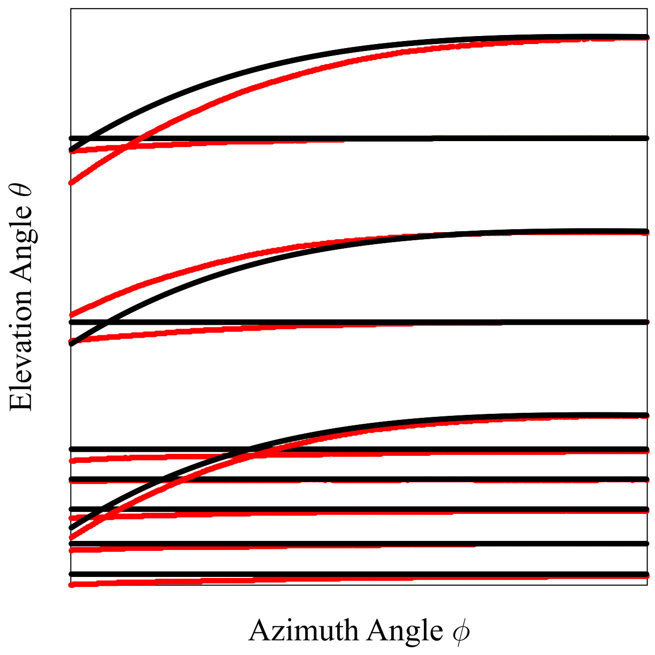


Figure 4-8: Section of the implemented beam pattern (black) compared to the real beam pattern (red). For reasons of confidentiality, the axis tick labels are missing.

Verification

The outcome of the verification is identical to the previous iteration which means that both verification test cases are passed.

Validation

To begin with, the **first validation test case** is retested, which failed in the previous iteration, since the threshold of $\pm 0.075°$ for the bias was exceeded. Figure 4-11 presents the outcome after improving the implementation of the beam pattern. It is evident that the threshold is yet again exceeded with a maximum bias of approximately 0.23°. Due to the fact that again only outliers are above the threshold, it is assumed that the simplification to model the curves with only one radius is incorrect. Therefore, a third iteration is necessary to improve the beam pattern further.

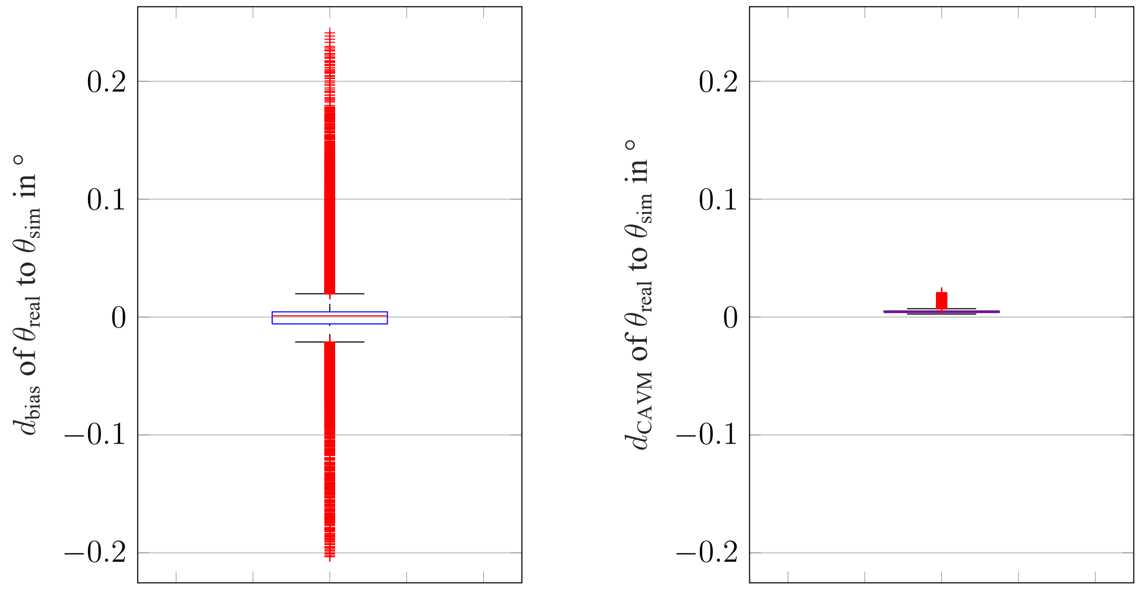


Figure 4-9: Validation result of the beam pattern elevation angle. The boxplots show the biases d_{bias} and scattering errors d_{CAVM} of the elevation angle θ of every azimuth bin. Detection points of five recordings with 150 frames each are compared to a single simulated recording of 150 frames.

Nothing has changed in the results of the other two test cases, as indicated by Figure 4-10 and Figure 4-11. This means that the validation test cases two and three are passed.

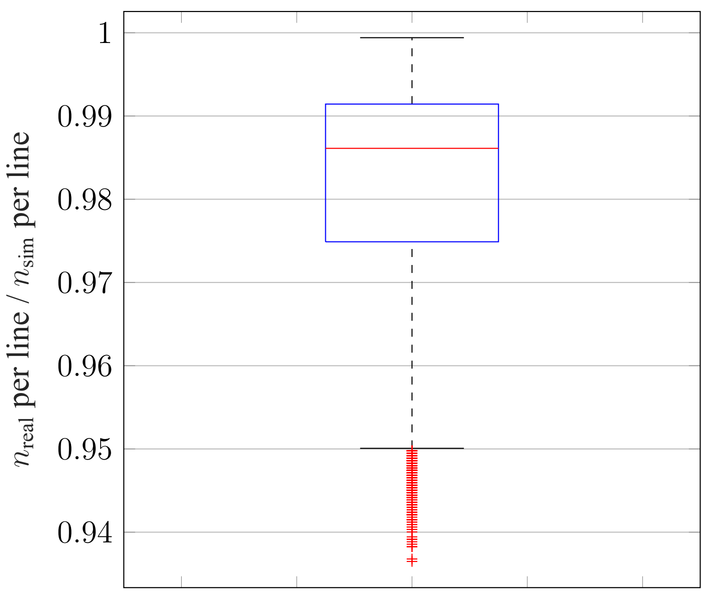
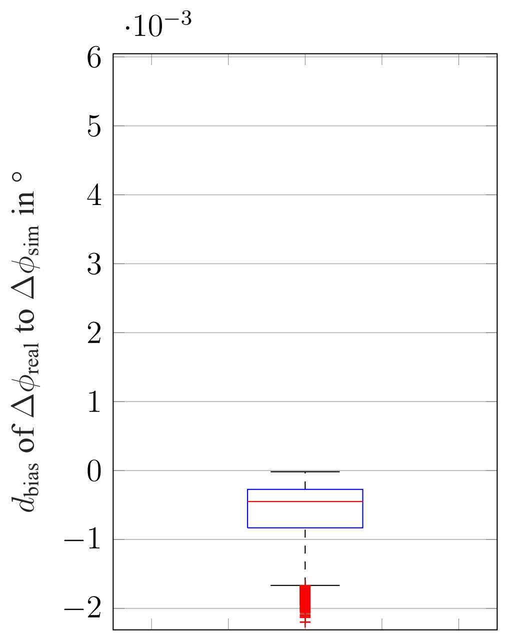


Figure 4-10: Validation result of number of points per scan line. The number of points of a real scan line is divided by the number of points of a simulated scan line. Each frame is processed individually and no frames are aggregated. One simulated recording is compared against five real recordings and each recording contains 150 frames.



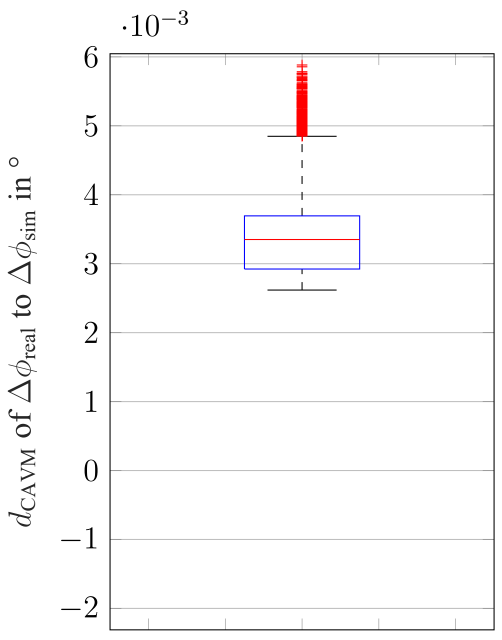


Figure 4-11: Validation result of the beam pattern azimuth resolution $\Delta\phi$. The boxplots show the bias d_{bias} and the scattering error d_{CAVM} of $\Delta\phi$ processed for the frame stage which means that a single frame is used to calculate the DVM for a single line. This step is repeated for every line and every frame and the resulting values are summarized. One simulated recording is compared against five real recordings and each recording contains 150 frames.

4.1.5 Third Iteration of Implementation and V&V of Beam Pattern

The following chapter briefly summarizes the third and last iteration of the beam pattern's implementation, verification and validation.

Implementation and Simulated Measurements

For the third iteration the curves of the horizontal scan lines on the left-hand side of the beam pattern are further refined. This time, the polynomial of degree four is fitted individually to the curvature of each horizontal line. As a result, every horizontal line is defined by an elevation angle and the parameters that describe the curvature. Figure 4-12 depicts the resulting beam pattern on the left edge of the FoV.

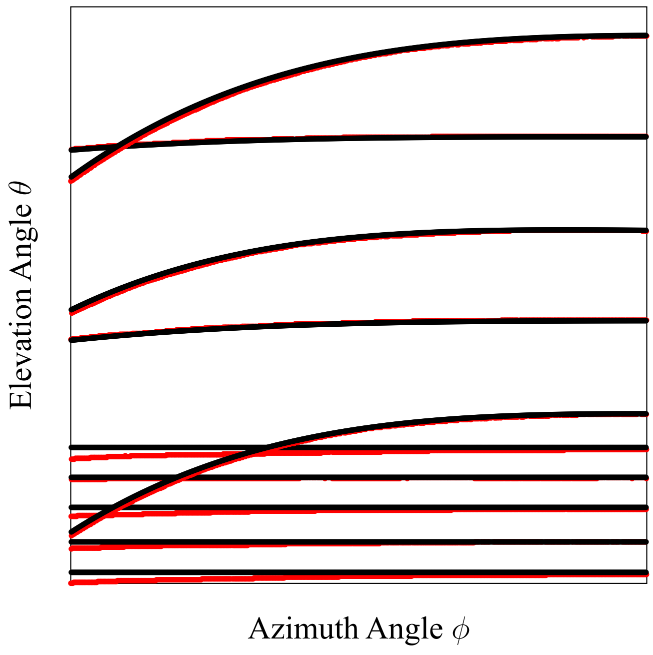


Figure 4-12: Section of the implemented beam pattern (black) compared to the real beam pattern (red). For reasons of confidentiality, the axis tick labels are missing.

Verification

The outcome of the verification is identical to the previous iteration which means that both verification test cases are passed.

Validation

For this iteration, the **first validation test case** is passed since the maximum bias does not exceed the threshold of $\pm 0.075°$ which is proven by Figure 4-13. Additionally, nothing has changed in the results of the other two test cases, as shown in Figure 4-14 and Figure 4-15. Thus the implemented beam pattern is sufficiently close to the beam pattern of the real sensor and is considered valid under the given conditions.

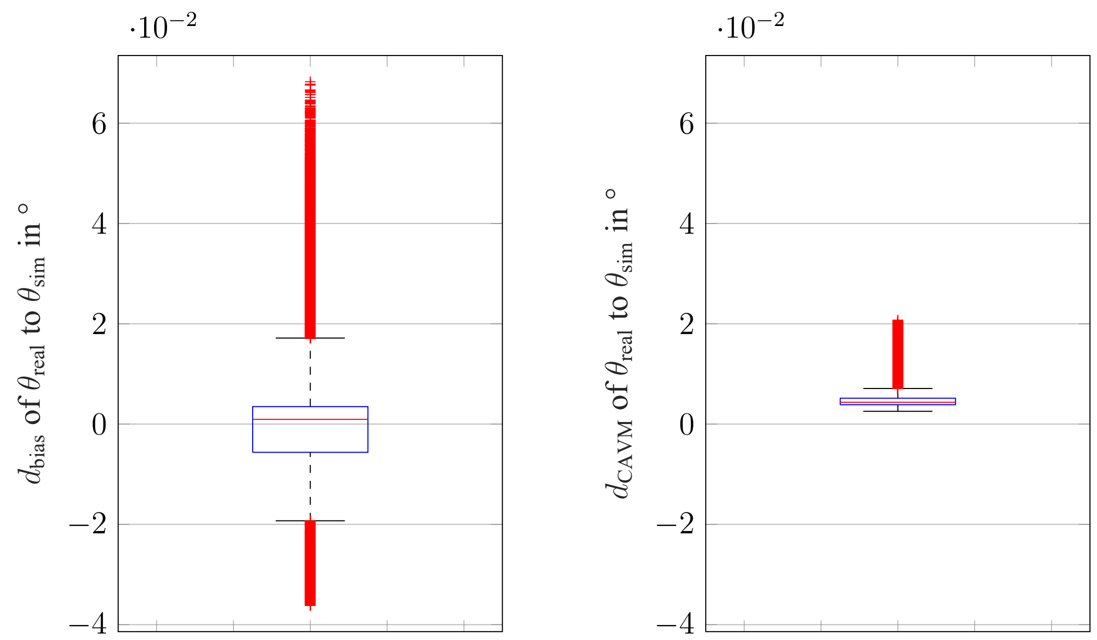


Figure 4-13: Validation result of the beam pattern elevation angle. The boxplots show the biases d_{bias} and scattering errors d_{CAVM} of the elevation angle θ of every azimuth bin. Detection points of five recordings with 150 frames each are compared to a single simulated recording of 150 frames.

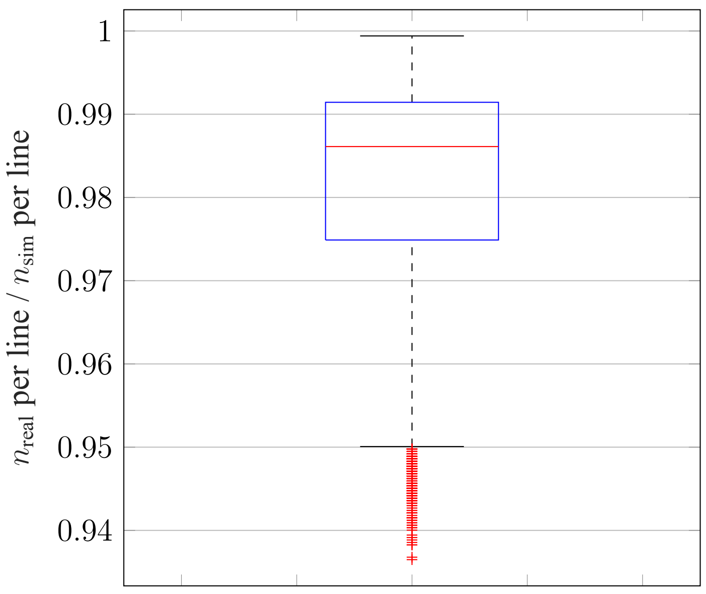


Figure 4-14: Validation result of number of points per scan line. The number of points of a real scan line are divided by the number of points of a simulated scan line. Each frame is processed individually and no frames are aggregated. One simulated recording is compared against five real recordings and each recording contains 150 frames.

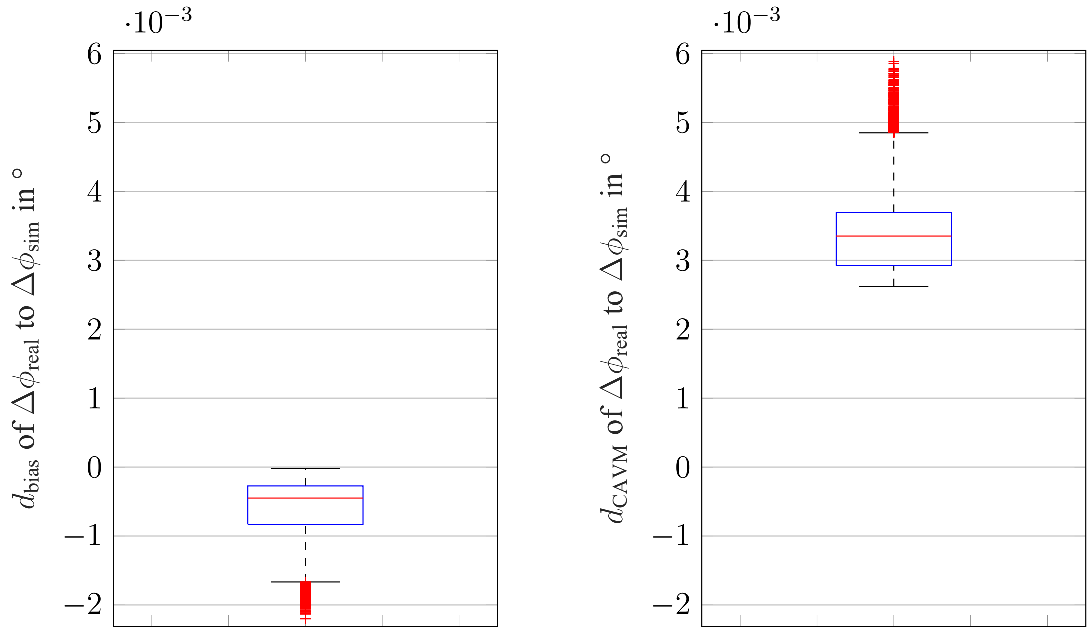


Figure 4-15: Validation result of the beam pattern azimuth resolution $\Delta\phi$. The boxplots show the bias d_{bias} and the scattering error d_{CAVM} of $\Delta\phi$ processed for the frame stage which means that a single frame is used to calculate the DVM for a single line. This step is repeated for every line and every frame and the resulting values are summarized. One simulated recording is compared against five real recordings and each recording contains 150 frames.

4.2 Radial Range

The following chapter covers the implementation, verification and validation of the radial range measuring of the sensor model. At the beginning, the methodology is introduced, requirements are further detailed and test cases are defined. Next, the experimental setup that is used to collect reference measurement data with the real FMCW lidar sensor is described. Part of the data is used to determine the necessary system parameters. Finally, the model's implementation of the range is discussed, verified and validated against reference measurement data.

4.2.1 Methodology and Detailed Requirements with Test Cases

In contrast to the implementation of the beam pattern, the assumption is made, that the range noise and the discretization of the measured range corresponding to a range resolution are important properties that are to be considered in the sensor model. This assumption needs to be proven, for example, by a sensitivity analysis, but in the context of this work, the result of the analysis is taken as given. For this reason, range noise and range resolution are included in the requirements in addition to the range itself.

This results in the following parameters to be checked in test cases as part of the verification and validation of the radial range implementation:

- Range r
- Range noise r_{noise}
- Range resolution Δr

Verification and validation are performed on a single pixel in the beam pattern that has a minimal vertical and horizontal incident angle at the target. The reason for this procedure is that the angle of incidence has an influence on the range noise as it is also shown in the later validation. Since the real sensor has no scan line at exactly 0° elevation angle, the scan line that is closest to that angle is selected, provided the line still has detection points on the target. Since the runway on which the measurements are carried out is slightly sloped, this problem occurs at higher ranges. Due to the alternating azimuth angle of the pixels, the one closest to 0° azimuth angle is chosen. Should the azimuth angle of the chosen pixel deviate from the 0° azimuth angle by a magnitude exceeding the azimuth resolution, the pixel will be considered empty for this frame. Since the sensor measures the radial range to the target, the slight deviation from 0° azimuth and elevation causes a small error compared to the nominal range to the target. These errors are compensated for during the verification process. No deviations arise during validation, as the same pixel is selected for both the real and simulated measurements. This whole process of selecting a single pixel is applied within all test cases in addition to the determinations of system parameters within Chapter 4.2 and Chapter 4.3, and is referred to as single pixel analysis in the further course of this work.

For the verification, the three parameters are compared with their nominal values using the sensor model output of six simulated measurements with three recordings each. Equal to the previous beam pattern verification and validation, every individual simulated recording contains 150 frames. The model is used to generate measurement data of a static scene with a lidar target at different ranges. Based on this data the simulated values of the three parameters are determined. Subsequently, each parameter's result is compared with the nominal value, resulting in three test cases through this approach. These test cases are listed in Table 4-3. The nominal values of r_{noise} and Δr are derived in the next chapter with the help of real sensor reference measurements. The nominal range to the target r_{nom} for the simulated measurements is identical to the range to the target of the reference measurements r_{ref}. Therefore, it is possible to utilize the simulated measurements in both verification and validation.

Table 4-3: Test cases for radial range measuring verification.

Verification Test Case	Tested Parameters
1	Range r
2	Range noise r_{noise}
3	Range resolution Δr

For the validation of the model the simulated measurements are compared to real measurement data that represents the same static scene. A first test case checks whether or not the model is capable of replicating the range noise on angled surfaces. For this purpose, two additional measurements are used in which the target is rotated in relation to the sensor. The second test case is concerned with the accuracy of the models range measurement as well as the range noise on perpendicular surfaces. Both are validated simultaneously for different ranges. Finally, in a third test case, the model's range resolution undergoes assessment. To achieve this, the resolution is computed both in the simulated measurement data and the real measurement data, and then compared. Table 4-4 gives an overview of the three validation test cases. In all validation test cases, the uncertainties of the reference measurement devices are taken into account and the DVM is applied.

Table 4-4: Test cases for radial range measuring validation.

Validation Test Case	Tested Parameters
1	Range noise r_{noise} on angled surfaces
2	Range r and Range noise r_{noise}
3	Range resolution Δr

The test cases and their pass-fail criteria will be discussed in detail later on in this chapter when the verification and validation are performed.

4.2.2 Experimental Reference Measurements and Determination of System Parameters

First, the experimental setup of the real reference measurements is presented. Subsequently, it is explained how the missing system parameters, in this case r_{noise} and Δr, are determined using the reference data. The goal of the experimental measurement is to obtain range values of a static target at different ranges and with different target rotation angles.

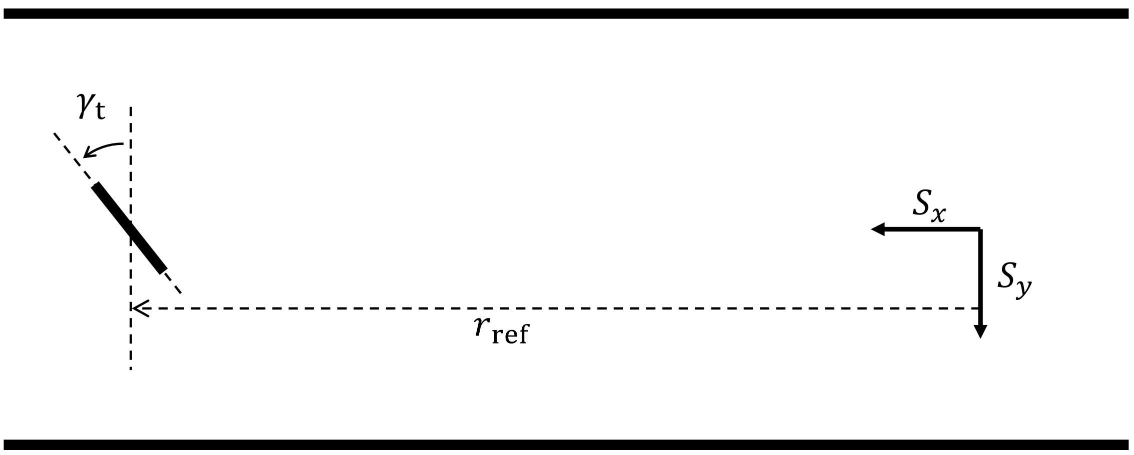


Figure 4-16: The sketch shows the experimental setup of the range reference measurements. The sensor is mounted on the origin of the S coordinate system at a height of 0.765 m above the surface. γ_t is the angle which indicates the rotation of the target and r_{ref} is the range of to the target.

Reference measurements are conducted with the experimental setup described in Figure 4-16 on the runway of the August Euler Airfield near Darmstadt. The origin of the coordinate system S indicates the position of the real sensor and its orientation facing the target in x-direction. The left image of Figure 4-17 shows the sensor test stand on which the sensor is mounted at a height of 0.765 m above the asphalt surface as well as the runway on which the measurements are performed. The target is placed at a range r_{ref} and rotated by an angle γ_t. In case of γ_t being equal to zero, the surface of the target is perpendicular to the x-axis of the sensor. Furthermore, the target is positioned in such a way that the x-axis of the sensor points to the target's center. The target's surface has a reflectivity of 80 % and is 1 m in height as well as 1 m in width. It is located directly above the asphalt surface as shown in the center image of Figure 4-17. The weather on the day of the measurement was alternately sunny and cloudy with temperatures above 20 °C.

The whole sensor test stand is leveled with help of a spirit level. Further, the sensor test stand and the target are aligned to each other based on a long measuring tape that is rolled out. The reference range r_{ref} to the target is measured with a Bosch GLM 120 C Professional laser range finder that is visible in the right image of Figure 4-17. A total of eight measurements with different reference ranges and target rotations are performed which are listed in Table 4-5. The reference ranges of measurement number 2 and 3 are just a estimated values as it is difficult to precisely determine the range to an angled target with the existing setup. However, this does not cause any problems, since the measurements are only used to investigate the influence of angled surfaces on the range noise. A measurement uncertainty of ± 5 mm is assumed for the reference range r_{ref}. This is based on the uncertainty of the laser range finder which is specified with ± 1.5 mm and the fact that both, the range of the sensor test stand to the target and the position of the sensor on the test stand, are measured with the range finder. This adds up to a uncertainty of ± 3 mm. The additional ± 2 mm compensate for not hitting the target perfectly perpendicular with the laser range finder. For the rotation angle of the target, a pessimistic uncertainty of $\pm 3°$ is assumed as the angle is measured with a simple protractor. To avoid confusion, it is important to clarify that the index ref refers to the reference measuring device, in this case the laser range finder, and index real relates to values measured by the real lidar sensor.

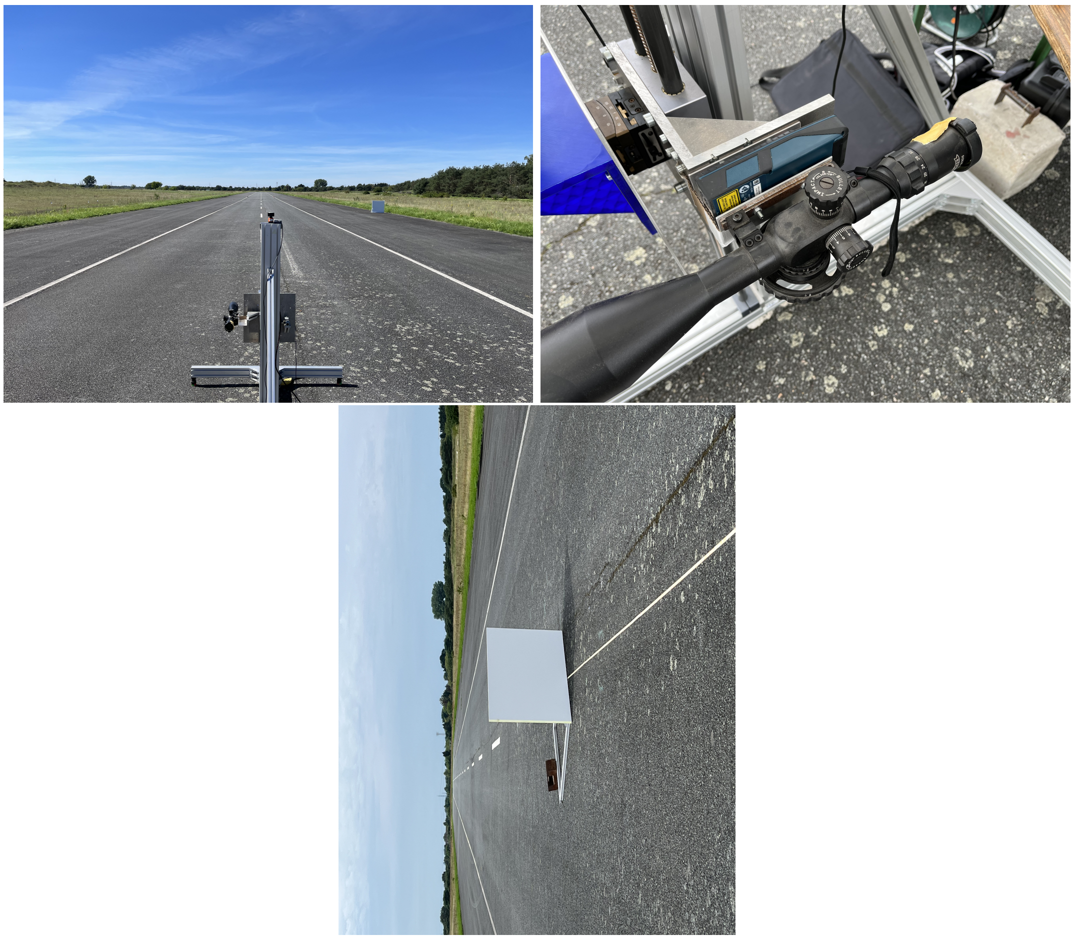


Figure 4-17: Sensor test stand on the runway (left), range finder mounted on the sensor test stand (right) and lidar target with 80 % reflectivity.

Table 4-5: Experimental reference measurements for radial range. Each measurement includes 16 recordings. The numbering of the reference recordings follows the scheme: <*Measurement Number*>.<*Recording Number*>.

Ref. No.	r_{ref} in m	γ_{t} in °
1.1-16	3.815 $\pm$0.005	0 $\pm$3
2.1-16	$\sim$3.815	70 $\pm$3
3.1-16	$\sim$3.815	80 $\pm$3
4.1-16	7.869 $\pm$0.005	0 $\pm$3
5.1-16	19.943 $\pm$0.005	0 $\pm$3
6.1-16	30.173 $\pm$0.005	0 $\pm$3
7.1-16	40.823 $\pm$0.005	0 $\pm$3
8.1-16	58.867 $\pm$0.005	0 $\pm$3

Every measurement contains 16 recordings with 150 frames each. In general, the first eight recordings are utilized to determine the system parameters range noise r_{noise} and range resolution Δr for the sensor model while the other eight recordings are used to validate the model. Furthermore, the reference range r_{ref} serves as r_{nom} for the verification process, since r_{ref} is directly transferred into the simulated measurements.

In a first (and only) iteration of the range implementation, it is assumed that the range noise follows a normal distribution in order to keep the modeling of the noise as simple as possible. For this purpose, the standard deviation of the range r_{stdv} is determined based on the reference measurements 1, 4, 5, 6, 7 and 8 and calculated with the first eight recordings of each measurement. To achieve this, for every of the 150 frames of a recording, the range r of a single pixel is taken with the aforementioned method of maintaining a minimal vertical and horizontal incident angle at the target (single pixel analysis). The standard deviation r_{stdv} is computed per recording based on these obtained range values. Finally, the results for all recordings are averaged leading to a range independent description of the noise.

The range resolution Δr is determined in a similar way. Instead of calculating the standard deviation with the obtained values of one recording, the range values are clustered with a DBSCAN-algorithm. Before that, range values outside the range of $\pm r_{\text{stdv}}$ are filtered out to minimize the influence of outliers. The range of each cluster is then averaged and the range differences between neighboring clusters are taken as samples for the range resolution. Again, all obtained range resolution samples are averaged.

4.2.3 Implementation and V&V of Radial Range

In the following, the implementation, verification and validation are performed step-by-step and the results are presented.

Implementation and Simulated Measurements

As described earlier, each pixel is sampled by exactly one ray by the ray tracer. This means that for the model's range measurement no further adjustments of the transmitter-side signal processing are necessary. However, on the receiver-side signal processing, the range noise and range resolution are to be implemented.

To model the range resolution, the range output of the ray tracer r_{rt} is divided by the range resolution Δr. Then the integer quotient of this division is again multiplied by the range resolution Δr to receive the discretized simulated range r_{sim}. The range noise could be modeled in a similar way by applying normally distributed noise to the range output of the ray tracer r_{rt} before the descriticization. But due to performance reasons, the normally distributed range noise is applied to the individual rays during the ray tracing process instead. The noise follows the previously determined range standard deviation r_{stdv}. That means, the output of the ray tracer already includes the intended noise and no further processing is needed.

The adapted model is then deployed in a virtual scene that replicates the real experimental setup of the range measurements. The measurements are re-simulated by positioning a model target of the same size as the real target at the respective reference ranges and target angles. Table 4-6 provides an overview of all simulated measurements, their individual recordings, and the corresponding test cases where the recordings find application. Furthermore, the table shows that the uncertainties of the reference range and target angle are also simulated, which means that they are taken into account in the results of the validation.

Table 4-6: Overview of all simulated radial range measurements. The right side of the table indicates which recordings are applied in which test cases. The numbering of the simulated recordings follows the scheme: *<Measurement Number>.<Recording Number>*.

Sim. No.	r_{ref} in m	γ_{t} in °	Verification TC			Validation TC		
			1	2	3	1	2	3
1.1	3.815	0	X	X	X	X	X	X
1.2	3.815	0 + 3				X		
1.3	3.815	0 - 3				X		
1.4	3.815 + 0.005	0	X	X	X		X	X
1.5	3.815 - 0.005	0	X	X	X		X	X
2.1	~3.815	70				X		
2.2	~3.815	70 + 3				X		
2.3	~3.815	70 - 3				X		
3.1	~3.815	80				X		
3.2	~3.815	80 + 3				X		
3.3	~3.815	80 - 3				X		
4.1	7.869	0	X	X	X		X	X
4.2	7.869 + 0.005	0	X	X	X		X	X
4.3	7.869 - 0.005	0	X	X	X		X	X
5.1	19.943	0	X	X	X		X	X
5.2	19.943 + 0.005	0	X	X	X		X	X
5.3	19.943 - 0.005	0	X	X	X		X	X
6.1	30.173	0	X	X	X			
6.2	30.173 + 0.005	0	X	X	X		X	X
6.3	30.173 - 0.005	0	X	X	X		X	X
7.1	40.823	0	X	X	X		X	X
7.2	40.823 + 0.005	0	X	X	X		X	X
7.3	40.823 - 0.005	0	X	X	X		X	X
8.1	58.867	0	X	X	X		X	X
8.2	58.867 + 0.005	0	X	X	X		X	X
8.3	58.867 - 0.005	0	X	X	X		X	X

Verification

Before the simulated data is compared with the real reference data in the validation, it is first verified that the output of the sensor model replicates the intended behavior. The previously determined system parameters r_{stdv} and Δr, that are implemented into model, define the intended behavior and serve as the nominal values $r_{\mathrm{stdv,nom}}$ and Δr_{nom} for the verification process.

The **first verification test case** is concerned with the range r. To break it down, the lidar target is placed at a certain nominal range r_{nom} (which is identical to r_{ref}) in the virtual scene. Then the simulation is run and the range r_{sim}, measured by the sensor model, is compared to the nominal value. As it is more difficult to determine the exact range of angled targets, only those recordings are considered where the target is not rotated in any way (see Table 4-6). Consequently, 18 recordings with 150 frames each are available for the first verification test case.

In a first step, the single pixel analysis is applied to every frame and the respective radial ranges r_{sim} are collected. Due to the non-perpendicular incident angle on the target, the nominal range is corrected for every simulated range taking into account the actual incident angle. Thereafter the difference between the simulated ranges r_{sim} and the corrected nominal ranges $r_{\text{nom,c}}$ is calculated. The result is visualized in Figure 4-18. The pass-fail criterion of this test case is defined as whether or not the difference of r_{sim} and $r_{\text{nom,c}}$ is justified by the implemented range noise r_{stdv}. For reasons of confidentiality the exact threshold value cannot be disclosed. However, the range difference falls below this threshold, indicating a passed first verification test case.

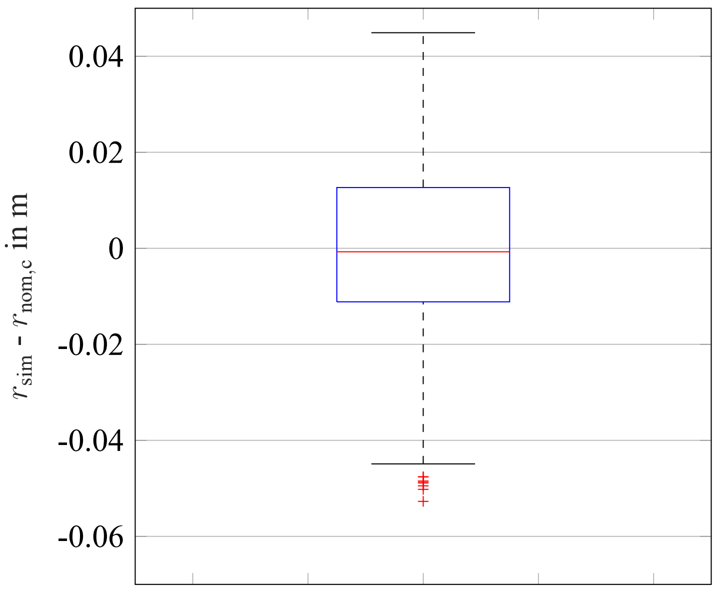


Figure 4-18: Verification result of the simulated range r_{sim}. For the difference of the simulated range r_{sim} and the corrected nominal range $r_{\text{nom,c}}$ 18 recordings with 150 frames each are evaluated.

The correct implementation of the range noise r_{noise} is evaluated in the **second verification test case**, which is based on the exact same recordings as the previous test case. For every recording, the simulated range standard deviation $r_{\text{stdv,sim}}$ is compared to the nominal range standard deviation $r_{\text{stdv,nom}}$ that is implemented into the sensor model. The range values of 150 frames are gathered with the single pixel analysis for every recording. Then $r_{\text{stdv,sim}}$ is calculated for each of the 18 recordings and deducted by $r_{\text{stdv,nom}}$. Figure 4-19 illustrates the outcome. It is noticeable that the standard deviation of the simulated ranges is always smaller than the nominal standard deviation. This is caused by the range resolution and the concomitant discretization of the range.

Furthermore, the deviation of different recordings is relatively large, considering the maximum value of the boxplot is three times greater than the minimum value. This means that statistical factors still have an impact on the outcome despite the 150 frames. To take statistical deviations and the influence of the discretization into account, a threshold of ± 0.001 m is set for the difference of $r_{\text{stdv,nom}}$ and $r_{\text{stdv,sim}}$. Since this threshold is not exceeded, the test case is considered to be passed.

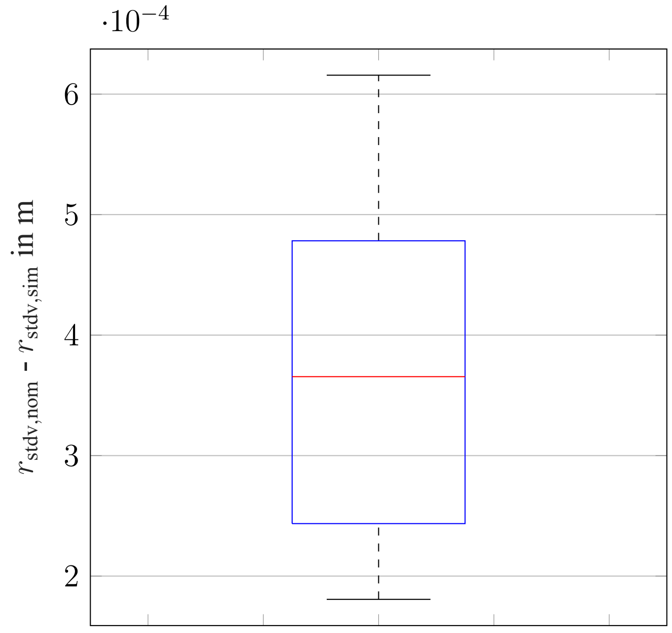


Figure 4-19: Verification result of the simulated range noise $r_{\text{noise,sim}}$. Six measurements of three recordings with 150 frames each are used to compute the difference of the nominal standard deviation $r_{\text{stdv,nom}}$ and the standard deviation of the simulated recordings $r_{\text{stdv,sim}}$.

The **third verification test case** reviews the range resolution Δr. Again, the same recordings of the previous two test cases are utilized. After accumulating the range values for one recording with the single pixel analysis, the range resolution Δr_{sim} is determined with the aforementioned method of clustering the data and taking the differences of neighboring clusters as samples of the range resolution. Subsequently, the samples are checked against the nominal range resolution Δr_{nom} which is implemented into the model. Since the simulated range resolution values are to the sixteenth decimal place equal to the nominal value, the test case is considered to be passed.

Validation

Following the verification, the sensor model's simulation output is assessed in comparison to the real sensor's reference measurements within the validation of the model. As in the validation of the beam pattern, the DVM is the metric of choice for all three range validation test cases. Since the first eight recordings of each reference measurement are utilized to determine the system parameters r_{stdv} and Δr, the remaining eight recordings per measurement are available for the validation (see Table 4-5). Table 4-6 lists the simulated recordings and the corresponding test cases where they are utilized.

The **first validation test case** investigates the influence of angled surfaces on the sensor's range noise r_{noise} and provides information about whether or not the model is able to replicate the real sensors range noise. As this question addresses the model scattering error rather than the bias, the focus lies on the resulting d_{CAVM}.

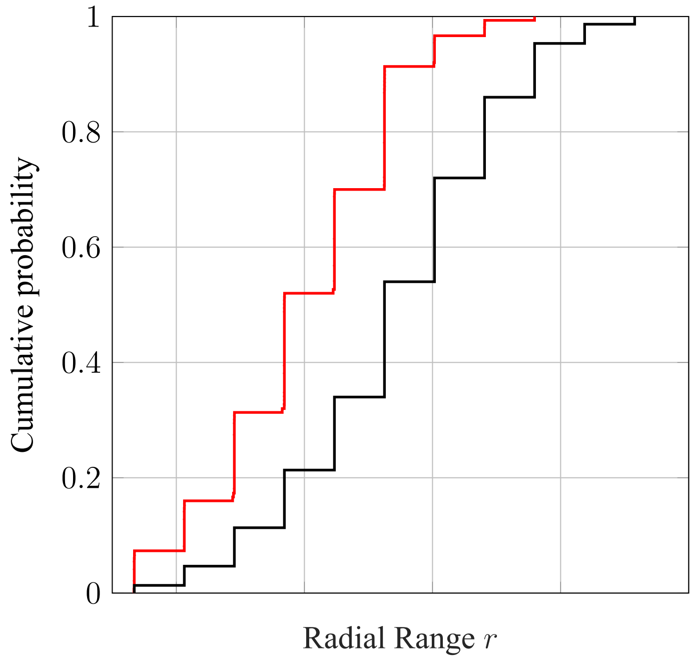
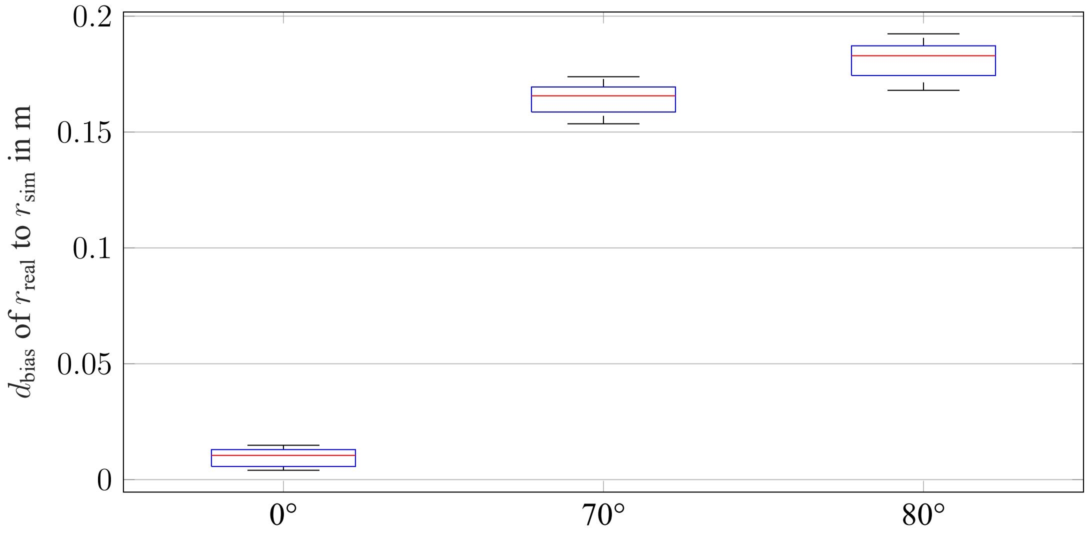


Figure 4-20: Exemplary EDF of range values of a static target. The EDF illustrates the distribution of the real reference range values r_{real} (red) in comparison to the simulated range values r_{sim} (black). Both compared sets contain 150 range values each. For reasons of confidentiality, the x-axis tick labels are missing.

The validation procedure is as follows: The three reference measurements (Ref. No. 1.9-16, 2.9-16, 3.9-16) with target angles 0°, 70° and 80° are each compared to their respective simulated measurements (Sim. No. 1.1-3, 2.1-3, 3.1-3). In doing so, each of the eight reference recordings is tested against all three simulated recordings, leading to 24 comparisons per measurement in total. As a first step of every comparison, the range values r_{real} and r_{sim} are gathered for the real reference and the simulated recording using the single pixel analysis. Then, with these two sets of range values the DVM is calculated. Figure 4-20 shows an example of how the two sets are distributed. In conclusion 24 values of d_{bias} and d_{CAVM} are obtained per measurement. The result is summarized in Figure 4-21.

The two boxplots indicate that both d_{bias} and d_{CAVM} are increasing with higher target angles. As the aim of this test case is the range noise and not the range itself, the bias is not considered any further. However, the large bias of almost 0.2 m underlines the problem of measuring the range of angled targets with the existing experimental setup. Since the requirement for the implementation of the range noise originates from the assumption of a sensitivity analysis, the pass-fail criterion is also assumed to be a result of this analysis. Thereby a pass-fail threshold for the scattering error d_{CAVM} is set at 5 cm. As a result, the test cases are considered to be passed, which means that the model is able to replicate the noise behavior of the real sensor for the investigated target angles. However, it is questionable whether this is valid for even larger angles, since a noticeable increase in noise is observed for the scan lines that hit the asphalt surface at a very flat angle. This observation is discussed again as part of the assessment in Chapter 5.



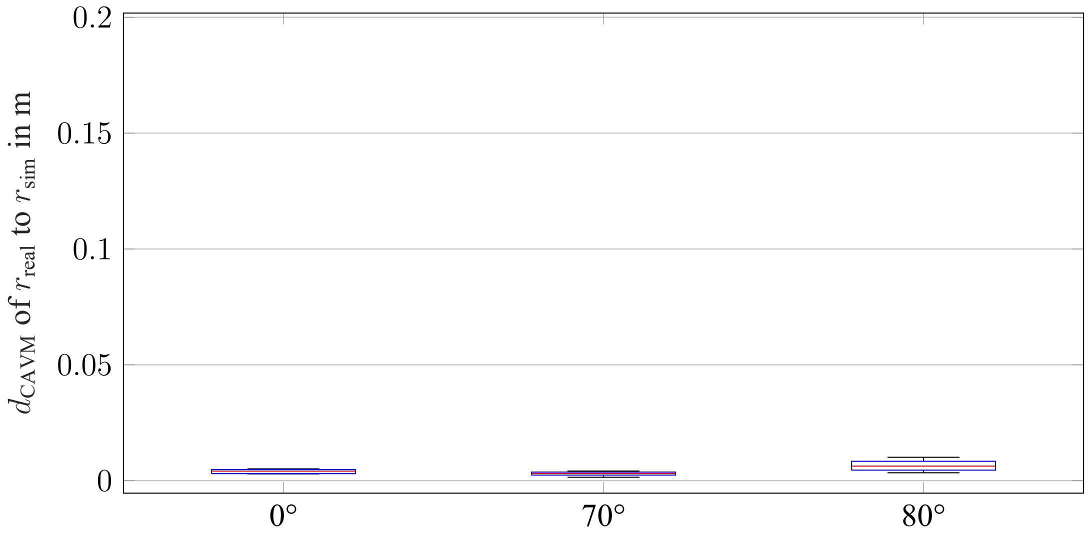


Figure 4-21: The boxplots show the model bias d_{bias} (top) and the model scattering error d_{CAVM} (bottom) of the range measuring for three different target angles. Given that 24 comparisons are performed for each measurement, every box contains 24 biases or scattering errors.

In addition to the range noise, the **second validation test case** covers the range itself. Except for the measurements used, the procedure is exactly the same as in the previous test cases. This time, the lidar target is not angled in any way and the aim is to investigate the sensor model bias and scattering error for varying ranges. Therefore, the six real reference measurements 1, 4, 5, 6, 7, 8 are compared to the six respective simulated measurements. Again, for each measurement 24 values of d_{bias} and d_{CAVM} are calculated. The summarized result is presented in Figure 4-22. The boxplot of the d_{CAVM} demonstrates that the threshold of 5 cm is not exceeded at any range and is thereby considered as passed regarding the scattering error. The threshold for the bias is set to be 1 % of the reference range and the absolute value is thereby depending on the range. This is based on the ODD defined in Chapter 3 which envisages the sensor to be used in a traffic jam pilot. Thus, at greater ranges, the system has more time to react to e. g.

an obstacle, which is why greater absolute deviations of the model are tolerated. At close ranges however, the model's accuracy should be increased. As the top boxplot of Figure 4-22 shows, the threshold of 1 % is not surpassed at any range. This means that the whole test case is passed.

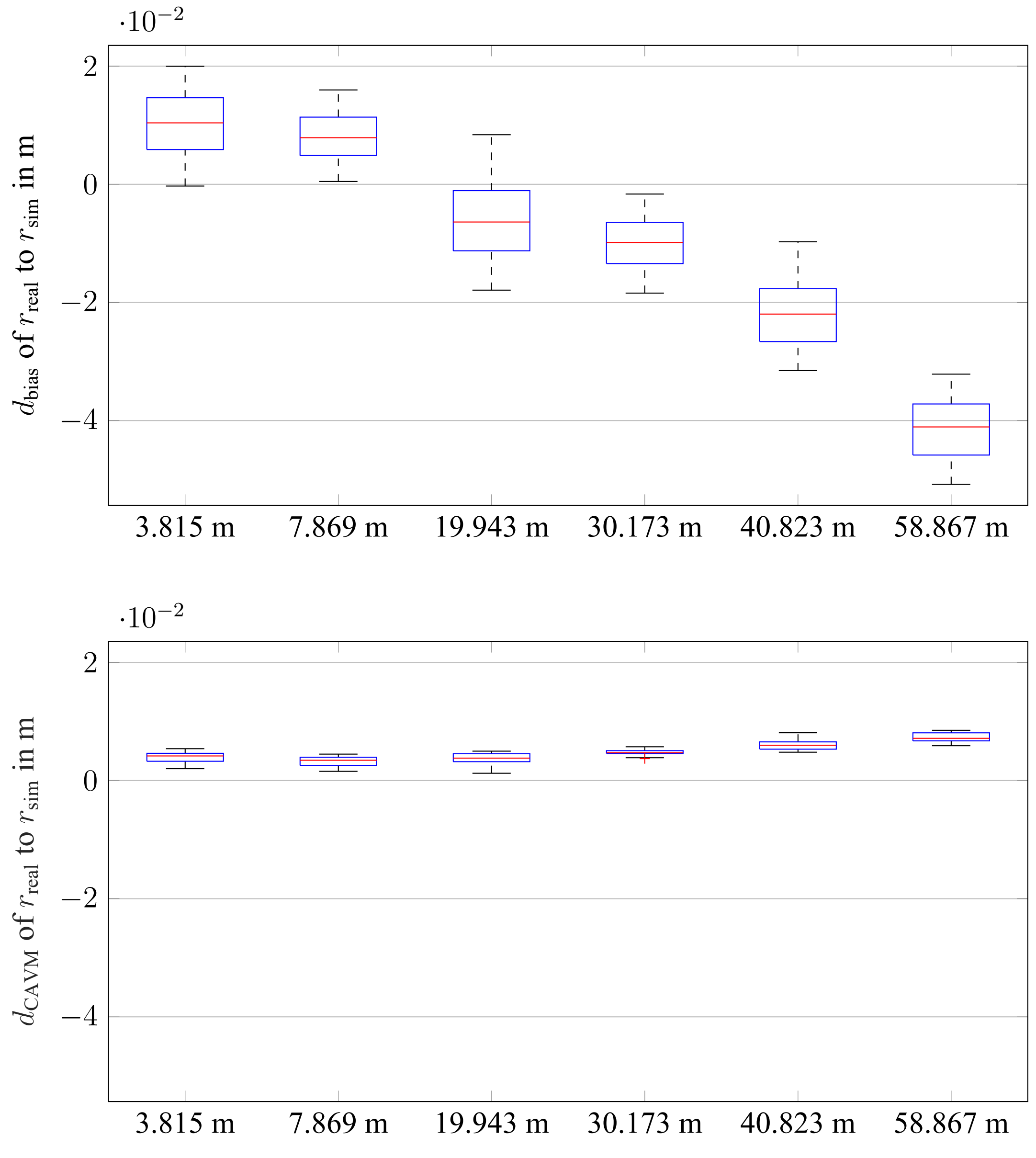
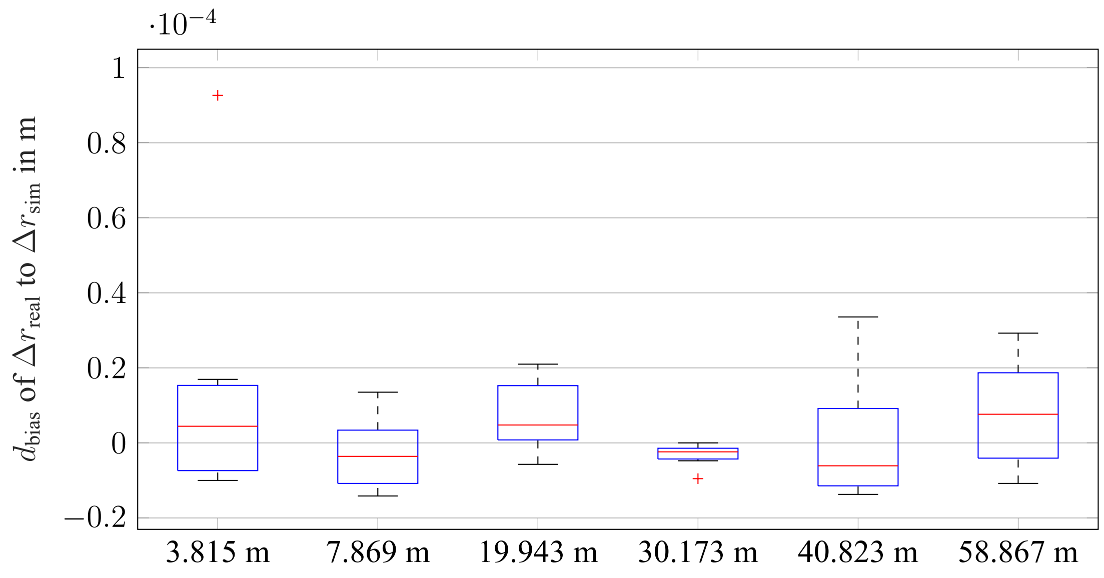


Figure 4-22: The boxplots show the model bias d_{bias} (top) and the model scattering error d_{CAVM} (bottom) of the range measuring for six different target ranges. Given that 24 comparisons are performed for each measurement, every box contains 24 biases or scattering errors.

Finally, the range resolution Δr is resolved by the **third validation test case**. It is based on the same measurements as the second validation test case. Moreover, the procedure is also adopted and an additional step is added. Following the acquisition of range values, they are clustered, and samples of the range resolution are derived by calculating the range difference between adjacent clusters. Preceding this, those values that lie outside the range of the standard deviation are filtered out in order to minimize the influence of outliers, especially in the real reference measurements. This way, samples of the range resolution are gathered for the reference recording as well as the simulated recording. The DVM is then used to compare the two sample sets. After repeating this step for every recording and measurement, the result presented in Figure 4-23 is acquired. Similar to the range noise, the requirement for the implementation of the range resolution originates from the assumed sensitivity analysis, and so does the pass-fail threshold. It is estimated that the impact of the range resolution on an object detector is less significant than that of the range noise. For this reason, a pass-fail threshold of 10 cm is selected for both, the bias and the scattering error. The boxplots indicated that this threshold is not exceeded and the test case is considered to be passed.



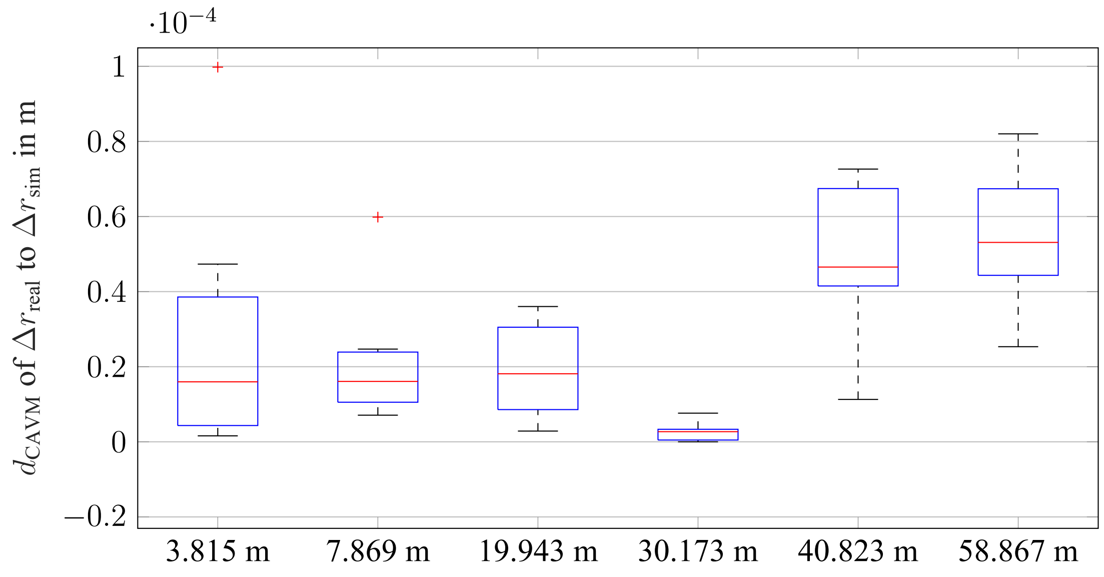


Figure 4-23: The boxplots show the model bias d_{bias} (top) and the model scattering error d_{CAVM} (bottom) of the range resolution for six different target ranges. Given that 24 comparisons are performed for each measurement, every box contains 24 biases or scattering errors.

4.3 Relative Radial Velocity

The upcoming chapter discusses the direct relative radial velocity measurement of the sensor model which is a distinctive feature of FMCW lidar sensors. At first the general methodology, the requirements and the test cases for verification and validation are introduced. Following that, the experimental setup of the conducted dynamic reference measurements is explained and the necessary system parameters are derived. The last section covers the implementation of the relative radial velocity into the model and its verification and validation.

4.3.1 Methodology and Detailed Requirements with Test Cases

In principle, the radial velocity is quite similar to the radial range. Again, it is assumed that the noise and resolution of the measured velocity are relevant properties of the real sensor, that are also be taken into account in the corresponding sensor model. For the purpose of this work, this hypothesis is assumed to be proven by a sensitivity analysis. Consequently, the velocity noise and resolution pose as additional requirements that are to be considered.

Accordingly, the following parameters are to be implemented into the sensor model and verified and validated with test cases:

- Radial velocity $\dot{r}$
- Radial velocity noise $\dot{r}_{\text{noise}}$
- Radial velocity resolution $\Delta\dot{r}$

Exactly as with the radial range, only a single pixel from the beam pattern is considered for the determination of the system parameters, the verification and the validation. This is important as it allows an analysis of the radial velocity over several frames and the influence of a non-perpendicular incident angle is minimized. Practically, the pixel of interest is determined with the single pixel analysis described in Chapter 4.2.

In the iteration of the velocity implementation presented in this work, some simplifications are made that allow for the simplest possible case of testing the radial velocity. As explained in more detail in the following chapter, the velocity of a target vehicle moving away from the sensor in a linear motion is examined. This means that the vehicle is assumed to be moving exactly alongside the x-axis of the sensor. Since, for a human driver, it is not possible to keep the vehicle in a perfect straight line at all times, small deviations of the measured radial velocity occur. For the following implementation, verification and validation, any influences on the velocity due to e. g. rotation of the target vehicle are neglected and only the pure linear motion is assumed. When looking at the ODD, it is apparent that a linear motion of the vehicle driving ahead is the most common scenario a traffic jam pilot will be confronted with. In case one wants to include the induced velocity by the vehicle's rotation, it would have to be validated first, which is not possible with the available experimental setup.

Next up, the verification and validation test cases are introduced. The verification test cases are similar to those of the range, with the difference that no specific velocity resolution is implemented and thus no respective test case is needed. The reason for this is explained later on. To be able to verify the correct implementation of the velocity and the velocity noise, the sensor model is used to generate simulated data. As with the real reference measurements, a vehicle moving linearly away from the virtual sensor at different velocities is simulated. The radial velocity $\dot{r}$ is covered by the first verification test case and the second one addresses the radial velocity noise $\dot{r}_{\text{noise}}$. In both test cases it is checked whether or not the parameters are equal to their nominal values. Table 4-7 lists the verification test cases.

Table 4-7: Test cases for radial velocity measuring verification.

Verification Test Case	Tested Parameters
1	Velocity $\dot{r}$
2	Velocity noise $\dot{r}_{\text{noise}}$

In a recurrent manner, for the validation of the velocity, real reference measurements are compared to simulated ones which are generated with the sensor model. In order to be able to re-simulate the real reference measurement in a virtual environment, the dynamics of the real vehicle are measured by an Inertial Measurement Unit (IMU) with an additional Global Navigation Satellite System (GNSS) and transferred to the virtual vehicle in the simulation. Thus, it is possible to repeat the real reference measurements in the simulation. As before, one reference recording is compared to the respective simulated recording. With the single pixel analysis the velocity of the target vehicle is determined for each frame of the recording and in the end, the DVM is calculated with the reference and simulated velocity values. Thereby, the first test case tackles the velocity $\dot{r}$ and the velocity noise $\dot{r}_{\text{noise}}$. The model bias makes a statement about $\dot{r}$ and the model scattering error about $\dot{r}_{\text{noise}}$. For completeness, the second test case evaluates the velocity resolution $\Delta\dot{r}$ and instead of velocity values, the DVM is calculated with values of the velocity resolution. The test cases are summarized in Table 4-8.

Table 4-8: Test cases for radial velocity measuring validation.

Validation Test Case	Tested Parameters
1	Velocity $\dot{r}$ and Velocity noise $\dot{r}_{\text{noise}}$
2	Velocity resolution $\Delta\dot{r}$

4.3.2 Experimental Reference Measurements and Determination of System Parameters

To begin with, the experimental setup of the velocity reference measurements is introduced. Afterwards, the process of deriving the system parameters $\dot{r}_{\text{noise}}$ and $\Delta\dot{r}$ is described.

Similar to the range reference measurements, the measurements for the radial velocity are carried out on the runway of the August Euler Airfield near Darmstadt. Furthermore, the experimental setup is basically the same, except that instead of the static lidar target a BMW i3 is used as a moving target (see Figure 4-24).

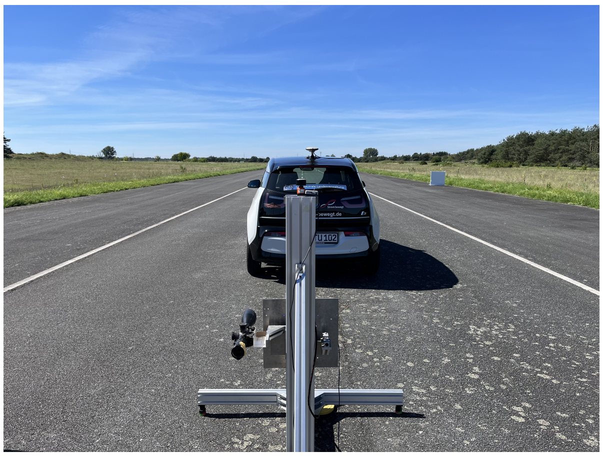


Figure 4-24: Sensor test stand and BMW i3 target vehicle at the beginning of a velocity reference recording.

Figure 4-25 depicts the experimental setup. The coordinate system S indicates the position of the sensor which is mounted on the sensor test stand at a height of 0.765 m above the ground. At the beginning of every recording, the vehicle (coordinate system V) is positioned in front of the sensor at a range of 1.5 m to 2.5 m facing the direction of the sensor's x-axis. From standstill, the vehicle is accelerated in direction of the sensor's x-axis to the desired velocity v_{desired}, which is permanently maintained until the end of the recording. A GeneSys Automotive Dynamic Motion Analyzer (ADMA), which is an IMU with additional GNSS information, is used to measure the actual reference velocity of the vehicle v_{ref}. Note that the desired velocity v_{desired} is the velocity, the driver is aiming for, while the reference velocity v_{ref} is the vehicle velocity, actually measured by the ADMA. Again, to avoid confusion, it is pointed out that the index ref refers to the reference measuring device (ADMA) and the index real relates to values measured by the real lidar sensor. The position, orientation and velocity of the vehicle is determined in reference to the A coordinate system of the ADMA. During the measurements, the weather was always dry and sunny with temperatures above 20 °C.

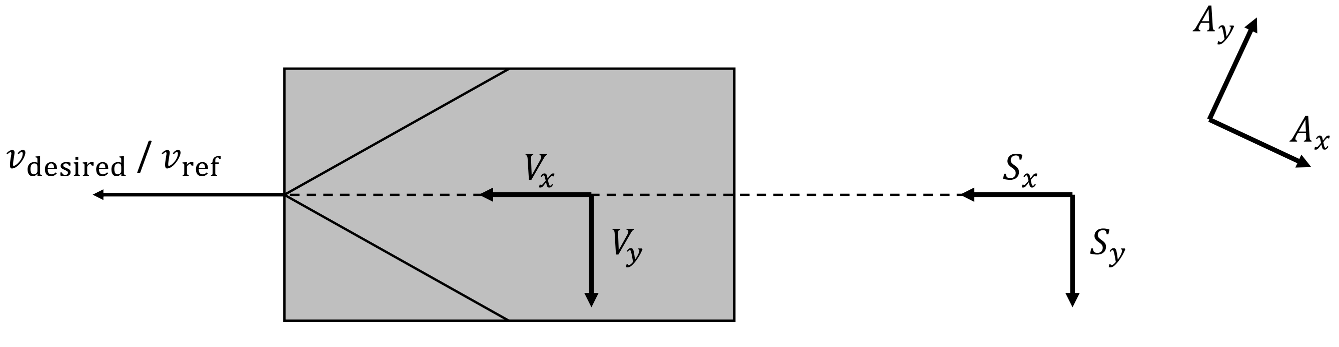


Figure 4-25: The sketch shows the experimental setup of the velocity reference measurements. The sensor is mounted on the origin of the S coordinate system at height of 0.765 m above the surface. V is the coordinate system of the target vehicle and A the coordinate system of the ADMA. The vehicle is moving along the x-axis of the sensor with the velocity v_{ref}. v_{desired} is the velocity to be reached and maintained by the driver.

In total, the four reference measurements, listed in Table 4-9, are conducted with the desired velocities

5 km/h, 10 km/h, 20 km/h and 30 km/h. While the ODD specifies velocities up to 60 km/h, a maximum velocity of 30 km/h is enforced due to a speed limit on the runway. Every measurement contains five recordings, three are designated for the determination of the sensor model's system parameters and two are for its validation. The vehicles velocity is measured with an uncertainty of ± 0.03 km/h in reference to the vehicle coordinate system's x-axis. Due to the aforementioned simplifications, the velocity uncertainty of the vehicles y-axis is neglected. Since the velocity of the target vehicle is measured by the ADMA in relation to its coordinate system A, the mentioned uncertainty is transformed from the vehicle coordinate system V to the ADMA coordinate system A and thereby split up into a x and a y component.

To be able to transfer the real reference measurements into the simulation, the timestamps of the ADMA and the real lidar sensor are both synchronized via GNSS. This enables the current position and velocity of the target vehicle to be determined for each frame. Subsequently, for every reference recording the target vehicle's position and velocity information is saved to a CSV file. The virtual target vehicle, which is a simple box equal to the size of a BMW i3, is then moved in the simulation as specified by these CSV files.

Table 4-9: Experimental reference measurements for radial velocity. Each measurement includes 5 recordings. The numbering of the reference recordings follows the scheme: *<Measurement Number>.<Recording Number>*.

Ref. No.	v_{desired} in km/h
1.1-5	5
2.1-5	10
3.1-5	20
4.1-5	30

The experimental setup used in this work entails two problems. First, without any driver assistance systems, it is very difficult for a human driver to precisely maintain a certain velocity, as illustrated in Figure 4-26, that shows the velocity of the vehicle over time measured by the ADMA. The drivers are instructed to continue driving as constantly as possible after reaching the desired velocity. However, while the velocity of about 5 km/h is maintained relatively well for a certain time, a constant section is hardly recognizable when driving at a velocity of 30 km/h. To determine the standard deviation of the velocity $\dot{r}_{\text{stdv}}$, that is used to model the noise, it is best to have constant vehicle velocity. Since the fluctuations in vehicle velocity are also reflected in the standard deviation, it is important to keep these as low as possible. For this reason, only the three recordings of the measurement with a desired velocity of 5 km/h are used to estimate the standard deviation. Furthermore, only the constant sections of the recordings are selected. The velocity values of the target vehicle, that serve the calculation of the standard deviation, are determined with the single pixel analysis. In contrast, for the determination of the velocity resolution $\Delta\dot{r}$, it is only a matter of collecting as many points as possible, to be able to identify a resolution. Thus, all three recordings of each measurement are utilized. Since clustering did not provide a result this time, the smallest distance of two velocity values is taken which leads to a very high resolution.

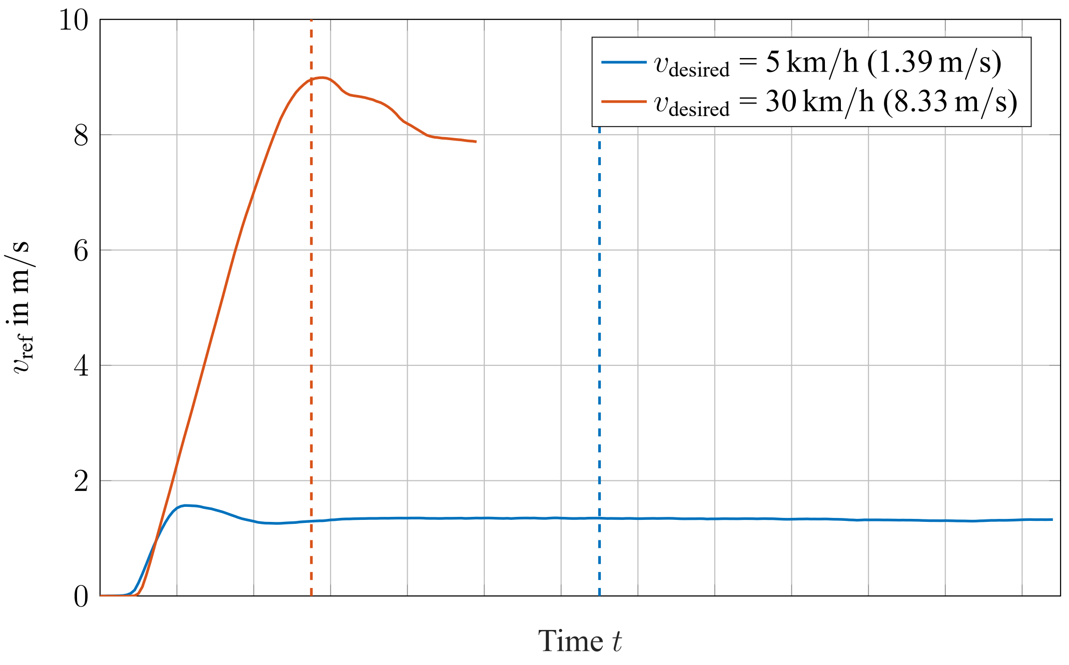


Figure 4-26: Velocity of the target vehicle over time of two exemplary reference recordings. The velocities depicted are measured by the ADMA. The dashed lines indicate the point at which the number of detections of the target vehicle is no longer sufficient for evaluation. For reasons of confidentiality, the x-axis tick labels are missing.

The second problem is that, especially at higher velocities, the vehicle is out of the sensor's range shortly after the desired velocity is reached (see Figure 4-26). This leads to the fact that not a constant velocity is evaluated, but mainly an acceleration. Especially the measurements with a desired velocity v_{desired} of 20 km/h and 30 km/h are affected by this problem.

4.3.3 Implementation and V&V of Relative Radial Velocity

Next, the implementation of the radial velocity is explained, verified and validated.

Implementation and Simulated Measurements

The implementation of the radial velocity is very similar to the implementation of the radial range, therefore no further modifications are needed on the transmitter-side signal processing. Nonetheless, modeling the velocity noise and resolution requires changes of the receiver-side signal processing. Due to the ray tracer's ability to directly output the radial velocity $\dot{r}_{\text{rt}}$ in addition to the radial range r_{rt}, the following processing of the velocity output is nearly the same as for the range. Thus, the velocity noise, which is assumed to be normally distributed, is applied to the individual rays during the ray tracing process. The previously determined standard deviation of the real sensor's radial velocity measuring $\dot{r}_{\text{stdv}}$ defines the implemented normally distributed noise. However, the implementation of the velocity resolution is different as it is neglected. This is reasoned by the fact that the determined velocity resolution is very high, which leads to the assumption that the resulting error is negligible. Whether or not this assumption is justified, is answered within the validation.

Moving forward, the evolved sensor model is utilized to generate simulated measurements for the verification and validation process. Since no relation to real measurements is needed for the verification, four made up measurements with one recording each are used. In contrast to the real measurements, the simulation allows for a consistently constant velocity of the target vehicle. On top of that, the vehicle does not have to be accelerated first, but is placed in front of the sensor already at the desired velocity. This means, that for the simulated verification recordings v_{desired} is equal to v_{ref}. To cover the velocities of the real measurements, the desired vehicle velocity v_{desired} is set to be $5\,\text{km/h}$, $10\,\text{km/h}$, $20\,\text{km/h}$ and $30\,\text{km/h}$ for the four simulated measurements. The number of frames of each recording is different depending on the vehicle's velocity. Table 4-10 gives an overview of all recordings. Since the simulated verification measurements are not based on the real reference measurements, all verification measurements are labeled as measurement 0 and just the four individual recordings are marked with 1 to 4.

On the other hand, each simulated recording, that is used for the validation, derives from a real reference measurement, as the simulated vehicle moves according to the real velocity measured by the ADMA v_{ref}. For the static range validation, every real reference recording of a certain range could be compared to a single respective simulated recording as the lidar target range is the same for every recording. The reference recordings of the radial velocity, however, are all unique as it is not possible for a human driver to exactly replicate a performed drive. Therefore, it is necessary to simulate an individual recording for every reference recording. In order to be able to consider the uncertainty of the ADMA, three recordings are simulated for every real reference recording. The first and second column of Table 4-10, which lists all simulated recordings, indicate the relation of every recording.

Table 4-10: Overview of all simulated radial velocity measurements. The right side of the table indicates which recordings are used in which test cases. $v_{\text{ref,sim}}$ is the velocity of the simulated target vehicle. The numbering of the simulated recordings follows the scheme: *<Measurement Number>.<Recording Number>*.

Sim. No.	Based on Ref. No.	v_{desired} in km/h	$v_{\text{ref,sim}}$ in km/h	Verification TC			Validation TC	
				1	2	3	1	2
0.1	-	5	5	X	X	X		
0.2		10	10	X	X	X		
0.3		20	20	X	X	X		
0.4		30	30	X	X	X		
1.1	1.4	5	v_{ref}				X	X
1.2		5	v_{ref} + 0.03				X	X
1.3		5	v_{ref} - 0.03				X	X
1.4	1.5	5	v_{ref}				X	X
1.5		5	v_{ref} + 0.03				X	X
1.6		5	v_{ref} - 0.03				X	X
2.1	2.4	10	v_{ref}				X	X
2.2		10	v_{ref} + 0.03				X	X
2.3		10	v_{ref} - 0.03				X	X
2.4	2.5	10	v_{ref}				X	X
2.5		10	v_{ref} + 0.03				X	X
2.6		10	v_{ref} - 0.03				X	X
3.1	3.4	20	v_{ref}				X	X
3.2		20	v_{ref} + 0.03				X	X
3.3		20	v_{ref} - 0.03				X	X
3.4	3.5	20	v_{ref}				X	X
3.5		20	v_{ref} + 0.03				X	X
3.6		20	v_{ref} - 0.03				X	X
4.1	4.4	30	v_{ref}				X	X
4.2		30	v_{ref} + 0.03				X	X
4.3		30	v_{ref} - 0.03				X	X
4.4	4.5	30	v_{ref}				X	X
4.5		30	v_{ref} + 0.03				X	X
4.6		30	v_{ref} - 0.03				X	X

Verification

The following verification process checks whether or not the sensor model's radial velocity measuring returns the expected output. For this purpose, for each test case, the four simulated recordings number 0.1 to 0.4 are utilized. Since the velocity resolution is neglected, the velocity $\dot{r}$ and the velocity noise $\dot{r}_{\text{noise}}$ are the two parameters to be verified.

To begin with, the **first verification test case** addresses the velocity $\dot{r}$ by comparing the radial velocity $\dot{r}_{\text{sim}}$ measured by the virtual sensor with the simulated target vehicle velocity $v_{\text{ref,sim}}$. For each frame of the simulated recordings, the radial velocity value is determined with the single pixel analysis together with its respective corrected nominal velocity $\dot{r}_{\text{nom,c}}$. The corrected nominal velocity $\dot{r}_{\text{nom,c}}$ is based on the target vehicle velocity $v_{\text{ref,sim}}$, but additionally accounts for the non-perpendicular incident angle on the vehicle. Since the vehicle moves perfectly along the x-axis of the sensor, every deviation from the 0° azimuth or elevation angle reduces the measured radial velocity $\dot{r}_{\text{sim}}$.

Subtracting $\dot{r}_{\text{nom,c}}$ from $\dot{r}_{\text{sim}}$ for each frame across all four recordings yields the result of this test case, which is illustrated in Figure 4-27. Similar to the range verification, the pass-fail criterion of this test case is defined as whether or not the difference is justifiable by the implemented velocity noise. The boxplot proves that the median is close to zero and the minimum and maximum difference is explainable by the velocity noise. Therefore the test case is considered as passed.

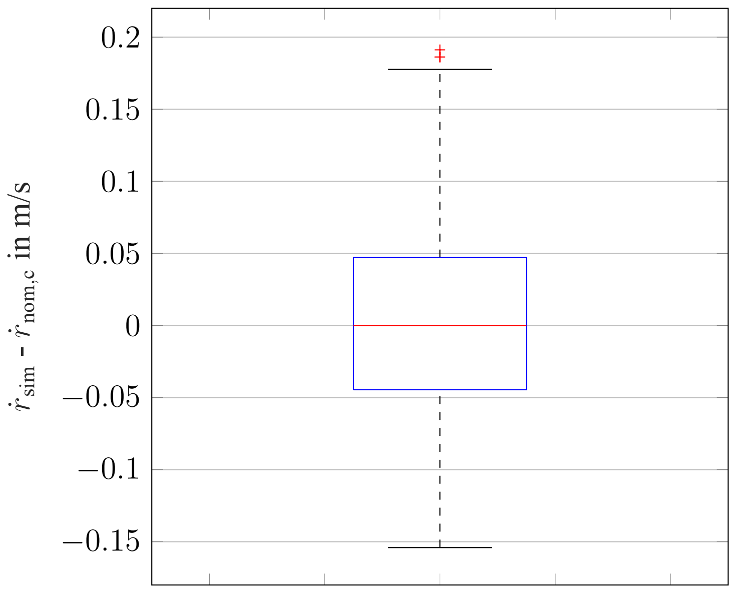


Figure 4-27: Verification result of the simulated velocity $\dot{r}_{\text{sim}}$. For the difference of the simulated velocity $\dot{r}_{\text{sim}}$ and the corrected nominal velocity $\dot{r}_{\text{nom,c}}$ four recordings of different lengths are evaluated.

Next, the correct implementation of the velocity noise $\dot{r}_{\text{noise}}$ is checked by the **second verification test case**. For each of the four simulated recordings, the velocity values of all frames are determined with the single pixel analysis. Subsequently, the standard deviation of the velocity measured in the simulation $\dot{r}_{\text{stdv,sim}}$ is calculated for every recording and then compared to the nominal standard deviation $\dot{r}_{\text{stdv,nom}}$ that is implemented into the sensor model. Note that, due to the different velocities, the number of frames is different for every recording, to ensure they stop approximately at the same driven distance of the target vehicle. Table 4-11 summarizes the difference of the standard deviation together with the number of frames and the simulated velocity of the vehicle. It is noticeable, that the difference decreases as the number of frames increases which leads to the assumption that statistical deviations still have an influence on the outcome. To further support this assumption, an additional experimental recording with 1000 frames is captured. The unique feature of this experimental recording is that the vehicle is assigned a velocity, but the position remains the same. Thereby, it is possible to record many frames without the problems that come with the increasing range to the target vehicle. A standard deviation difference of 0.0018 m/s for the experimental recording backs up the assumption that the number of frames used is insufficient. To account for these statistical deviations, a pass-fail threshold of 0.01 m/s is selected. Since this threshold is not exceeded, the second verification test case is passed.

Table 4-11: Verification result of the simulated velocity noise $\dot{r}_{\text{noise,sim}}$. The number of frames is different as the simulation stops after a certain distance is driven by the simulated target vehicle.

$v_{\text{ref,sim}}$ in km/h	5	10	20	30
$\lvert \dot{r}_{\text{stdv,sim}} - \dot{r}_{\text{stdv,nom}} \rvert$ in m/s	0.0041	0.0076	0.0082	0.0073
Number of Frames	499	249	124	83

Validation

After verifying the correct implementation of the radial velocity, the sensor model's velocity output is contrasted with the real sensor's reference measurements. Given that four reference measurements are conducted with two validation recordings each, eight real reference recordings are available. As for every real reference recording three simulated recordings are necessary, there are 24 comparisons for the validation in total. Figure 4-28 illustrates the velocity values of a real reference recording, the respective simulated recording and the velocity of the target vehicle measured by the ADMA.

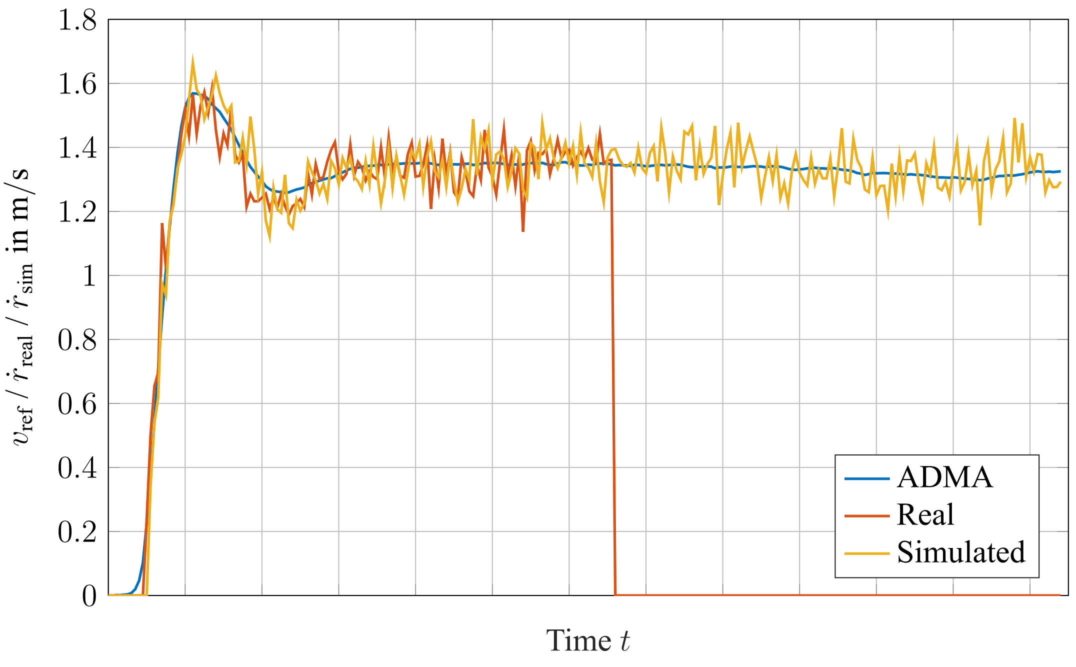


Figure 4-28: Comparison of the real measured velocity and the velocity measured in the simulation in an example recording. In addition, the velocity measured by the ADMA is plotted. For reasons of confidentiality, the x-axis tick labels are missing.

To begin with, the **first validation test case** investigates the radial velocity $\dot{r}$ and the velocity noise $\dot{r}_{\text{noise}}$. Every of the 24 aforementioned comparisons proceeds as follows: First, the radial velocity $\dot{r}$ measured by the sensor is determined for every frame of both the real reference and the simulated recording. This is done with the single pixel analysis. Due to unrelated effects that cause missing detections, some frames of the reference recordings remain without a velocity value. To compensate for that and balance the number of values, the respective frames of the simulated recordings are also left without a velocity value. Based on these two value sets the DVM is calculated. Repeating this procedure for every comparison leads to the result depicted in Figure 4-29.

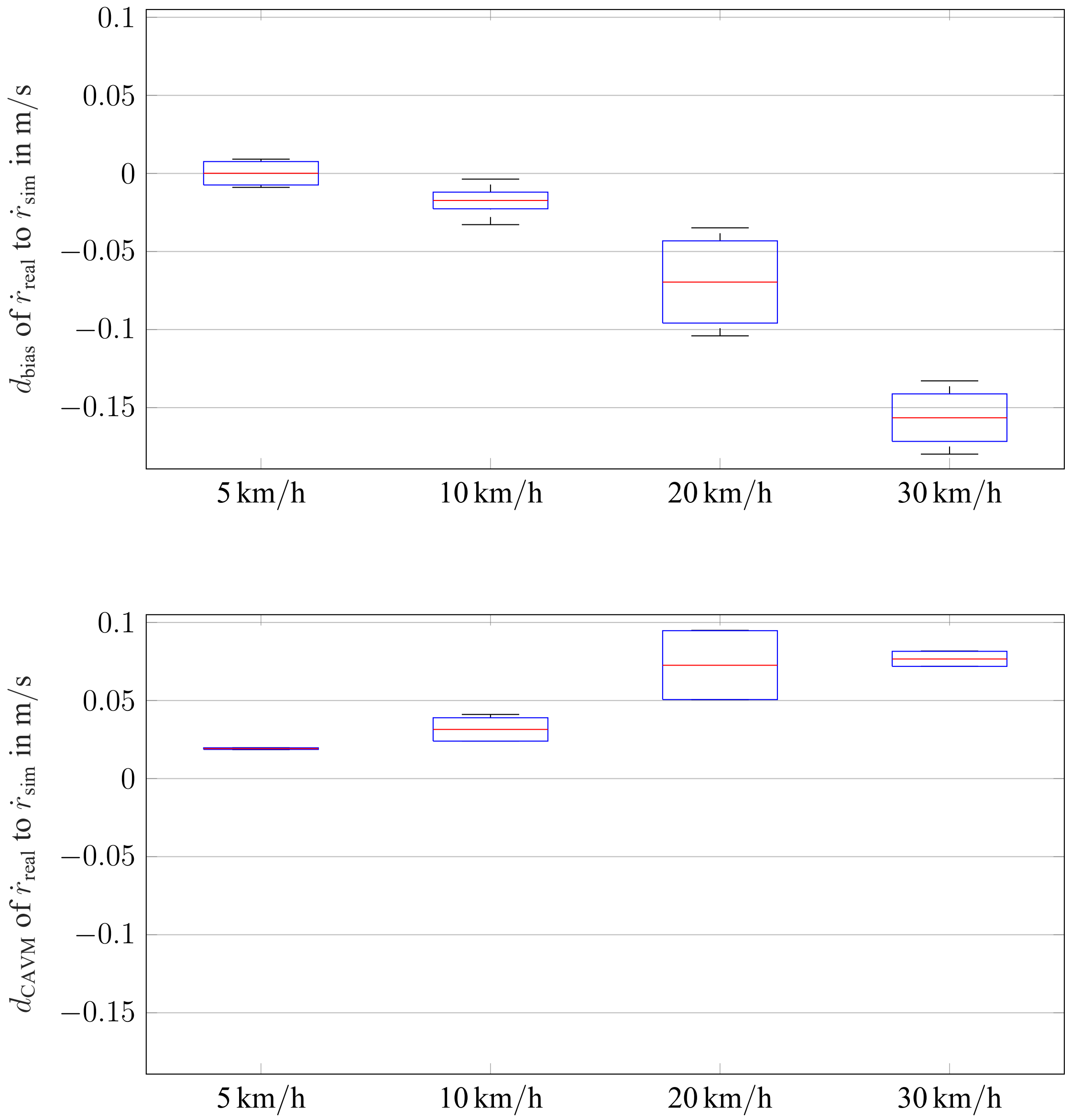


Figure 4-29: The boxplots show the model bias d_{bias} (top) and the model scattering error d_{CAVM} (bottom) of the velocity measuring for four different target vehicle velocities. Given that 24 comparisons are performed in total, each of the boxes contains six biases or scattering errors.

The boxplots show that both the model bias and the scattering error increase with higher desired velocities which is similar to the behavior of the radial range. Thereby, the pass-fail criteria are chosen to be similar to the range validation. For the bias, which is used as an indicator for the velocity $\dot{r}$, a maximum deviation of 1 % is tolerated, whereby, this is based on the desired velocity. The threshold for the scattering error, which provides information about velocity noise $\dot{r}_{\text{noise}}$, is assumed to be proven by a sensitivity analysis and is set to 5 cm/s.

The threshold of 1 % for the bias is only complied for recordings with a desired velocity of 5 km/h, which leads to the outcome that the test case is not passed for velocities greater than 5 km/h. Same is observed for the scattering error, only the recordings of 5 km/h and 10 km/h lie below the threshold of 5 cm/s. As a result, the first validation test case is considered failed.

At this point, it is referred to the two problems with the experimental setup described in Chapter 4.3.2. According to these, the real reference measurements with a desired velocity of 10 km/h, 20 km/h and 30 km/h correspond to an acceleration rather than a constant velocity. Therefore, there is a possibility that the test case did not fail due to a faulty sensor model, but because of improper reference measurements. Furthermore, the lower number of frames at the higher velocity measurements might lead to a statistically insignificant amount of velocity values.

Although the first test case has already failed, the **second validation test case**, which covers the velocity resolution is also checked for the purpose of completeness. As with the previous test case, the reference recordings are each compared to their respective three simulated recordings leading to 24 comparisons in total. The resolution is determined for every pair of recordings and then the difference of the two resolutions is calculated. To pass this test case, the difference must not exceed 1 mm/s. This threshold is again based on the results of an assumed sensitivity analysis. Since the maximum calculated difference is in the region of 1×10^{-8} m/s, the test case is considered as passed. The overall validation result is that the velocity measuring of the sensor model is only valid for a target vehicle velocity of 5 km/h.

5 Assessment of Methodology, Realization and Result of the Model Development

In the upcoming chapter, the requirement definition and the model development methodology, described in Chapter 3, are evaluated first. Subsequently, the practical realization of the signal processing model addressed in Chapter 4, as well as the resulting model itself, are assessed. For this purpose, further necessary improvements are pointed out and discussed.

5.1 Assessment of Requirement Definition and Model Development Methodology

Within the context of the requirement definition (Chapter 3.2.2), the first thing to mention is the bypassing of the CEPRA step to obtain the internal sensor simulation requirements. Instead, the most basic design parameters are selected, resulting in a subjective decision in the process. Therefore, for larger projects with more resources, it is recommended to conduct a CEPRA in order to establish objective justifications for the selection of design parameters, effects or sensor-independent causes.

The next problem concerns the selection or development of the implementation approaches, which define how the previously selected design parameters, effects and sensor-independent causes are realized in the model. Within the model developed of this work, this is rather straightforward, as only three design parameters have to be considered and it is therefore simple to find a compatible solution. However, for more complex models with a large number of requirements, it is likely that this step will be much more difficult. On the one hand, it is important to ensure that all effects are covered by the implementation approaches, and on the other hand, it is essential that all approaches are compatible with each other. The iterative extension of the model by adding new design parameters, effects and sensor-independent causes step-by-step bears the risk of complications arising at a later stage, if compatibility is not ensured beforehand. If this task proves to be a problem for more complex models, it is necessary to develop a systematic methodology to support the selection and development of implementation approaches. This problem also raises the question in which order the implementation should take place. However, this might be solved by following the sequence of the respective cause-effect chains.

Another weakness is the determination of the test cases' pass-fail criteria through a sensitivity analysis. For this purpose, the influence of different simulation parameters on the SUT is investigated in order to identify the true significance of each parameter. Based on this analysis, pass-fail criteria are designed more strictly or more loosely. From the point of view of the model developers, the SUT is required for this analysis which is not always fulfilled if the development of both components is carried out by two different organizations, for example. Therefore, a close exchange is necessary in this case or if possible, the developers of the model should be given easy access to the SUT.

In the description of the test cases in Chapter 3.2.2 it is stated that the test cases specify what reference data is required for the validation of the sensor model. Nevertheless, it is not defined how the data is to be collected and what experiments and measurements need to conducted. Accordingly, this work lacks a methodology that systematically describes the process of deriving practical experimental setups from the previously defined internal sensor simulation requirements. Due to the rather simple sensor model with just three design parameters and the expectation from the assignment that the validation should only be demonstrated exemplarily, the selection of the experimental setups is fairly intuitive in the context of this work. However with more complex models that need to be fully validated, this step is certainly not as simple and thus an additional methodology is required to assist this task.[101]

In relation to this, another missing step is the validation of the recorded reference measurement data, which is taken into account in the methodology of Viehof and Rosenberger et al.(see Figure 2-9), for example. In order to be able to conduct the validation exemplarily within the context of this work, the assumption is made that the measurement data are validated, which represents a simplification that cannot be justified in future sensor model developments.

The model development methodology proposed in Chapter 3.3 intends to individually and separately implement, verify and validate the selected design parameters, effects or sensor-independent causes. However, for effects that are strongly linked to other effects for example, it may not be possible to verify or validate these separately. One possible solution would be to allow certain strongly related effects to be implemented in groups and, subsequently, the verification and the validation are performed for the whole group and not individual effects. The problem of non-separable effects also concerns the realization of the model, as will become apparent in the next chapter.

Lastly, the revision step of the model development methodology, which is shown in Figure 3-6, needs to be addressed. The revision step is performed after a failed test case to identify the cause and to eliminate the underlying problem(s). For this purpose, a brief list of questions is given that possibly contributes to this. Yet, there is no explicit instruction provided on the course of action to follow in case of a failed test case. Therefore, the elaboration of the revision step is the subject of further research.

5.2 Assessment of Realization of Model Implementation and V&V

After dealing with the problems that mainly concern the methodology, the realization of the model is discussed. Within the realization of the FMCW lidar signal processing model in Chapter 4, an assumed sensitivity analysis is often utilized as a basis to specify the more detailed requirements or pass-fail thresholds. However, this analysis is not conducted within the scope of this work, as the SUT is not available. Therefore, many of the justifications are based on assumptions. While care is taken to ensure that these assumptions appear realistic, they can only be confirmed with certainty by the sensitivity analysis, that would need to be carried out in the future development of the model.

A further simplification is that the test cases of the beam pattern are not retested in the following iterations of the radial range and the radial velocity, as it is actually required by the model development methodology. This also applies to the test cases of the radial range, that are not retested in the iteration of the radial velocity. For one thing, the rechecking is omitted in this work because the signal processing model with the three design parameters is relatively simple and it is therefore possible to predict with certainty that the results of the test cases are not affected by the implementation of the radial range and velocity. Furthermore, the number of necessary simulations (see Table 4-6 and Table 4-10), which all need to be processed manually, makes it evident that rechecking the test cases involves a great amount of time and effort that exceed the resources of this work. This problem directly leads to the next deficiency of the model development methodology, which is that without an automated evaluation of the test cases, the verification and validation effort increases significantly with each iteration. For this reason, it is recommended to realize the automation described in Chapter 3.3.2.

[101] This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

The issue of effects that are difficult to separate, which is addressed in the previous chapter, also leads to problems within the realization of the model, specifically the validation. For illustration purposes, the first validation test case of the radial range implementation in Chapter 4.2.3 is considered. This test case investigates whether the range noise behavior is correctly reproduced by the sensor model on angled surfaces. Despite the fact that the test case is considered to be passed, the scattering error suggests that the range noise increases at larger angles of the target (see Figure 4-21). This assumption is potentially supported by the observation that scan lines that hit the asphalt surface at a very flat incident angle (equivalent to target angles of $> 89°$) show a significant increase in noise. However, since the detections on the asphalt surface occur at greater ranges due to the flat angle, there is a possibility that the range noise is caused by the range and not the incident angle. A third potential factor is the roughness of the asphalt surface. Thus, this observed effect yields at least three possible causes and in order to validate it, at least three different experimental setups are necessary in which each of the three causes are investigated separately. The influence of range is covered in the second validation test case, although for a complete validation of the range noise, the influence of surface properties and flatter angles still needs to be investigated systematically.

For future dynamic measurements, it is also advisable to select a different target for the velocity reference measurements, if the exact surface properties and geometries of the BMW i3 are unknown. Simpler targets as, for example, the lidar target mounted on a trailer are easier to transfer to a virtual environment as the properties and dimensions of the lidar target are known. Since the reflectivity and the exact geometries of the BMW i3 are not particularly relevant in the scope of this work, it is not a problem that a simple 3D box is used to model the car in the simulation. For simulations where the material properties of the objects need to be considered, it is important that the corresponding 3D assets of the objects are available and validated.[102]

Another point of criticism, which concerns the validation, is that for almost all calculations of the DVM recordings with 150 frames each are employed. The chosen length is primarily based on the fact that the recordings are processable in Matlab in a reasonable amount of time. Consequently, it is necessary to analyze the influence of the length of the recordings on the validation results in the future.

Furthermore, it is important to mention that some test cases are only testable in the described way due to the low complexity of the model. The second validation test case of the beam pattern is an example for this (see Chapter 4.1.3). As part of the validation of the azimuth angle, it compares the number of detection points per scan line of the simulated recordings with the number of the real reference recordings. This is possible, since in the simulation all existing pixels contain a detection point and the beam pattern is therefore 100 % complete. If, for example, effects are implemented that result in missing detections, this test case loses its significance. Therefore, it is important to ensure that the test cases are applicable regardless of the model's complexity. Another issue with the validation of the beam pattern is that only the pixels' relative azimuth angles are validated and not the absolute ones. However, this could be fixed with a pixel ID.

[102]This requirement originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

Since the validation within the scope of this work is only carried out on an exemplary and sample basis, further steps are necessary to obtain a fully validated model. Rosenberger[103] mentions two of these steps in his work. The first concerns the validity of the range and the velocity. Since the range and velocity values used in the validation are determined with the single pixel analysis, a validity statement is only possible for this one pixel. In order to make a conclusion about the other pixels, further measurements are required that include the remaining FoV. Furthermore, the validation of the range only covers certain discrete ranges, as only six different ones are represented in the measurements. In order to assess the validity of ranges that are not directly measured in real reference measurements, an interpolation or extrapolation method is necessary, as described by Rosenberger. Finally, the external simulation requirements such as the execution time of the simulation need to be reviewed in the future.

5.3 Assessment of Resulting Signal Processing Model

Lastly, the resulting signal processing model itself is evaluated. From the previous two chapters it is clear that the model is not fully validated in regard to the beam pattern, radial distance and radial velocity. To achieve a complete validation, further measurements with other experimental setups are necessary to obtain more suitable reference data. Existing experimental setups also require optimization, such as the one for the radial velocity. As already briefly discussed in the validation of the radial velocity (see Chapter 4.3.3), the failure of the test cases for the velocities of 10 km/h, 20 km/h and 30 km/h is not necessarily caused by the sensor model, but presumably by the data of the reference measurements, which represent an acceleration rather than a constant velocity.

In addition to the incomplete validation, the limited extensibility of the model is to be mentioned. This is primarily the result of the very simple implementation approach. Because each pixel is sampled by only one ray, it is difficult to reproduce effects such as the beam divergence or multiple returns. For example, the aforementioned (Chapter 3.2.2) super-sampling approach described by Rosenberger et al.[104] would be better suited to represent these effects. However, this would also require changes to the existing implementation of the beam pattern, radial range and radial velocity.

[103] Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022), pp. 129ff.

[104] Rosenberger, P. et al.: Sequential lidar sensor system simulation (2020).

6 Conclusion and Outlook

In this work, a signal processing model of an FMCW lidar sensor is developed. After resolving the question of what is understood by a signal processing model and separating it from the other two components of the sensor simulation, the environment simulation and the signal propagation model, the methodology for deriving the requirements is introduced. For this purpose, an ODD is defined according to which the FMCW lidar sensor is hypothetically deployed in a vehicle traveling at velocities of up to 60 km/h on a multi-lane highway under optimal environmental conditions and high traffic volume. Using the ODD and reflecting on which are the most basic design parameters, the internal sensor simulation requirements are deduced. The distinction between internal and external sensor simulation requirements is one of the modifications to the five-step requirement definition approach of Rosenberger et al. which serves as the basis for the advanced methodology presented. In the end, the three design parameters: beam pattern, radial range and relative radial velocity are selected for the implementation into the signal processing model. With these three design parameters, the basic functions of an FMCW lidar are covered and the ability of directly measuring the velocity sets it apart from ToF lidar models. To realize the model, an implementation approach is chosen that samples each pixel of the beam pattern with exactly one ray from the ray tracer. Following that, the model development methodology is introduced, whereby the design parameters are implemented successively one by one and in the process the model is continuously verified and validated.

The model development methodology is subsequently applied for the realization of the signal processing model. In preparation for this, the test cases are defined for each of the three design parameters and real reference measurements are conducted. The experimental setup for the beam pattern is the most straightforward, as all that is needed is a white wall of sufficient size to cover the sensor's FoV. For the range and the velocity, the experimental reference measurements are performed on an airfield runway. A static lidar target placed at different ranges is used to obtain range reference data. The setup for the velocity measurements is similar but instead of a static setting, a BMW i3 moving away from the sensor at different velocities serves as the target. On the one hand, the reference data will be needed for validation and, on the other hand, they are used to derive system parameters. With the help of the system parameters and the reference data, the signal processing model is iteratively built. In each iteration, the current model is applied to re-simulate the real reference measurements in order to obtain simulated data, which is then compared to the real reference data within the validation of the model. For this purpose, with one exception in the validation of the beam pattern, the DVM is employed, which proves to be suitable.

The DVM enables an intuitive understanding of both its application and the result. Another advantage is that the scattering error is determined in addition to the bias, so that, for example, both the range and the range noise are covered at the same time during range validation. In the end, the developed signal processing model is validated, on a sample basis and with the assumptions made, for the beam pattern and the radial range. The next step is to confirm the validity for ranges and pixels, that are not covered directly, by additional measurements or interpolation and extrapolation approaches. Since it is suspected that the velocity validation test cases fail due to unsuitable reference measurements, it is necessary to design other experimental setups to validate the model in regard to the velocity in the future. Furthermore, the assumptions and pass-fail criteria, based on a sensitivity analysis, are to be checked by actually performing the analysis. For this purpose, it is conceivable to train an object detector with real sensor data and then investigate its performance on different simulation data.[105]

In order to enable a more complex model and incorporate effects such as beam divergence or the intensity, a more sophisticated implementation approach is required in the future, one that is based on a super-sampling approach, for example. However, it has to be kept in mind that if other interfaces are simulated as part of these approaches, they must also be available in the real reference data.

In the context of the model development methodology, it is important to clearly define what needs to be done in the event of a failed test case by further specifying the revision step. It is also evident that the proposed automatic review of the test cases is indispensable for the realization of more complex models. To enable the validation of these complex models, a systematic deduction of experimental setups based on the derived requirements is needed, which leads to an additional subject of future research. Another future research question is how to separate strongly related effects for both implementation and validation.

Although, the applied methodologies still contain weaknesses as well as limitations and the developed signal processing model is simple and not fully validated, this work demonstrates that the basic ideas and approaches applied are potentially capable of enabling more complex models.

[105]This idea originates from a consultation with Dr.-Ing. Clemens Linnhoff (Persival GmbH) and M. Sc. Lukas Elster (FZD).

Bibliography

Ahmann, M. et al.: Towards Continuous Simulation Credibility Assessment (2022)
Ahmann, Maurizio; Le, Van Thanh; Eichenseer, Frank; Steimann, Frederik; Benedikt, Martin: Towards Continuous Simulation Credibility Assessment, in: Proceedings of Asian Modelica Conference, pp. 171–182, 2022

Amersbach, C. T.: Diss., Functional Decomposition (2020)
Amersbach, Christian Thomas: Functional Decomposition Approach - Reducing the Safety Validation Effort for Highly Automated Driving, Dissertation Technischen Universität Darmstadt, 2020

ASAM e.V.: ASAM OpenDRIVE® (2021)
ASAM e.V.: ASAM OpenDRIVE®, URL: https://www.asam.net/standards/detail/opendrive/, 2021, visited on 09/25/2023

ASAM e.V.: ASAM OpenSCENARIO® (2022)
ASAM e.V.: ASAM OpenSCENARIO®, URL: https://www.asam.net/standards/detail/openscenario/, 2022, visited on 09/25/2023

ASAM e.V.: ASAM OSI® (Open Simulation Interface) (2023)
ASAM e.V.: ASAM OSI® (Open Simulation Interface), URL: https://www.asam.net/standards/detail/osi/, 2023, visited on 09/25/2023

Automotive Solution Center for Simulation e.V.: OpenMSL SL-5-3 OSMP Network Proxy (2023)
Automotive Solution Center for Simulation e.V.: OpenMSL SL-5-3 OSMP Network Proxy, URL: https://github.com/openMSL/sl-5-3-osmp-network-proxy, 2023, visited on 10/15/2023

Automotive Solution Center for Simulation e.V.: OpenMSL SL-5-6 OSI Trace File Writer (2023)
Automotive Solution Center for Simulation e.V.: OpenMSL SL-5-6 OSI Trace File Writer, URL: https://github.com/openMSL/sl-5-6-osi-trace-file-writer, 2023, visited on 10/15/2023

Barry, J. R. et al.: Performance of coherent optical receivers (1990)
Barry, John R.; Lee, Edward A.: Performance of coherent optical receivers, in: Proceedings of the IEEE, Vol. 78, pp. 1369–1394, 1990

Behroozpour, B. et al.: Lidar System Architectures and Circuits (2017)
Behroozpour, Behnam; Sandborn, Phillip A. M.; Wu, Ming C.; Boser, Bernhard E.: Lidar System Architectures and Circuits, in: IEEE Communications Magazine, Vol. 55, pp. 135–142, 2017

Benke, K. K. et al.: Error propagation in computer models (2018)
Benke, K. K.; Norng, S.; Robinson, N. J.; Benke, L. R.; Peterson, T. J.: Error propagation in computer models: analytic approaches, advantages, disadvantages and constraints, in: Stochastic Environmental Research and Risk Assessment, Vol. 32, pp. 2971–2985, 2018

British Standards Institution: PAS 1883:2020 (2020)
British Standards Institution: PAS 1883:2020: Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification, URL: https://www.bsigroup.com/globalassets/localfiles/en-gb/cav/pas1883.pdf, 2020, visited on 09/26/2023

Cao, P.: Diss., Modeling Active Perception Sensors for Real-Time Virtual Validation (2018)
Cao, Peng: Modeling Active Perception Sensors for Real-Time Virtual Validation of Automated Driving Systems, Dissertation TU Darmstadt, 2018

DIN Deutsches Institut für Normung e. V.: DIN SAE SPEC 91471:2023-05 (2023)
DIN Deutsches Institut für Normung e. V.: DIN SAE SPEC 91471:2023-05 Assessment methodology for automotive LiDAR sensors, 2023

Eclipse Foundation: Eclipse OpenMCx (2023)
Eclipse Foundation: Eclipse OpenMCx, URL: https://projects.eclipse.org/projects/automotive.openmcx, 2023, visited on 09/25/2023

Elster, L. et al.: Introducing the Double Validation Metric (2023)
Elster, Lukas; Rosenberger, Philipp; Holder, Martin; Mori, Ken; Staab, Jan; Peters, Steven: Introducing the Double Validation Metric for Radar Sensor Models, URL: https://www.researchsquare.com/article/rs-3088648/v1, 2023, visited on 10/06/2023

Ferson, S. et al.: Model validation and predictive capability (2008)
Ferson, Scott; Oberkampf, William L.; Ginzburg, Lev: Model validation and predictive capability for the thermal challenge problem, in: Computer Methods in Applied Mechanics and Engineering, Vol. 197, pp. 2408–2430, 2008

GitHub - esmini: Environment Simulator Minimalistic (esmini) (2023)
GitHub - esmini: Environment Simulator Minimalistic (esmini), URL: https://github.com/esmini/esmini, 2023, visited on 09/25/2023

Haines, E. et al.: Ray Tracing Terminology (2019)
Haines, Eric; Shirley, Peter: Ray Tracing Terminology, in: Haines, Eric; Akenine-Möller, Tomas (ed.): Ray Tracing Gems, Apress, 2019

Hashemi, H.: A Review of Silicon Photonics LiDAR (2022)
Hashemi, Hossein: A Review of Silicon Photonics LiDAR, in: 2022 IEEE Custom Integrated Circuits Conference (CICC), pp. 1–8, 2022

Heinkel, H.-M. et al.: Credible Simulation Process Framework (2023)
Heinkel, Hans-Martin; Steinkirchner, Kim: Credible Simulation Process Framework, URL: https://gitlab.setlevel.de/open/processes_and_traceability/credible_simulation_process_framework, 2023, visited on 10/08/2023

International Organization for Standardization: ISO 23150:2021(E) (2021)
International Organization for Standardization: ISO 23150:2021(E): Road vehicles - Data communication between sensors and data fusion unit for automated driving functions - Logical interface, 2021

Kim, T.: Diss., Realization of Integrated Coherent LiDAR (2019)
Kim, Taehwan: Realization of Integrated Coherent LiDAR, Dissertation EECS Department, University of California, Berkeley, 2019

Lee, J. et al.: Frequency Modulation Control of an FMCW LiDAR (2023)
Lee, Jubong; Hong, Jinseo; Park, Kyihwan: Frequency Modulation Control of an FMCW LiDAR Using a Frequency-to-Voltage Converter, in: MDPI Sensors, Vol. 23, p. 4981, 2023

Linnhoff, C.: Diss., Analysis of Environmental Influences for Simulation (2023)
Linnhoff, Clemens: Analysis of Environmental Influences for Simulation of Active Perception Sensors, Dissertation Technische Universität Darmstadt, 2023

Linnhoff, C. et al.: PerCollECT - LidarLimbs (2022)
Linnhoff, Clemens; Hinsemann, Timo; Rosenberger, Philipp; Elster, Lukas: PerCollECT - LidarLimbs, URL: https://github.com/PerCollECT/LidarLimbs, 2022, visited on 09/26/2023

Linnhoff, C. et al.: Measuring the Influence of Environmental Conditions on Lidar Sensors (2022)
Linnhoff, Clemens; Hofrichter, Kristof; Elster, Lukas; Rosenberger, Philipp; Winner, Hermann: Measuring the Influence of Environmental Conditions on Automotive Lidar Sensors, in: MDPI Sensors, Vol. 22, p. 5266, 2022

Linnhoff, C. et al.: Towards Serious Sensor Simulation for Safety Validation (2021)
Linnhoff, Clemens; Rosenberger, Philipp; Schmidt, Simon; Elster, Lukas; Stark, Rainer; Winner, Hermann: Towards Serious Perception Sensor Simulation for Safety Validation of Automated Driving - A Collaborative Method to Specify Sensor Models, in: 2021 IEEE 24th International Conference on Intelligent Transportation Systems (ITSC), pp. 2688–2695, 2021

Ludwig Friedmann: OpenMATERIAL (2023)
Ludwig Friedmann: OpenMATERIAL, URL: https://github.com/LudwigFriedmann/OpenMATERIAL, 2023, visited on 09/25/2023

Marti, E. et al.: A Review of Sensor Technologies for Perception in Automated Driving (2019)
Marti, Enrique; Miguel, Miguel Angel de; Garcia, Fernando; Perez, Joshue: A Review of Sensor Technologies for Perception in Automated Driving, in: IEEE Intelligent Transportation Systems Magazine, Vol. 11, pp. 94–108, 2019

Mercedes-Benz: DRIVE PILOT (2023)
Mercedes-Benz: DRIVE PILOT, URL: https://www.mercedes-benz.de/passengercars/technology/drive-pilot.html, 2023, visited on 05/29/2023

Mercedes-Benz: Vorreiter bei automatisierten Fahr- und Sicherheitstechnologien (2022)
Mercedes-Benz: Vorreiter bei automatisierten Fahr- und Sicherheitstechnologien, URL: https://group.mercedes-benz.com/innovation/case/autonomous/drive-pilot.html, 2022, visited on 05/29/2023

Mitschke, F.: Fiber Optics (2010)
Mitschke, Fedor: Fiber Optics, Springer, Berlin Heidelberg, 2010

Modelica Association: Functional Mock-up Interface (FMI) (2023)
Modelica Association: Functional Mock-up Interface (FMI), URL: https://fmi-standard.org/, 2023, visited on 09/25/2023

Modelica Association: Functional Mock-up Interface Specification (2023)
Modelica Association: Functional Mock-up Interface Specification, URL: https://fmi-standard.org/docs/3.0/#_overview, 2023, visited on 09/25/2023

Mohammed, A. S. et al.: The Perception System of Intelligent Ground Vehicles (2020)
Mohammed, Abdul Sajeed; Amamou, Ali; Ayevide, Follivi Kloutse; Kelouwani, Sousso; Agbossou, Kodjo; Zioui, Nadjet: The Perception System of Intelligent Ground Vehicles in All Weather Conditions: A Systematic Literature Review, in: MDPI Sensors, Vol. 20, p. 6532, 2020

NVIDIA Corporation: NVIDIA DEVELOPER Ray Tracing (2023)
NVIDIA Corporation: NVIDIA DEVELOPER Ray Tracing, URL: https://developer.nvidia.com/discover/ray-tracing, 2023, visited on 10/04/2023

Open Source Robotics Foundation: ROS - Robot Operating System (2023)
Open Source Robotics Foundation: ROS - Robot Operating System, URL: https://www.ros.org/, 2023, visited on 09/25/2023

Pharr, M.; Jakob, W.; Humphreys, G.: Physically based rendering (2017)
Pharr, Matt; Jakob, Wenzel; Humphreys, Greg: Physically based rendering: from theory to implementation, Morgan Kaufmann Publishers/Elsevier, Cambridge MA, 2017

Piggott, A. Y.: Understanding the physics of coherent LiDAR (2022)
Piggott, Alexander Y.: Understanding the physics of coherent LiDAR, URL: http://arxiv.org/abs/2011.05313, 2022, visited on 05/06/2023

Rablau, C.: LIDAR - Introducing optics to broader engineering and non-engineering audiences (2019)
Rablau, Corneliu: LIDAR - A new (self-driving) vehicle for introducing optics to broader engineering and non-engineering audiences, in: Fifteenth Conference on Education and Training in Optics and Photonics: ETOP 2019, 2019

Rogers, C. et al.: A universal 3D imaging sensor on a silicon photonics platform (2021)
Rogers, Christopher; Piggott, Alexander Y.; Thomson, David J.; Wiser, Robert F.; Opris, Ion E.; Fortune, Steven A.; Compston, Andrew J.; Gondarenko, Alexander; Meng, Fanfan; Chen, Xia; Reed, Graham T.; Nicolaescu, Remus: A universal 3D imaging sensor on a silicon photonics platform, in: Nature, Vol. 590, pp. 256–261, 2021

Roriz, R. et al.: Automotive LiDAR Technology (2022)
Roriz, Ricardo; Cabral, Jorge; Gomes, Tiago: Automotive LiDAR Technology: A Survey, in: IEEE Transactions on Intelligent Transportation Systems, Vol. 23, pp. 6282–6297, 2022

Rosenberger, P.: Diss., Metrics for Simulation of Active Perception Sensor Systems (2022)
Rosenberger, Philipp: Metrics for Specification, Validation, and Uncertainty Prediction for Credibility in Simulation of Active Perception Sensor Systems, Dissertation Technische Universität Darmstadt, 2022

Rosenberger, P. et al.: Sequential lidar sensor system simulation (2020)
Rosenberger, Philipp; Holder, Martin Friedrich; Cianciaruso, Nicodemo; Aust, Philip; Tamm-Morschel, Jonas Franz; Linnhoff, Clemens; Winner, Hermann: Sequential lidar sensor system simulation: a modular approach for simulation-based safety validation of automated driving, in: Automotive and Engine Technology, Vol. 5, pp. 187–197, 2020

Rosenberger, P. et al.: Towards a Accepted Validation Methodology for Sensor Models (2019)
Rosenberger, Philipp; Wendler, Jan Timo; Holder, Martin; Linnhoff, Clemens; Berghöfer, Moritz; Winner, Hermann; Maurer, Markus: Towards a Generally Accepted Validation Methodology for Sensor Models - Challenges, Metrics, and First Results, in: 12th Grazer Symposium Virtuelles Fahrzeug (GSVF), 2019

SAE International: SAE-J3016 (2021)
SAE International: SAE-J3016: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles, URL: https://www.sae.org/standards/content/j3016_202104/, 2021, visited on 05/29/2023

Saleh, B. E. A.; Teich, M. C.: Fundamentals of photonics (1991)
Saleh, Bahaa E. A.; Teich, Malvin Carl: Fundamentals of photonics, Wiley series in pure and applied optics, Wiley, New York, 1991

Schlager, B. et al.: State-of-the-Art Sensor Models for Virtual Testing (2020)
Schlager, Birgit; Muckenhuber, Stefan; Schmidt, Simon; Holzer, Hannes; Rott, Relindis; Maier, Franz Michael; Saad, Kmeid; Kirchengast, Martin; Stettinger, Georg; Watzenig, Daniel; Ruebsam, Jonas: State-of-the-Art Sensor Models for Virtual Testing of Advanced Driver Assistance Systems/Autonomous Driving Functions, in: SAE International Journal of Connected and Automated Vehicles, Vol. 3, pp. 233–261, 2020

Thomas, P. et al.: Identifying the causes of road crashes in Europe (2013)
Thomas, Pete; Morris, Andrew; Talbot, Rachel; Fagerlind, Helen: Identifying the causes of road crashes in Europe, in: Annals of Advances in Automotive Medicine. Association for the Advancement of Automotive Medicine. Annual Scientific Conference, Vol. 57, pp. 13–22, 2013

Tsai, D. et al.: See Eye to Eye: A Lidar-Agnostic 3D Detection Framework (2022)
Tsai, Darren; Berrio, Julie Stephany; Shan, Mao; Worrall, Stewart; Nebot, Eduardo: See Eye to Eye: A Lidar-Agnostic 3D Detection Framework for Unsupervised Multi-Target Domain Adaptation, in: IEEE Robotics and Automation Letters, Vol. 7, pp. 7904–7911, 2022

Velasco-Hernandez, G. et al.: Autonomous Driving Architectures, Perception and Data Fusion (2020)
Velasco-Hernandez, Gustavo; Yeong, De Jong; Barry, John; Walsh, Joseph: Autonomous Driving Architectures, Perception and Data Fusion: A Review, in: 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 315–321, 2020

Viehof, M.: Diss., Objektive Qualitätsbewertung von Fahrdynamiksimulationen (2018)
Viehof, Michael: Objektive Qualitätsbewertung von Fahrdynamiksimulationen durch statistische Validierung, Dissertation Technische Universität Darmstadt, 2018

Voyles, I. T. et al.: Model Validation in the Presence of Aleatory and Epistemic Uncertainties (2015)
Voyles, Ian T.; Roy, Christopher J.: Evaluation of Model Validation Techniques in the Presence of Aleatory and Epistemic Input Uncertainties, in: 17th AIAA Non-Deterministic Approaches Conference, 2015

Wachenfeld, W. et al.: The Release of Autonomous Vehicles (2016)
Wachenfeld, Walther; Winner, Hermann: The Release of Autonomous Vehicles, in: Maurer, Markus; Gerdes, J. Christian; Lenz, Barbara; Winner, Hermann (ed.): Autonomous Driving, Springer, Berlin Heidelberg, 2016

Yeong, D. J. et al.: Sensor and Sensor Fusion Technology in Autonomous Vehicles (2021)
Yeong, De Jong; Velasco-Hernandez, Gustavo; Barry, John; Walsh, Joseph: Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review, in: MDPI Sensors, Vol. 21, p. 2140, 2021