SUNRISE

SAFETY ASSURANCE FRAMEWORK FOR CONNECTED, AUTOMATED MOBILITY SYSTEMS

D4.5

Report on the validated core features of the V&V simulation framework

Project short name
SUNRISE

Project full name
Safety assUraNce fRamework for connected, automated mobility SystEms

Horizon Research and Innovation Actions | Project No. 101069573
Call HORIZON-CL5-2021-D6-01

Funded by the European Union

ccam-sunrise-project.eu/

Dissemination level	Public (PU) - fully open
Work package	WP4: CCAM V&V framework
Deliverable number	D4.5: Report on the validated core features of the V&V simulation framework
Deliverable responsible	Dino Dodig, AVL
Status - Version	V1.0
Submission date	27/02/2025
Keywords	simulation toolchain, validation methodology, safety assurance framework, automated driving systems, verification and validation, use case requirements

Authors

Role	Name
Main author	Dino Dodig (AVL)
Contributing authors	Sarp Yetkin (AVL), Karlo Horvat (AVL), Philippe Nitsche (AVL) Gabriel Villalonga Pineda (CVC) Anastasia Bolovinou (ICCS), Ilias Panagiotopoulos (ICCS) Thaddaeus Menzel (IDI), Eloy Collado (IDI), Thomas Amler (IDI) Georg Stettinger (IFAG) Jobst Beckmann (IKA), Philipp Legran (IKA) Behrooz Sangchoolie (RISE), Mateen Malik (RISE), Ramana R. Avula (RISE) Hakim Mohellebi (RSA) Alperen Kiral (SISW), Mohsen Alirezaei (SISW), Tajinder Singh (SISW) Jason Zhang (UoW) Gerhard-B. Weiß (VIF), Bernhard Hillbrand (VIF), Martin Kirchengast (VIF)

Quality Control

	Name	Organisation	Date
Peer review 1	Marcos Nieto	VICOMTECH	11/02/2025
Peer review 2	Stefan de Vries	IDIADA	13/02/2025

Version history

Version	Date	Author	Summary of changes
0.1	12/09/2024	Dino Dodig (AVL)	First draft of document structure
0.2	12/09/2024	Gerhard-B. Weiß, Alperen Kiral, Behrooz Sangchoolie, Eloy Collado, Thaddaeus Menzel	Contents for validation methodologies A and C and usability of simulation toolchain
0.3	05/12/2024	All partners	Various updates by many partners. Filling the deliverable with text
0.4	10/12/2024	Dino Dodig (AVL)	Cleaning up changes and some legacy text. Re-arranging sections, updating relevant RQ for UC2.1 as an example, and starting suggestion for validation guideline for UC.
0.5	29/01/2025	Dino Dodig (AVL)	Structure updates
0.6	10/02/2025	Dino Dodig (AVL) All contributors	Overall text update, structure updates
0.7	20/02/2025	Dino Dodig (AVL) All contributors	Overall update on many points
0.8	26/02/2025	Dino Dodig (AVL)	Overall update on many points
1.0	27/02/2025	Dino Dodig (AVL)	Final version for submission

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND ACRONYMS

Abbreviation	Meaning
ABS	Anti-lock Braking System
AD	Automated Driving
ADAS	Advanced Driving Assistance System
ADE	Average Displacement Error
ADF	Automated Driving Function
ADS	Automated Driving Systems
AEB	Automatic Emergency Brake
ALKS	Automated Lane Keeping System
ASAM	Association for Standardization of Automation and Measuring Systems
C-ACC	Cooperative Adaptive Cruise Control
CAM	Cooperative Awareness Message
CCAM	Cooperative, Connected, and Automated Mobility
CFD	Computational Fluid Dynamics
CI/CD	Continuous Integration / Continuous Development
C-ITS	Cooperative Intelligent Transport Systems
CPM	Cooperative Perception Message
CPU	Central Processing Unit
C-V2X	Cellular Vehicle to Everything
DENM	Decentralized Environmental Notification Message
DOF	Degrees Of Freedom
DVM	Double Validation Metric

E-OL	Explicit open-loop simulation
ETSI	European Telecommunications Standards Institute
FDE	Final Displacement Error
FFT	Fast Fourier Transformation
FMI	Functional Mock-up Interface
FoV	Field of View
GLOSA	Green Light Optimal Speed Advisory
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPU	Graphics Processing Unit
GSR	General Safety Regulation
HW	Hardware
HWP	Highway Pilot
IAMTS	International Alliance for Mobility Testing and Standardization
IMU	Internal Measurement Unit
IoU	Intersection-over-Union
ISO	International Organization for Standardization
KPI	Key Performance Indicator
LiDAR	Light Detection and Ranging
MTF50	Modulation Transfer Function at 50%
NATM	New Assessment/Test Method
OD/BC	Operating Domain / Boundary Condition
ODD	Operational Design Domain
OECF	Opto-Electronic Conversion Function

OEM	Original Equipment Manufacturer
ORAD	On-Road Automated Driving
OSI	ASAM Open Simulation Interface
OSMP	OSI Sensor Model Packaging
PCC	Pearson Correlation Coefficient
PCD	Point Cloud Data
PLL	Phase-locked loop
PN	Phase noise
RAM	Random Access Memory
RDM	Range-Doppler map
RMSE	Root Mean Square Error
ROS	Robot Operating System
RX	Receiver
SAE	Society of Automotive Engineers
SAF	Safety Assurance Framework
SCDBs	Scenario Databases
SFR	Special Frequency Response
SNR	Signal-to-noise ratio
SOTIF	Safety Of The Intended Functionality
SPATEM	Signal Phase and Timing Extended Message
SUT	System Under Test
SW	Software
TTC	Time To Collision
TX	Transceiver

UC	Use Case
UNECE	United Nations Economic Commission for Europe
V&V	Verification and Validation
V2I	Vehicle-to-Infrastructure
V2N	Vehicle-to-Network
V2V	Vehicle-to-Vehicle
V2X	Vehicle–to-everything
VRU	Vulnerable Road User
XML	Extensible Markup Language

EXECUTIVE SUMMARY

Safety assurance of Cooperative, Connected, and Automated Mobility (CCAM) systems is a crucial factor for their successful adoption in society, yet it remains a significant challenge. It is generally acknowledged that for higher levels of automation, the validation of these systems by conventional test methods would be infeasible. Furthermore, certification initiatives worldwide struggle to define a harmonized safety assurance approach enabling massive deployment of CCAM systems.

The **SUNRISE** project develops and demonstrates a **CCAM Safety Assurance Framework** (SAF). The overall objective of the SUNRISE project is to accelerate the large-scale and safe deployment of CCAM systems. In alignment with international twin projects and initiatives, the project aims to achieve this objective by providing a SAF consisting of three main components: a Method, a Toolchain and a Data Framework. The **Method** is established to support the SAF safety argumentation, and includes procedures for scenario selection, sub-space creation, dynamic allocation to test instances and a variety of metrics and rating procedures. The **Toolchain** contains a set of tools for safety assessment of CCAM systems, including approaches for virtual, hybrid and physical testing. The **Data Framework** provides online access, connection and harmonization of external Scenario Databases (SCDBs), allowing its users to perform query-based extraction of safety relevant scenarios, allocation of selected scenarios to a variety of test environments, and reception of the test results.

This deliverable focuses on the **validation methodology of the harmonised V&V simulation framework** developed in T4.4. As elaborated in the following sections in more detail, one single Harmonised V&V Simulation Framework resulting from D4.4 [1] represents a common framework. From Harmonised V&V Simulation Framework, several simulation toolchains were derived, depending on SUNRISE defined use cases (as elaborated in [2]). Due to this **universal validation methodologies** were elaborated in this deliverable, section 3.

“Universal” in the context of validation methodologies, means it represents validation of different simulation toolchains that are derived from the Harmonised V&V Simulation Framework, and additionally, to other simulation toolchains outside of the SUNRISE project.

In this way, authors and partners believe that more stakeholders will be able to apply the suggested approach, primarily in the validation of virtual simulation toolchains as this is the focus of the deliverable, to different simulation architectures and projects in general.

To also cover the specific differences between simulation toolchains that have resulted from D4.4, **section 4** and **section 3** of this deliverable covers the specificalities of these toolchains by **giving specific guidance on** the recommended approach in **validating each simulation subsystem for the given toolchain** from D4.4.

1 INTRODUCTION

1.1 Project intro

Safety assurance of Connected, Cooperative, and Automated Mobility (CCAM) systems is a crucial factor for their successful adoption in society, yet it remains a significant challenge. CCAM systems need to demonstrate reliability in all driving scenarios, requiring robust safety argumentation. It is acknowledged that for higher levels of automation, the validation of these systems by means of real test-drives would be infeasible. In consequence, a carefully designed mixture of physical and virtual testing has emerged as a promising approach, with the virtual part bearing more significant weight for cost efficiency reasons.

Worldwide, several initiatives have started to develop test and assessment methods for Automated Driving (AD) functions. These initiatives already transitioned from conventional validation to a scenario-based approach and combine different test instances (physical and virtual testing) to avoid the million-mile issue.

The initiatives mentioned above, provide new approaches to CCAM validation, and many expert groups formed by different stakeholders, are already working on CCAM systems' testing and quality assurance. Nevertheless, the lack of a common European validation framework and homogeneity regarding validation procedures to ensure safety of these complex systems, hampers the safe and large-scale deployment of CCAM solutions. In this landscape, the role of standards is paramount in establishing common ground and providing technical guidance. However, standardising the entire pipeline of CCAM validation and assurance is in its infancy, as many of the standards are under development or have been very recently published and still need time to be synchronised and established as common practice.

Scenario Databases (SCDBs) are another issue tackled by several initiatives and projects, that generally tend to silo solutions. A clear concrete approach should be used (at least at European level), dealing with scenarios of any possible variations, including the creation, editing, parameterisation, storing, exporting, importing, etc. in a universally agreed manner.

Furthermore, validation methods and testing procedures still lack appropriate safety assessment criteria to build a robust safety case. These must be set and be valid for the whole parameter space of scenarios. Another level of complexity is added, due to regional differences in traffic rules, signs, actors and situations.

Evolving from the achievements obtained in HEADSTART[1] and taking other project initiatives as a baseline, it becomes necessary to move to the next level in the development and demonstration of a commonly accepted **Safety Assurance Framework** (**SAF**) for the safety validation of CCAM systems, including a broad portfolio of Use Cases (UCs) and

[1] The HEADSTART (Harmonised European Solutions for Testing Automated Road Transport) project is an EU funded project that aimed to define testing and validation procedures of Connected and Automated Driving functions including key technologies such as communications, cyber-security and positioning. (Link: Headstart - Headstart Project)

comprehensive test and validation tools. This will be done in **SUNRISE**, which stands for **S**afety ass**U**ra**N**ce f**R**amework for connected, automated mob**I**lity **S**yst**E**ms.

The SAF is the main product of the SUNRISE project. As the following figure indicates, it takes a central role, fulfilling the needs of different automotive stakeholders that all have their own interests in using it.

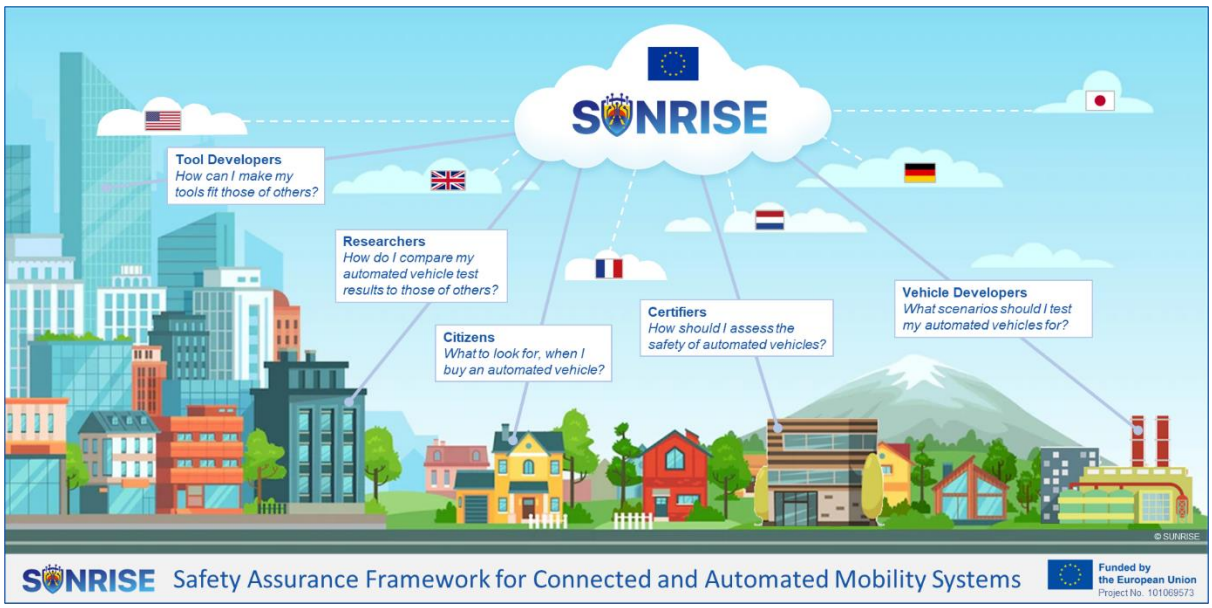


Figure 1: Safety Assurance Framework stakeholders

The **overall objective** of the SUNRISE project is to accelerate the safe deployment of innovative CCAM technologies and systems for passengers and goods by creating demonstrable and positive impact towards safety, specifically the EU's long-term goal of moving close to zero fatalities and serious injuries by 2050 (Vision Zero), and the resilience of (road) transport systems. The project aims to achieve this objective by providing a SAF consisting of three main components: a Method, a Toolchain and a Data Framework. The **Method** is established to support the SAF safety argumentation, and includes procedures for scenario selection, sub-space creation, dynamic allocation to test instances and a variety of metrics and rating procedures. The **Toolchain** contains a set of tools for safety assessment of CCAM systems, including approaches for virtual, hybrid and physical testing. The **Data Framework** provides online access, connection and harmonization of external Scenario Databases (SCDBs), allowing its users to perform query-based extraction of safety relevant scenarios, allocation of selected scenarios to a variety of test environments, and generation of the test results. The SAF will be put to the test by a series of **Use Cases demonstrations**, designed to identify and solve possible errors, gaps and improvements to the underlying methods, tools and data.

Following a common approach will be crucial for present and future activities regarding the testing and validation of CCAM systems, allowing to obtain results in a standardised way, to improve analysis and comparability, hence maximising the societal impact of the introduction of CCAM systems.

The following figure shows the general workplan of the SUNRISE project.

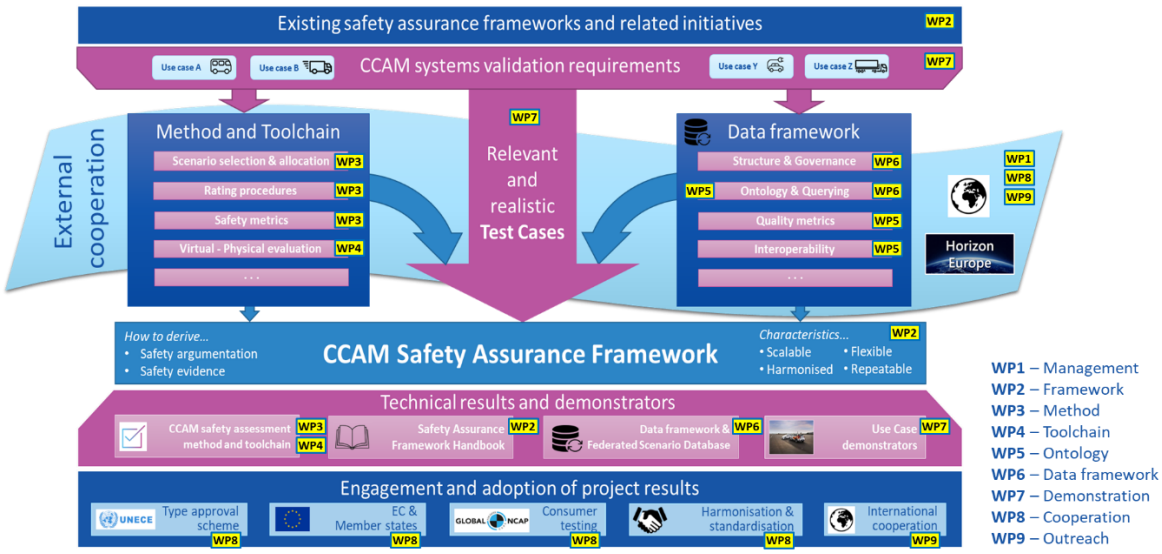


Figure 2: Workplan of the SUNRISE Project

1.2 Purpose of deliverable

Within SUNRISE project, this deliverable is part of the Work Package 4 (WP4) “CCAM V&V framework”.

WP4 aims to:

- Identify relevant subsystems
- Specify subsystem requirements
- Define and validate a toolchain containing:
  - V&V virtual simulation architecture
  - Virtual, hybrid and real-world testing, assessment and validation approaches

The following Table 1 outlines tasks within WP4

Table 1: Overview of WP4 tasks

Task	Name
4.1	Identification of relevant subsystems to virtually validate CCAM systems
4.2	Mapping of use case requirements to subsystems
4.3	Tooling landscape specification
4.4	Harmonized V&V simulation framework

4.5	Validation of the harmonized V&V simulation framework
4.6	Hybrid and real-world testing and validation approaches

This task 4.5 (T4.5) has, as the main purpose, to **provide validation methodology of harmonised V&V simulation framework** with the focus on virtual testing, while T4.6 is focusing on hybrid and real-world testing and validation approaches.

Deliverable 4.4 (D4.4) provided a Harmonized V&V Simulation Framework which resulted in several simulation toolchains to be used in the use case demonstrators in WP7 (as outlined in [1]), and it serves as the main input to T4.5 in which we provide a guidance on the approach to validate these simulation toolchains that originated from D4.4.

Additionally, the focus area of T4.5 is validation of simulation toolchains by virtual testing and is not intended to cover the validation of physical testing, that is the task for T4.6 with focus on hybrid and real-world testing validation approaches.

To further "localize" the purpose of T4.5, a schematic overview of the SUNRISE Safety Assurance Framework (SAF) is showcased on

Figure 3 below. T4.5 belongs to "Audit" block.

This "blueprint" is describing the methodologies and approaches in validating different simulation toolchains depending on the intended purpose of the given use case or target model fidelity.

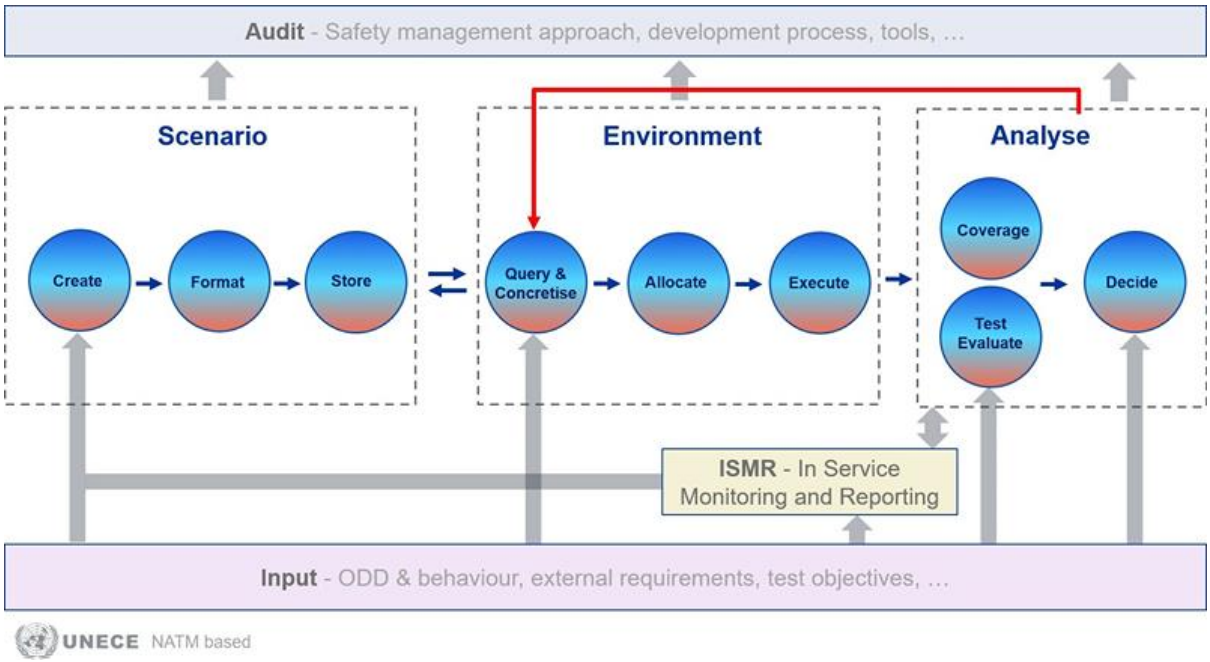


Figure 3: Draft of SUNRISE Safety Assurance Framework

To summarize the role of T4.5 in WP4:

- it takes input from D4.1 [3] as a basis for tooling landscape specification
- it takes inputs from D4.2 [4] and D7.1 [2] with respect to requirements of specific simulation sub-systems and particular use cases
- it takes the input from T4.4 in form of simulation toolchains that have resulted from Harmonised V&V Simulation Framework

T4.4 deliverable presented a Harmonized V&V Simulation Framework; this framework needs to be adjusted for each SUNRISE use case. This led to simulation toolchains that differ from use case to use case (More information on use cases can be seen in D7.1 [2]).

For these reasons, T4.5 is not providing a validation methodology for the Harmonised V&V Simulation Framework but it takes rather a universal approach. This universal approach covers various potential simulation toolchains that are relevant for SUNRISE but it also covers many different applications used in the industry.

The focus is not limited to universality. T4.5 is providing partners in Work Package 7, T7.3, with the correct approach in validating their simulation toolchain depending on their SUNRISE use case.

The resulting methodologies for validation of simulation toolchains are based on unique experiences of all contributing partners in applying these methodologies, but is also based on various industry standards, academic findings and already existing approaches with further expansion. One of the additional expansions of this deliverable is the intent to incorporate objective and subjective methodology in evaluating toolchain functionality, not just methodologies based on simulation toolchain (validation setups presented in section 3).

One additional goal that partners in T4.5 agreed on, is to keep the validation methodologies as toolchain independent as possible and not depend on any given simulation software promoting equal treatment and, in line with the spirit of EU funded projects, providing valuable input to internal as well as external stakeholders.

As intended, this deliverable will serve as an audit (part of “Audit” block in

Figure 3) of specific simulation toolchains and all relevant inputs that are essential in providing as accurate results as possible compared to measurement data, which is the main goal of any validation of simulation models.

Lastly and equally as important, this deliverable elaborates on standardized approach to interfaces, formats and measurements metrics. Although not a direct part of validation process, these standards can effectively streamline the validation process by ensuring interoperability, consistency, and accuracy, enabling seamless integration, efficient data exchange, and reliable comparisons.

1.3 Intended audience

The intended audience of the D4.5 is SUNRISE internal stakeholders in validating their simulation toolchains. The most direct internal beneficiary of the methodologies will be partners involved in T7.3 of the SUNRISE project where this approach can be used to develop validated simulation toolchains that, combined with other work packages and tasks, contribute in achieving successful result in SAF demonstration.

Key contents for internal stakeholders are located in section 5 with direct guidelines to validating their particular simulation toolchains for their specific SUNRISE use case.

Not only the internal stakeholders will benefit, but additionally the external stakeholders can benefit, by following the recommended validation approach to their specific simulation toolchains. OEMs, automotive suppliers and research institutions involved in developing simulation toolchains can achieve accurate simulation results, reduce cost and time-to-market.

Key contents for external stakeholders are located in section 3 with universal validation setups conceptualized to apply to many different virtual validation approaches.

1.4 Deliverable structure and relation to other parts of project

The content of the deliverable is divided into following sections:

- Section 2 – provides an overview of existing validation methodologies that were used as a base for this deliverable

- Section 3 – represents a core section within this deliverable as it describes the expanded methodologies as well as some novel aspects with respect to simulation functionality evaluation and metrics. Essentially, all of the methodologies described in this section are an expansion of the existing industry approach, but this section also outlines the importance of metrics for correlation analysis that are important when evaluating the results quality

- Section 3 also expands on the importance of toolchain functionality and capability during the process of development and validation.

  As simulation toolchains have become increasingly important in the validation process, they don't only impact the quality of simulation results or models they provide, but they impact the way users utilize the toolchain. Sections 3.7, 3.8, and 3.9 are dealing with important capabilities of these toolchains, objective as well as subjective metrics in evaluating and comparing different tools

- Section 4 – analyses and outlines a direct output from T4.2 with respect to use case requirements that are relevant to validation methodology and need to be thoroughly analysed by users to be able to satisfy these requirements while using appropriate validation methodology for their target output and calculations.

- Section 5 – the only section in the deliverable where specific toolchains from T4.4 are discussed and given guidelines in validating the approach for each use case specific simulation. It can be understood as a reference guidance blueprint to T7.3 in the validation efforts of simulation toolchains.

2 OVERVIEW OF VALIDATION METHODOLOGIES

The goal of this section is to provide an overview of existing validation methodologies that were used as a base for this deliverable, in particular, section 3.

For digital testing to be recognized not only as a development aid but also as a regulatory approval method, a significant hurdle is establishing trust in simulation methodologies and model accuracy. The goal of validating simulation toolchains, meaning primarily their simulation models, is to demonstrate their reliability for the specific scenarios they assess. Ongoing industry initiatives, such as those led by IAMTS [5] focus on developing structured validation frameworks for simulation tools and workflows, ensuring they are consistent, verifiable, and applicable across different markets.

Validation methodologies in context of simulation toolchain can be described as the process which aims to determine whether a simulation toolchain and/or a simulation model within the toolchain represents the real-world system or process it's intending to simulate. The methodology evaluates the toolchain/model outputs and behaviour to ensure their suitability, consistency and credibility.

Simulation-based validation is utilized across the entire automotive and automated driving ecosystem. Original Equipment Manufacturers (OEMs), including both vehicle producers and component suppliers, use simulation to test and refine vehicle dynamics, ADAS/AD functionalities, and overall system safety. Tier 1 suppliers, which develop critical automotive systems such as braking, steering, and perception modules, rely on virtual testing to ensure their products meet performance and safety standards before integration. Technology firms specializing in AI, sensor fusion, and autonomous driving software leverage simulation to validate complex algorithms and sensor interactions. Regulatory bodies and certification agencies apply simulation techniques to assess compliance with safety standards and streamline homologation processes. Research institutions and universities conduct studies on new validation methodologies, human behaviour modelling, and scenario-based testing. Additionally, independent test and validation service providers offer simulation-based verification to support automotive certification. By enabling efficient and cost-effective evaluation, simulation plays a vital role in ensuring the safety and reliability of automated systems before real-world deployment.

This current methodology (from [5]) can be illustrated by the figure below:

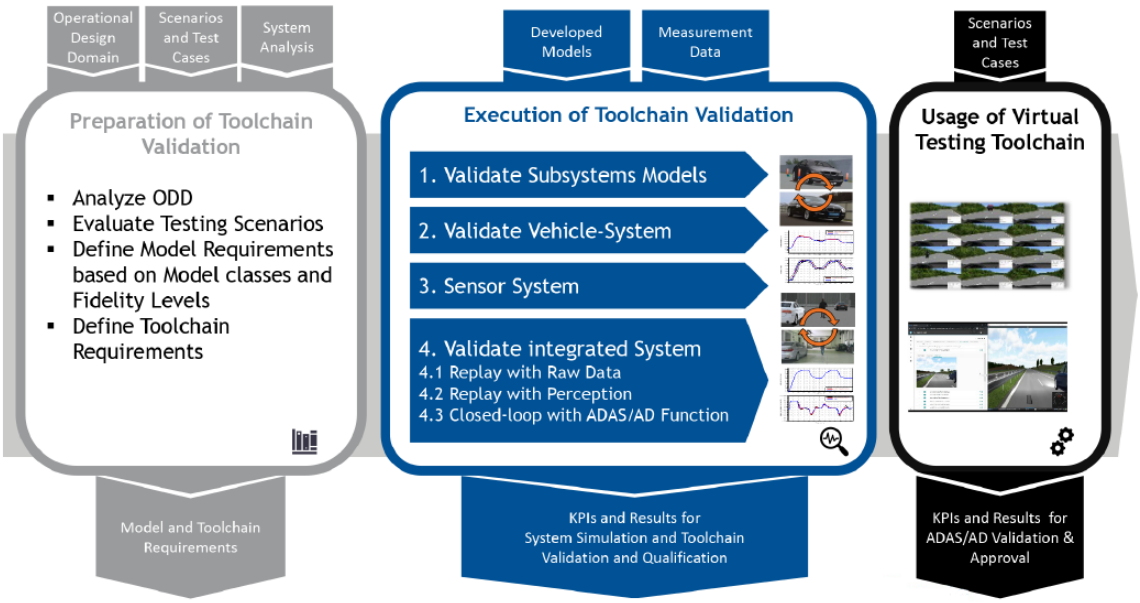


Figure 4: Basic simulation architecture [5]

Dueser, T. et al. [5] summarizes the simulation validation in 4 steps, starting with the validation of each separate sub-system model, then validating the vehicle system and then sensor system, finally, validation of the complete and integrated system completes the validation process.

Preparation of the validation process includes definition of all boundary conditions, mainly focusing on ODD definition that effectively covers the boundaries for all other simulation sub-models. Additionally, a minimum level of data from measurements should be provided.

Execution of the validation as mentioned can be summarized in 4 steps:

1. Validation of the sub-system models:
    - vehicle dynamics + environmental models + sensor models:
      separate validation of fidelity and requirements of each separate sub-system
    - sub-systems derived from [6] and [7]
2. Validation of the vehicle system:
    - Virtual vehicle with 3D road – deriving a digital twin:
      combined validation of vehicle dynamics and environmental models
    - Validation of longitudinal and lateral manoeuvres
3. Validation of the sensor system:
    - Validation of sensors integrated into the environment system
4. Validation of the Integrated systems:
    - Integrated simulation architecture as shown in Figure 5

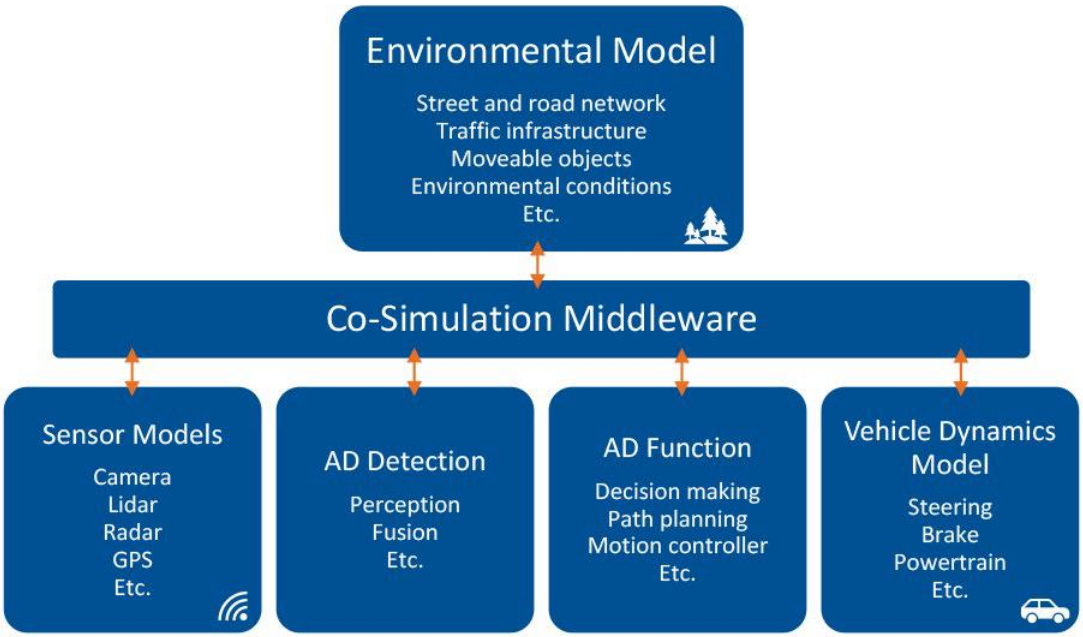


Figure 5: Simulation architecture with sub-model components from [5]

Figure 5 provides an overview of the simulation architecture components consisting of "**Environmental Model**", "**Sensor Models**", "**AD detection**", "**AD Function**" and "**Vehicle Dynamics Model**". The following section 3 of this deliverable will expand the approach from [5] by listing different validation setups that can form a particular simulation toolchain. In this context, validation setup represents a resulting simulation toolchain derived from different simulation sub-system models.

Further additions to validation methodologies are "Toolchain Functionality Methodology" in section 3.7, 3.8 and 3.9. Although not a validation methodology directly, it fits the section 3 where users are provided with important metrics that can make validation processes more robust and streamlined.

3 VALIDATION SETUP METHODOLOGY AND METRICS

With section 2 providing an overview of validation methodologies for a representative simulation architecture that corresponds to toolchain in this deliverable. This section also expands on methodology from section 2 by making it more universal and provide more details on different validation methodologies depending on the validation setup that users have.

Before an overview of the validation methodologies and validation setups given in this section, the Harmonised V&V Simulation Framework that resulted from D4.4 [1] is analysed on how is it fitting to the universal approach of defining validation setups that result in deferent simulation toolchains, including those derived from the SUNRISE project.

D4.1 [3] defined relevant subsystems to validate CCAM systems, D4.4 included those resulting subsystems in its proposed Harmonised V&V Simulation Framework [1]. Those subsystems are:

- Test case manager
- Environment
- Subject vehicle
- Traffic Agents
- Connectivity
- Simulation model validation

There are some inherent differences between the subsystems highlighted in D4.1 and D4.4 and subsystems provided in this deliverable, and they are discussed below.

We can observe on the left side of Figure 6, five different simulation sub-models that are forming a particular validation setup:

- **Vehicle model**
- **Environment model**
- **Sensor model**
- **Perception model**
- **Planning & control function model**

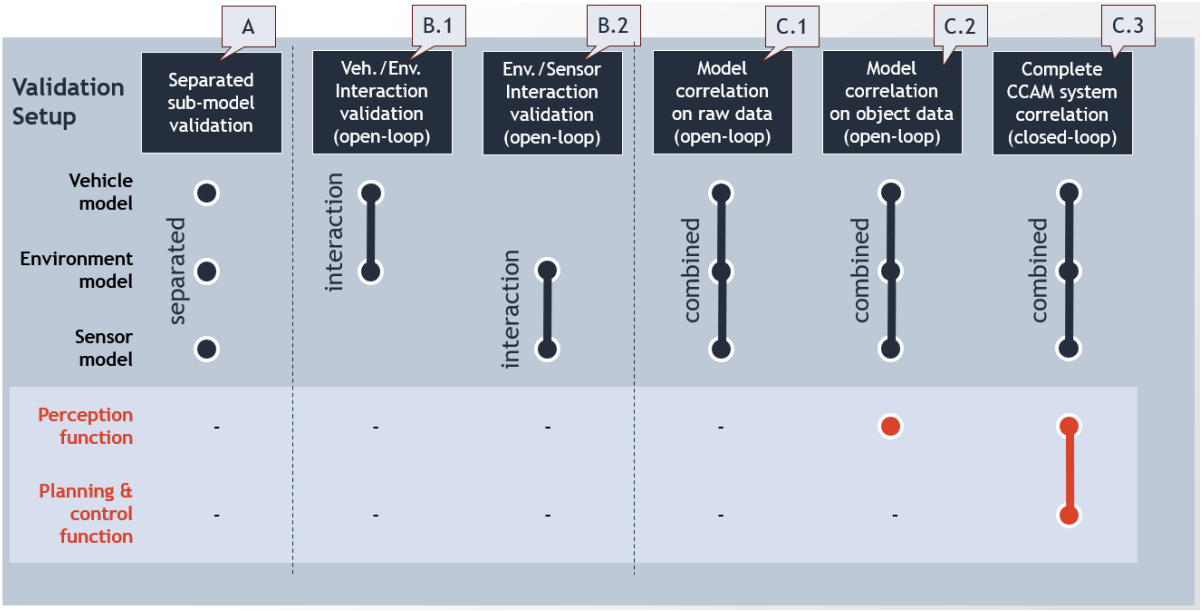


Figure 6: Schematic overview of the above-mentioned validation setups

Table 2 showcases a cross-comparison of D4.5 simulation sub-models vs D4.1 and Harmonised V&V Simulation Framework from D4.4.

Table 2: Difference in relevant subsystems for validation in T4.5 and T4.1

D4.1 subsystems & D4.4 Harmonised V&V Simulation Framework	**D4.5 sub-models relevant for the validation setups:**
Test case manager	This is **not validated directly** as validation is concerned with correlation to measurement data. We still elaborate on toolchain functionality that is relevant for test case manager as well (section 3.7, 3.8 and 3.9). D4.3 [5] is involved with the toolchain requirement analysis, if those are met, it is not envisioned to "validate" test case manager further.
Environment	**Environment** – same as D4.1/D4.4.
Subject vehicle	**Vehicle Dynamics** – contains all components as stated in D4.1/D4.4
Traffic agents	This is combined in the **Environment** sub-model, but content-wise it does include components as stated in D4.1/D4.4. Environment is usually split into static and dynamic part. It is a matter of simplification to include traffic agent into the environment
Connectivity	This is combined in the Environment sub-model as well, and throughout the deliverable, as well as in the evaluation of toolchain functionality. Connectivity

	does not represent a simulation sub-model eligible for validation, but more of a requirement or feature that needs to be either standardised or evaluated through all simulation sub-models
Simulation model validation	This subsystem is not a simulation sub-model appropriate for validation. The whole purpose of this deliverable is the process of validation approach, so it is not considered as an additional sub-model within this deliverable.

Harmonised V&V Simulation Framework cannot directly be validated with the approach proposed with this deliverable since it is a framework, not a strictly defined single toolchain containing models for validation. Out of the Harmonised V&V Simulation Framework, several different simulation toolchains have been derived from specific SUNRISE use cases (as described in D4.4 [1]). It needs to be noted, that while D4.4 defined the simulation toolchain for different use cases, and T7.3 incorporates these toolchains for demonstration purposes of SUNRISE SAF, it is not mandatory that T7.3 follows the exact approach as elaborated in this deliverable, although it is recommended where it's possible.

The final process of actual validation applied in T7.3 will be decided based on real circumstances and possibilities within WP7. As mentioned in

Figure 3, D4.5 provided a blueprint, a set of principles to follow, as universal approach applies to those simulation toolchains as well but is not mandatory to be strictly followed.

The idea behind the proposed validation setups showcased on Figure 6 is to make them as universal as possible, and they are intended to be tool independent to provide maximum contribution to internal and external stakeholders that can benefit from these methodologies.

Table 3 and Figure 6 are showing summarized descriptions of validation methodologies based on different validation setups. Different simulation toolchains consist of different simulation sub-models, and they are called validation setups.

These simulation models are the building blocks of a simulation toolchain depending on the simulation target.

Table 3 below showcases:

- Validation setup column – simulation model architecture based on different sub-models
- Method column – gives simulation toolchain approach
- Criteria column – provides context to the validation setup on sub-systems or metrics to validate

Table 3: Summary of validation setups (Adapted from [5])

	#	Validation Setup	Method	Criteria
A Sub-Model Validation	A.1	**Validate vehicle sub-models** (steering, braking, suspension, powertrain, tires etc.)	Open-Loop I/O Check and comparison with measurements	Torque map, steering ratio, tire dimensions, tire friction etc.
	A.2	**Sensor sub-models** (Radar, Lidar, Camera, ultrasonic, IMU, GNSS)	Open-Loop I/O-Check in a simple environment, comparison with measurements and spec-sheet	e.g., Camera: Pixel size, resolution, FoV, distortion, chromatic aberration etc.
	A.3	**Validate environment sub-models** (road, surface, lanes, traffic, road-side objects, weather etc.)	Visual inspection and comparison with ODD specification and/or real-world environment	Lane width, lane marking appearance, surface friction, curvature
B Model Interaction	B.1	**Validation of the integrated vehicle-environment co-simulation**	Open-loop driving maneuver simulation to correlate vehicle dynamics	Max. deceleration, braking distance, lateral accel., pitch angle etc.
	B.2	**Validation of integrated sensor-environment co-simulation**	Open-loop sensor simulation to correlate sensor data	Reflections, camera image, …
C System Integration	C.1	**Raw data replay and correlation** without perception and control function	Open-loop simulation for sampled test cases and comparison of raw sensor data	Raw sensor data correlation, chassis movements with sensors, simul. performance, repeatability
	C.2	**Replay and correlation with perception** system but without control function	Open-loop simulation for sampled test cases and comparison of object sensor data	Object list correlation, position and classification of objects
	C.3	**Replay and correlation with perception and control** function	Closed-loop simulation for checking the behavior of the vehicle compared to real-world	Control output correlation, e.g. driven trajectory, min. TTC, controller reaction times etc.

Table 3 provides a more visual description of validation setups that have been marked as:

- **Validation setup A** – validation approach where simulation toolchain consists of vehicle, environment and sensor simulation sub-models. In particular, the case where the validation approach of these sub-models is separated, meaning validation is carried out for each of these sub-models separately. This “separated” approach is either wanted by the user or is the limitation with respect to measurements data they might have for. This setup contains all three sub validation setups A1 (vehicle), A2 (sensor) and A3 (environment) which are forming validation setup A. They may l be used separately if needed (e.g. if sensors are not validated then A2 sub-model does not have to be evaluated).

- **Validation setup B.1** – validation approach where simulation toolchain consists of vehicle and environment simulation sub-models. In particular, the case where the validation approach of these sub-models is interacting, meaning validation is carried out for both sub-models integrated in an architecture. This is the most often used approach whenever a vehicle model is being validated, as most of the measurements used for correlation with the simulation data is already impacted within an environment (e.g. measuring vehicle on a test track, etc.).

- **Validation setup B.2** – validation approach where simulation toolchain consists of environment and sensor simulation sub-models. In particular, the case where the validation approach of these sub-models is interacting. This is the most often used

approach whenever a sensor model is being validated in a controlled environment/laboratory.

- **Validation setup C.1** – validation approach where simulation toolchain consists of vehicle, environment and sensor simulation sub-models. In particular, the case where the validation approach of all 3 of these sub-models is interacting. This is the option often used when raw data is available measured in interaction between each other.

- **Validation setup C.2** – validation approach for a setup like C.1, but additionally, a perception function is also being integrated and tested. Here, along with interacting vehicle, environment and sensor sub-models, a perception function is being tested on correlation from simulated perception object data with the measurements.

- **Validation setup C.3** – validation approach for a setup like C.1 and C.2 but additionally, a planning and controls function model is also being integrated and tested. Here, along with interacting vehicle, environment and sensor sub-models and perception function, planning and controls sub-model is being tested and correlated.

As Work Package 4 is also concerned with toolchain specification (T4.3 [8]) so is T4.5 as well. In addition to the above-mentioned validation setups, it considers additional methodologies and metrics deemed important as aspects of toolchains called toolchain functionality.

Three toolchain functionality methodologies are elaborated below, as they impact validation processes (although not directly validation). They are:

1. **Repeatability and determinism** – toolchain functionality methodology with respect to simulation results repeatability and determinism

2. **Scalability and performance** – toolchain functionality methodology with respect to scalability and performance of a particular simulation toolchain

3. **Useability** – toolchain functionality methodology with respect to toolchain useability

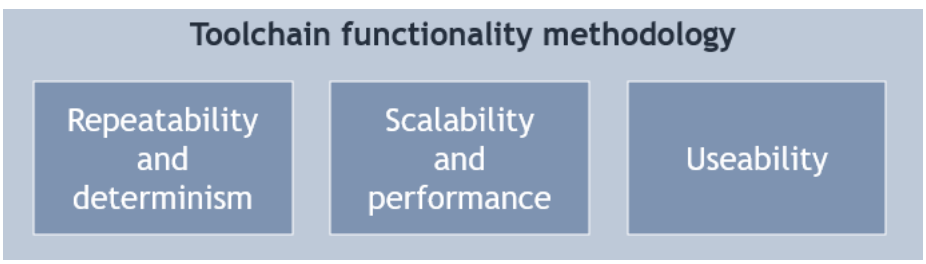


Figure 7: Summary of simulation functionality validation

3.1 Correlation Analysis

All validation methodologies outlined in Table 3 and are described below (section 3.2-0) share the same block at the very end, which is the correlation analysis. Within this step the result data of the simulations is compared to the real-world measurement data in order to validate the single models of the simulation toolchain. The output of such correlation analysis can always be used as a kind of feedback loop to iterate several times to improve the models. After each model modification the application of the methodology must be repeated which implies the repetition of the correlation analysis block.

The correlation analysis is the process of comparing the simulation results to real-world measurement data at the same operating conditions. Simulation and real test must follow the same test specifications if applicable.

The validation criteria must fulfil the accuracy requirements, means to compare the results against predefined acceptance criteria. Depending on model fidelity, for the most relevant subset of output signals validation criteria are defined as upper bounds on the chosen accuracy metrics. The weighted sum of individual (and normalized) validity metrics can be used to get a lumped validity number.

3.1.1 Point data metrics for correlation analysis

Different metrics are applicable for various types of underlying data. The choice of metric depends on the data type and specific validation requirements.

Standard Methods

Vector norms (mainly 1-, 2-, and infinity-norm), the related mean absolute, and mean squared errors, average residual and standard deviation. They are easy to compute but do not distinguish between magnitude and phase errors and provide no measure for the similarity of shape.

Normalized minimum Euclidian distance

A typical validation metric to determine correlation levels between simulation and the real world is the normalized minimum Euclidian distance between point clouds [9] defined as:

$$\frac{1}{N}\sum_{i}^{N} \min\left\| PC_{sim,i} - PC_{rw,i} \right\|,$$

where N represents the total number of rays per scan, and the availability of the point cloud (PC) for the simulation (sim) and the real world (rw) is assumed. Additional methods for assessing the correlation on raw data level are given in [10].

Metrics conducted by Rosenberger in 2022 [11] **and summarized by Elster et al.** [12]

A detailed evaluation of 34 different metric candidates used for perception sensor model validation in literature was conducted by Rosenberger in 2022 [11]. Table 4 presents an excerpt of these metrics, as considered and summarized by Elster et al. [12] with the addition of the Mahalanobis distance compared to the original table.

Table 4: Summary from Elster et al. [12] on the evaluation of metrics used for active perception sensor simulation, as discussed by Rosenberger [11, p. 99]

Metric	Interf.		Scen.		Scale		Unc.		Covered criteria						
	D	O	•	⇝	M	O	∫	⊢	1	2	3	4	5	6	7
d_{Ma} (Manhattan distance)		x		x	x				■	■	■	■			■
OE (Overall error)	⋆		x	x	x				■	■	■	■			■
$\overline{d}$ (Mean error)	x			x	x				■	■	■	■			■
RMSE (Root mean square error)		x	x	x	x				■	■	■	■			■
d_{M} (weighted Mahalanobis distance)	x								■	■	■		■		■
D_{KL} (Kullback–Leibler divergence)	⋆	x	x	x	x		x				■	■	■		
d_{JS} (Jensen-Shannon distance)	x			x	x		x				■	■	■		
d_{AVM} (Area validation metric)	⋆	x		x	x		x		■	■	■	■	■		■
f_{KS} (Kolmogorov-Smirnov)	x			x	x		x			■		■	■		
DEM (Deep evaluation metric)	x			x	x							■			

A key requirement for validation metrics is their ability to differentiate between model bias and model scattering error, facilitating the systematic elimination of these distinct modelling errors [11, p. 72]. Model bias represents an approximation of the mean deviation, while model scattering error reflects deviations in the shape of the distribution function. The Figure 8 highlights the distinctions between model and measurement bias and compares the measurement standard deviation with model scattering error. While measurement bias and scattering error resemble the differences in mean and variance of normal distributions, the distribution functions of the measurand may diverge from normal distributions. To address this, Rosenberger introduced the so-called Double Validation Metric (DVM), which separates these two components. The DVM is based on the on the Area Validation Metric (AVM) from Ferson et al. [13]. Due to its level of complexity, it is not further explained here but can be found in [11, p. 118].

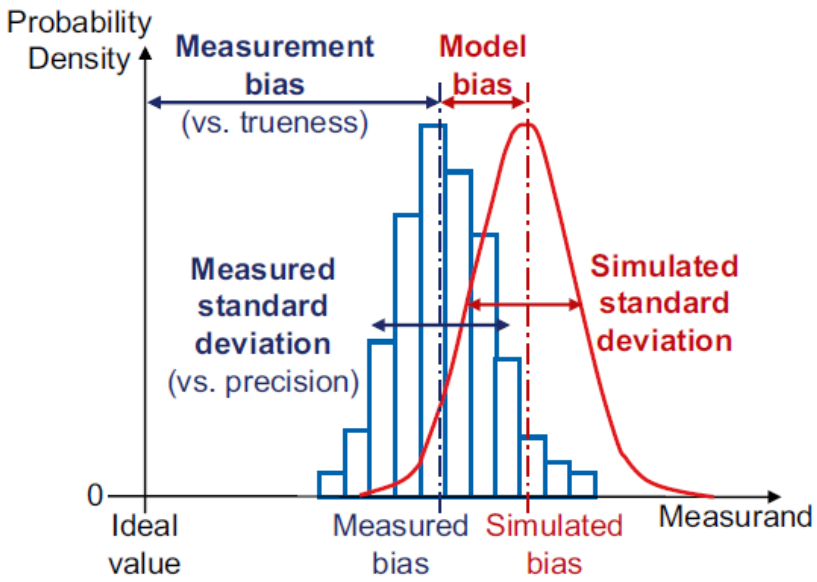


Figure 8: Bias and scattering of measurements and models [11, p. 11]. For normal distributions, these two factors correspond to the mean and standard deviation.

3.1.2 Time data metrics for correlation analysis

Distance metrics for one-dimensional time series data do not differ from distance metrics used for vectors (like the Manhattan metric).

Multivariate time series data can be converted to point cloud data and the before mentioned metrics for point data can be applied. But the loss of any temporal context should be considered in this case.

In the case where the data we compare have been produced by different models because it is not always easy to simulate a SuT (e.g. this is usually the case with sensor models in simulation and in real-world), a lighter correlation argument could be established by using Kendall's Tau Rank Correlation: this is a non-parametric statistical method used to measure the strength and direction of association between two variables, which unlike Pearson correlation, does not assume that the data follows a specific distribution, making it suitable for both continuous and ranked (ordinal) data.

Absolute or relative tolerance intervals

A standard method for validation is introducing an absolute or relative tolerance interval around the experimental reference data. Tolerance intervals are representative of the measurement and model's uncertainty. For example, a validation assessment procedure can require the virtual model's output to stay within the 5% amplitude interval of the corresponding real-world realization.

Correlation coefficient (Pearson coefficient!)

Time-histories distance analysis can also be supported by the computation of the Pearson correlation coefficient to detect the extent to which the model stays in a linear relationship with the real data. The coefficient of correlation indicates to which extent one time series is a linear

representation of another one, though it does not distinguish between magnitude and phase error either.

Cross correlation

The cross-correlation computes the linear relationship between two time series, one of them being time-shifted by n=0, 1, …, N-1 discrete time steps (N is the signal length). Maximizing the cross-correlation gives an estimate of the phase lag n*.

Normalized Integral Square Error (NISE)

Based on the cross-correlation, the summands of the Normalized Integral Square Error evaluate magnitude, phase, and shape errors ([14] and [15])

IAPE Method

This method is based on using a pair of curves (the real signal test curve and the simulated signal curve) and allows, through various indicators (I, A, P, E), to produce a scalar value between 0 and 1 (with 1 corresponding to perfect correlation) that characterizes the quality of the correlation. This method has been used and approved in the works on the AEB system (Automatic Emergency Braking) and is also applicable for the LSS system (Lane Support Systems). The different indicators I, A, P, and E are defined as follows:

Criterion of peak timing appearance (I)

This criterion is used to compare the timing of the max & min peaks between two curves f and. It is defined by the following formula:

$$I = \frac{\max(0, T_{f\_extr} * T_{g\_extr})}{\max({T_{f\_extr}}^2, {T_{g\_extr}}^2)}$$

Where T_{f_extr} is the moment when f_etr reaches its maximum and T_{g_extr} is the moment when g_extr reaches its maximum.

Criterion of peak amplitude (A)

This criterion is used to compare the amplitudes of the extreme peaks (max and min) of the two curves *f* and *g*. It is defined by the following formula:

$$A = \frac{\max(0, f_{extr} * g_{extr})}{\max({f_{extr}}^2, {g_{extr}}^2)}$$

Curve profile criterion (P)

This criterion is used to calculate the mean squared error at each point. It is defined by the following formula:

$$P = 1 - \sqrt{\frac{\sum_n \max(|f(t_n)|, |g(t_n)|).\left(1 - \frac{\max(0, |f(t_n)| * |g(t_n)|)}{\max(\delta, f(t_n)^2, g(t_n)^2)}\right)^2}{\sum_n \max(|f(t_n)|, |g(t_n)|)}}$$

Where δ is a small value compared to 1, is the n^{th} point of curve, and the nth point of curve.

Error criterion (E)

This criterion is used to calculate the integral of the error over the duration of the two curves. It is defined by the following formula:

$$E1 = \exp\left(-\cdot \frac{1}{t_{max} - t_{min}} \cdot \int_{t_{min}}^{t_{max}} \left|\frac{f(t) - g(t)}{\max(|f(t)|, |g(t)|)}\right| dt\right)$$

$$E2 = \exp\left(-\cdot \frac{1}{t_{max} - t_{min}} \cdot \left|\int_{t_{min}}^{t_{max}} \frac{f(t) - g(t)}{\max(|f(t)|, |g(t)|)} dt\right|\right)$$

$$E = 1 - \sqrt{\frac{(1 - E1)^2 + (1 - E2)^2}{2}}$$

The advantage of the IAPE method is that it is understandable by domain experts and is already used and approved as an empirical comparison method in AEB studies. These indicators (I, A, P, E) can also be enhanced with additional metrics used in time series comparison, providing complementary information not measured by the IAPE method.

Advanced Metrics: Discrete Fréchet Distance (1):

The discrete Fréchet distance, also known as the coupling distance, is a measure of similarity that takes into account the position and order of points along the curves. In fact, it is an approximation of the Fréchet distance for polygonal curves. It can be defined as the distance that corresponds to the minimum (inf) of all possible couplings between the points/vertices of the two curves, aiming to minimize the maximum length of the distance between them. The advantage of using this metric is that the Fréchet distance is discrete and can be calculated in polynomial time using dynamic programming. The time complexity of this algorithm is where n and m are the lengths of the time series.

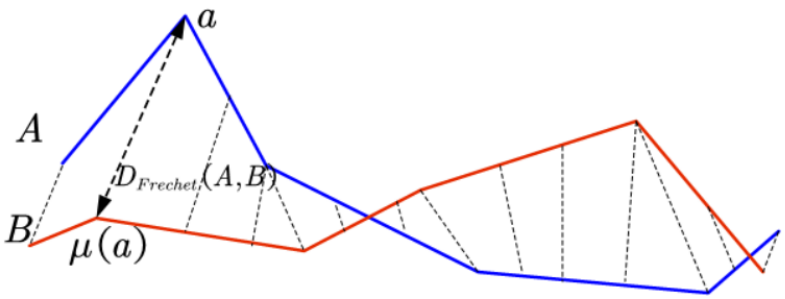


Figure 9: Discrete Fréchet distance

Advanced Metrics: Dynamic Time Warping (DTW) (2):

Dynamic Time Warping is one of the algorithms used to measure the similarity between two time series (signals/curves) and to calculate the optimal mapping between the two, considering the temporal distortion between the series. It resolves the issue of time series with different lengths by developing a “one-to-many” mapping so that peaks and troughs with the same “pattern” are perfectly aligned. Certain conditions are required for the application of DTW and are defined below:

- Each index of the first series must correspond to one or more indices of the other series and vice versa.
- The first index of the first series must correspond to the first index of the second series (but it does not need to be the only correspondence).
- The last index of the first series must correspond to the last index of the second series (but it does not need to be the only correspondence).
- The mapping of indices from the first series to the indices of the second series must be monotonically increasing, and vice versa.

The time complexity of this algorithm is O (n*m), where n and m are the lengths of the time series.

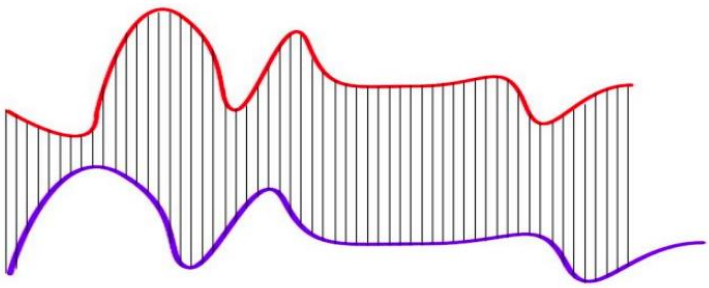


Figure 10: Lockstep (Euclidean) alignment

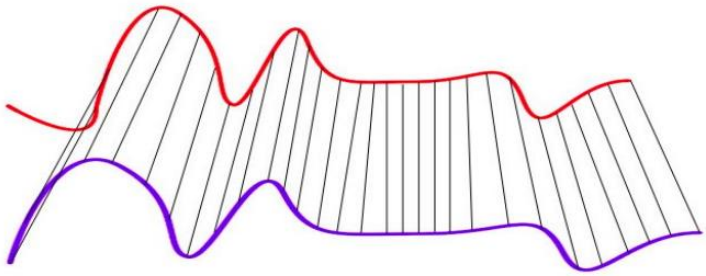


Figure 11: DTW alignment

Advanced Metrics: Frequency Spectrum Error Criterion:

We complement the metrics with an error calculation criterion based on the frequency spectrum of the two signals. First, we perform the Fourier transform for both time series, and then we calculate the root mean square error (RMSE) on their normalized magnitudes. This indicator captures the dynamics of both signals.

Advanced Metrics: Spearman's Cross-Correlation:

The Spearman correlation coefficient [16] is a non-parametric statistical measure of dependence between the ranks of two variables (two time series). In other words, it answers the question of how one variable changes as the other changes. It can be defined as the Pearson correlation coefficient (PCC) applied to the ranks of the two signals.

After the values of the two series and are converted into ranked values and, the Spearman correlation coefficient can be defined as follows:

$$\rho_{(r_f,r_g)} = \frac{cov(r_f, r_g)}{\sigma_{r_f} \cdot \sigma_{r_g}}$$

Where the covariance between the ranks and r_f, and r_g are the standard deviations of the ranks δr_f and δr_g.

Below is a summary table of the previously described indicators:

Table 5: Summary of the above-mentioned indicators

Indicators:	Description:
I	Criterion for the appearance of peak moments
A	Criterion for peak amplitudes

P	Criterion for the curve profile
E	Error criterion
Discreet Fréchet distance	Discrete Fréchet distance
DTW	Dynamic Time Warping
Frequency components comparison error	Frequency spectrum error criterion
Spearman correlation	Spearman cross-correlation

3.2 Methodology for Validation Setup A

Separated sub-model validation includes the validation of the vehicle, environment and the sensor model. Validating sensor, environment, and vehicle sub-models involves different focusses and methodologies due to their distinct roles within a simulation system.

Vehicle Sub-Model simulates the dynamics and behaviour of the entire vehicle, including its interactions with the environment. **Sensor Sub-Model** simulates the behaviour of sensors (e.g., cameras, radar, LiDAR, ultrasonic) to generate synthetic sensor data. **Environment Sub-Model** simulates the environment in which the vehicle operates, including road conditions, weather, and obstacles.

Table 6: Validation setup “A” sub-model methodology split

Validation Setup A	Description
A.1	**Validate vehicle sub-models**: steering, braking, suspension, powertrain, tires etc. **Method**: Open-Loop I/O Check and comparison with measurements **Criteria Examples**: Torque map, steering ratio, tire dimensions, tire friction etc.
A.2	**Sensor sub-models**: Radar, Lidar, Camera, ultrasonic, IMU, GNSS **Method**: Open-Loop I/O-Check in a simple environment, comparison with measurements and spec-sheet **Criteria Examples**: (e.g. Camera: Pixel size, resolution, FoV, distortion, chromatic aberration etc, sharpness, colour accuracy)
A.3	**Environment** sub-models: road, surface, lanes, traffic, road-side objects, weather etc. **Method**: Visual inspection and comparison with ODD specification and/or real-world environment **Criteria Examples**: Lane width, lane marking appearance, surface friction, curvature

3.2.1 Sub-model validation – vehicle model

Validating a digital vehicle model built in a software tool involves ensuring that the model meets the intended design specifications, performs as expected, and is accurate in its representation of the physical vehicle or system it simulates. The process includes several stages of validation, from initial design verification to performance testing and final validation. A digital twin in the automotive industry generates a virtual image of an actual vehicle, allowing for real-time study and simulation. A virtual digital twin represents the real vehicle in the digital world.

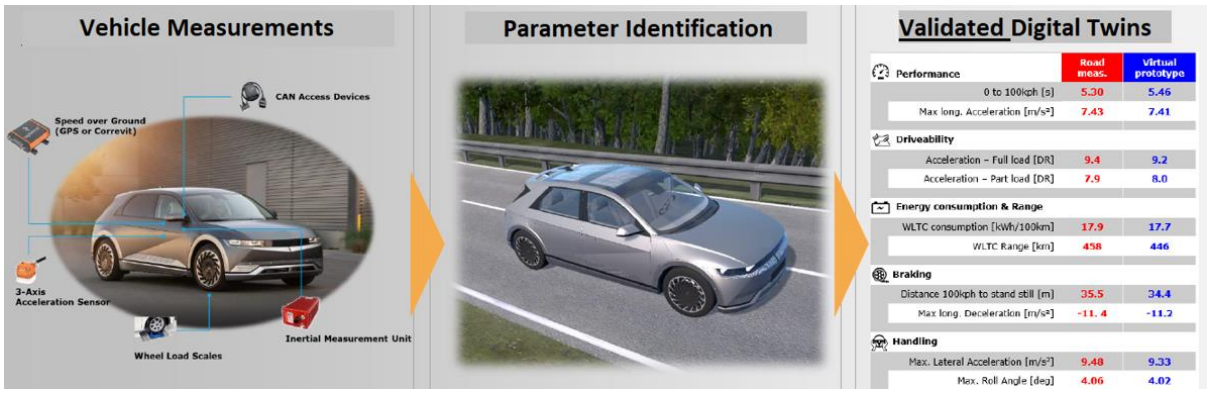


Figure 12: Example of vehicle sub-model validation by creating a digital twin

Model validation depends on some parameters. It starts with obtaining vehicle measurement data. The methods for measuring and saving vehicle data involve a combination of sensors, onboard computing, data logging, telematics, and cloud-based solutions. These systems work together to monitor vehicle performance, ensure safety, and provide valuable insights into vehicle operations and efficiency.

From these measurements, vehicle parameters can be determined, which are used in the development of certain vehicle components. Vehicle components are observed and validated separately (**Chasis** (Geometry, Aerodynamics, Brakes, Cabin), **Suspension** (Steering, Wheel setup), **Tyres** (Grip, Friction), **Springs** / **Dampers**, **Bumpstops**, **Drivetrain** (Setup, Gearbox, Clutch, Engine Mounting), **Engine** / Electrical System, Driving losses). The validation of a vehicle model's sub-models requires a detailed and systematic approach for each subsystem.

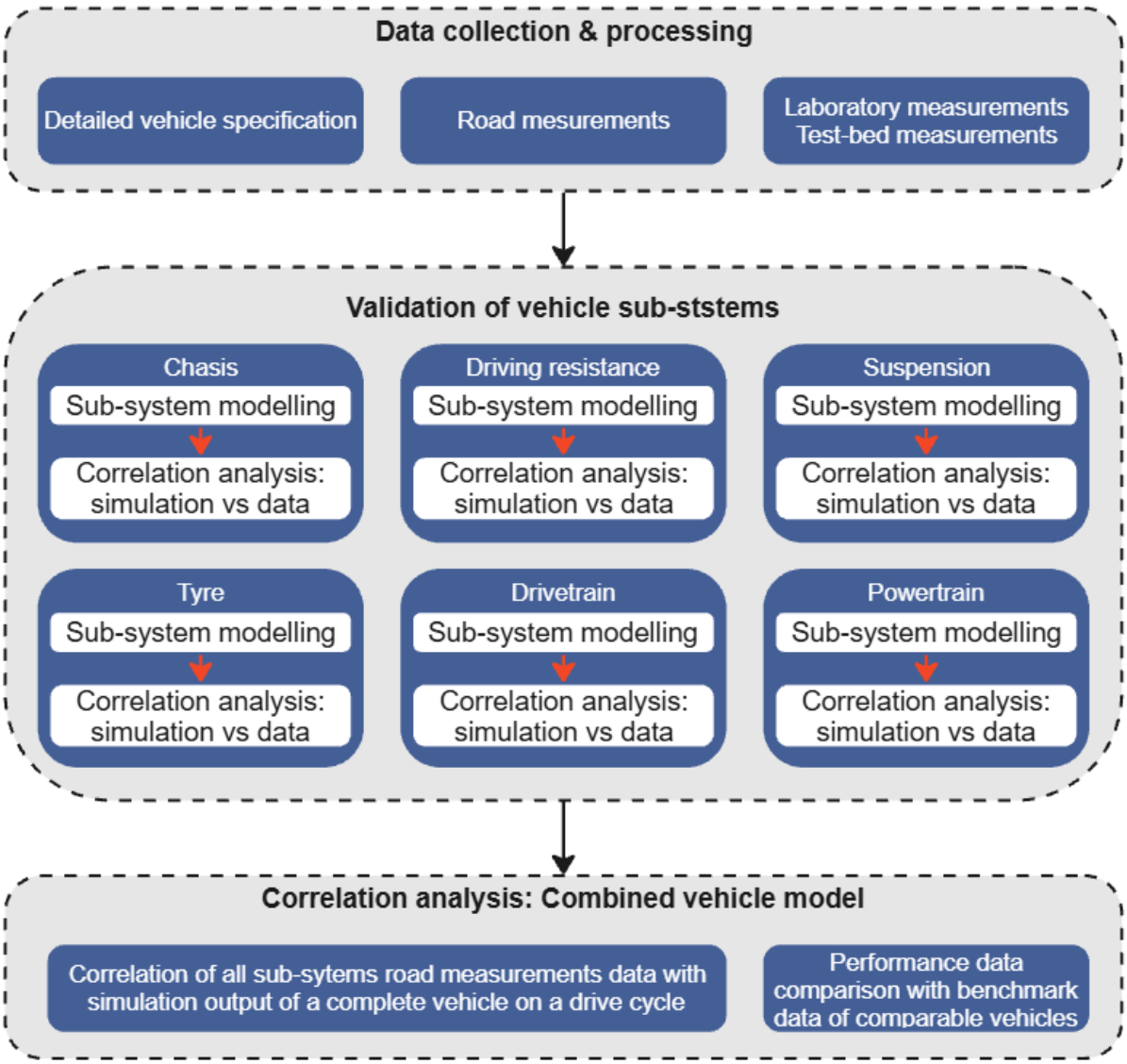


Figure 13: Schematic overview of vehicle sub-systems validation flow

The **chassis sub-model** represents the structural framework of the vehicle, affecting weight distribution, rigidity, and overall dynamics. Validation involves comparing the mass, center of gravity, and inertia properties with physical measurements. Simulations of dynamic events, such as acceleration, braking, and cornering, help ensure that the chassis reacts as expected. For more advanced models, structural deformation and stress distribution under extreme conditions are also validated. Aerodynamics is verified using wind tunnel data or computational fluid dynamics (CFD) simulations, ensuring the drag coefficient, lift forces, and yaw effects match real-world tests. Braking performance is validated by simulating stopping distances, brake force distribution, and anti-lock braking system (ABS) behavior under various road and load conditions. Validation focuses on ensuring that the braking force distribution between front and rear wheels matches physical behavior under different load and road conditions. Wheel slip and stopping distances should align with real-world data, and ABS behavior can be validated by simulating emergency braking on low-friction surfaces. For electric vehicles, the regenerative braking energy recovery should be tested for accuracy. The

cabin model can be validated for NVH (noise, vibration, and harshness) characteristics by comparing simulated in-cabin noise and vibrations with physical test data.

The **suspension system**, including steering and wheel setup, requires validation to confirm accurate handling and stability. Steering responses are validated by measuring yaw rate, lateral acceleration, and steering feedback during dynamic manoeuvres such as lane changes or cornering. The wheel setup is verified through kinematic analysis, ensuring parameters like camber, toe, and caster angles match physical measurements during suspension travel and compliance testing.

The **tire sub-model** represents the interaction between the tires and the road surface, affecting traction, handling, and stability. This model is validated by comparing tire forces, slip angles, and vertical loads with data from tire testing machines or real-world measurements. Longitudinal and lateral dynamics are assessed using standard tire modelling methods, such as Magic Formula, and validated on various road conditions, including wet or icy surfaces.

Springs and dampers are validated by measuring suspension travel, damping force, and stiffness during transient and steady-state conditions, such as bump tests or dynamic handling manoeuvres. Bump stops are validated by ensuring they accurately limit suspension travel under extreme loading conditions, with simulated force-displacement curves matching experimental results.

The **drivetrain**, which includes setup, gearbox, clutch, and engine mounting, is validated by ensuring accurate torque transmission and dynamic response. Gear ratios, shift points, and efficiency are tested under varying load conditions to match dynamometer test results. The clutch model is validated for engagement and disengagement behaviour during gear shifts or transient torque events. Engine mounting is tested for accurate vibration isolation and torque reaction under real-world operating conditions.

The **engine and electrical** system are validated by comparing the engine's power, torque, and fuel or energy consumption maps with experimental data from engine test benches. Electrical system components, such as batteries and power management systems, are validated for energy flow, thermal performance, and regenerative efficiency. Electrical losses and system responsiveness are compared with measured results from actual vehicle operation.

Driving losses, which include frictional and aerodynamic losses, are validated by comparing the simulated energy consumption and drag forces during real-world driving cycles with test data. Validation ensures that rolling resistance, drivetrain friction, and aerodynamic drag are modelled accurately for different speeds and conditions.

Each of these sub-models must be tested separately under controlled conditions, with simulated results compared against experimental or real-world data to ensure that they accurately represent their real-world counterparts. The process involves an iterative cycle of adjustments and comparisons to refine the model and achieve a high degree of reliability for each subsystem.

In the end a system (digital twin) is obtained that functions in accordance with the measurement data obtained. Finally, the measured results from the vehicle are compared with the results from the simulation.

3.2.1.1 Standalone Vehicle vs Vehicle + Environment model

Validating a vehicle model in experimental (laboratory, test-bed, etc.) conditions differs significantly from validating it using real-world data, primarily due to the level of control, complexity, and purpose of each method. Experimental validation focuses on testing individual sub-models or components of the vehicle in a controlled and repeatable environment. This approach isolates specific systems to ensure their behaviour matches expectations under predefined inputs, removing external influences such as environmental factors or dynamic interactions. For example, the powertrain might be tested on a dynamometer to measure torque, power output, and efficiency under controlled loads and speeds. Similarly, aerodynamics may be validated in a wind tunnel, where airflow can be precisely regulated to measure drag and lift coefficients. These tests are highly repeatable and provide detailed data specific to the subsystem being evaluated, enabling precise calibration of the model.

In contrast, real-world validation evaluates the performance of the complete vehicle as it interacts with its environment, including road surfaces, weather conditions, and traffic scenarios. This method accounts for the dynamic and often unpredictable nature of external factors that cannot be fully replicated in experimental setups. For instance, the performance of a suspension system tested on a rig may behave differently when exposed to real-world road conditions with varying surfaces and irregularities. Similarly, aerodynamic validation in real-world scenarios must consider crosswinds, turbulence from other vehicles, and temperature changes, which affect the vehicle's behaviour in ways that controlled wind tunnel testing cannot fully capture.

The iterative nature of validation also differs. Experimental validation allows for fine-tuning of individual sub-models by comparing their outputs to measured data in isolation, which is useful for identifying and correcting specific inaccuracies. Real-world validation, on the other hand, focuses on assessing the overall vehicle performance and identifying issues arising from the interaction of subsystems. For example, a drivetrain model validated on a test bench might perform optimally in isolation, but in a real-world driving cycle, interactions with the suspension, tires, and road conditions may expose inefficiencies or undesirable behaviours.

In summary, experimental validation is ideal for precise, isolated testing of individual systems under controlled conditions, ensuring their fundamental accuracy and reliability. Real-world validation, on the other hand, provides a holistic assessment of the vehicle's behaviour in dynamic, real-world environments, capturing the complexity and variability of system interactions and external influences. Both approaches are complementary, with experimental validation providing a foundation of accuracy for sub-models and real-world validation ensuring the model’s robustness and applicability in practical scenarios.

Powertrain

An example of parameter identification & validation is Powertrain. A vehicle powertrain consists of all the components that generate power and deliver it to the road, enabling the vehicle to move. A typical powertrain system consists of engine, transmission, driveshaft, differential and axles. Since there are several components, each must be modelled separately inside of a software tool.

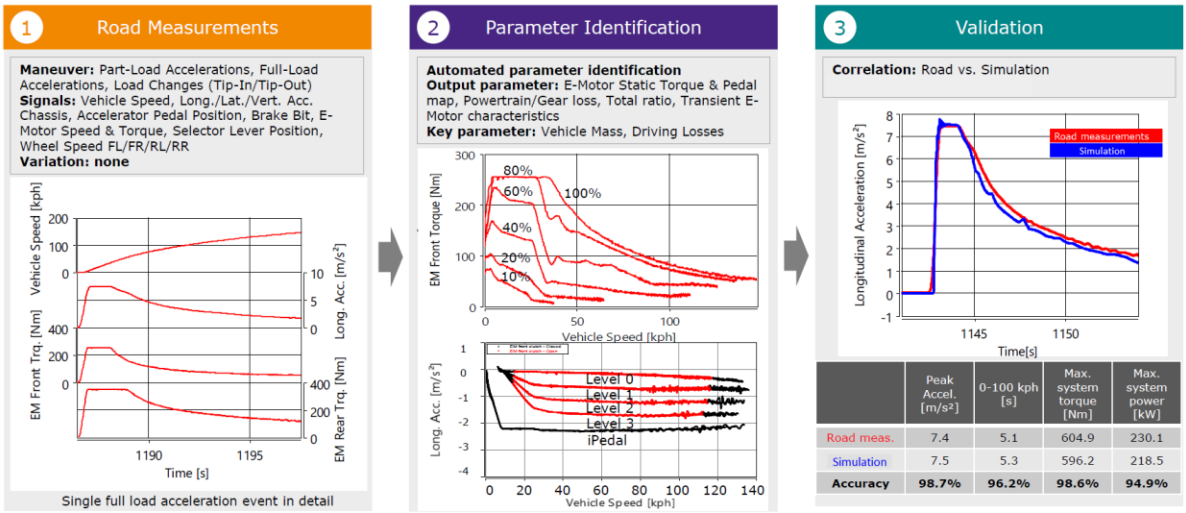
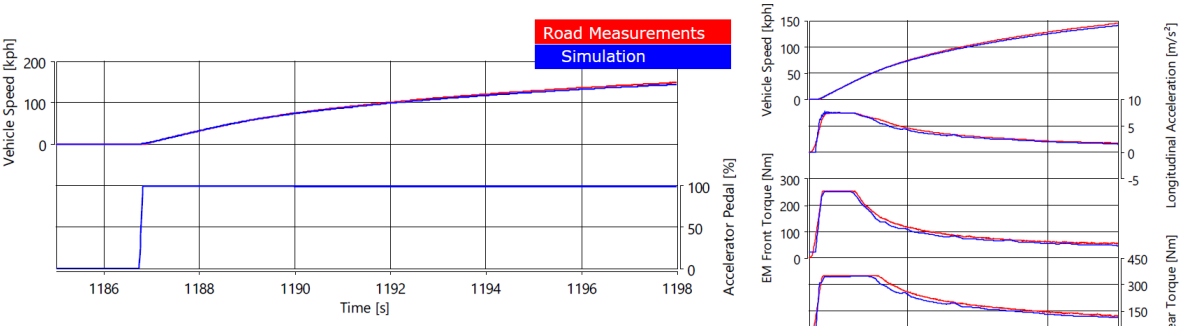


Figure 14: Example of powertrain model validation flow

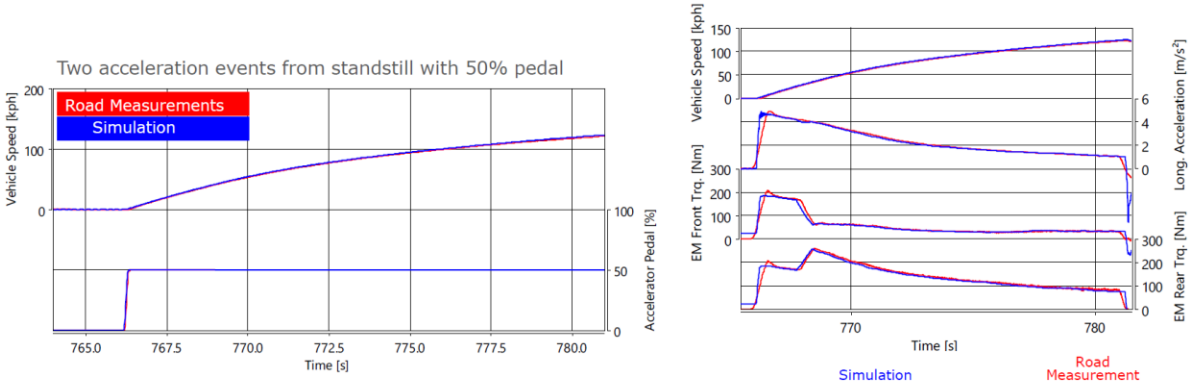
Figure below shows the validation results with a Full Load Acceleration.



KPI	Road Measurement	Simulation	Accuracy
Acceleration – Full Load [DR]	9.4	9.2	97.9%
Wide torque range [DR]	9.9	9.9	100%

Figure 15: Powertrain - Correlation of measured data with simulation output full load, time-based data

Figure below shows the validation results with a 50% Part Load Acceleration.



KPI	Road Measurement	Simulation	Accuracy
Acceleration – Part Load [DR]	7.9	8.0	98.7%
Expected acceleration [DR]	9.6	9.1	94.8%
Pedal map [DR]	7.4	7.6	97.3%

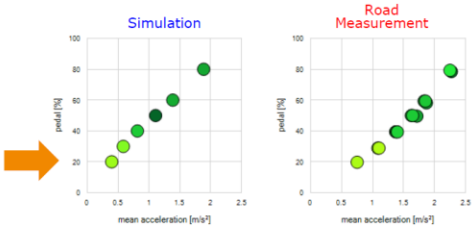


Figure 16: Powertrain - Correlation of measured data with simulation output part load, time-based data

The key takeaway from the simulation results is that the values obtained agree with more than 90%, the defined KPIs are satisfied.

Suspension

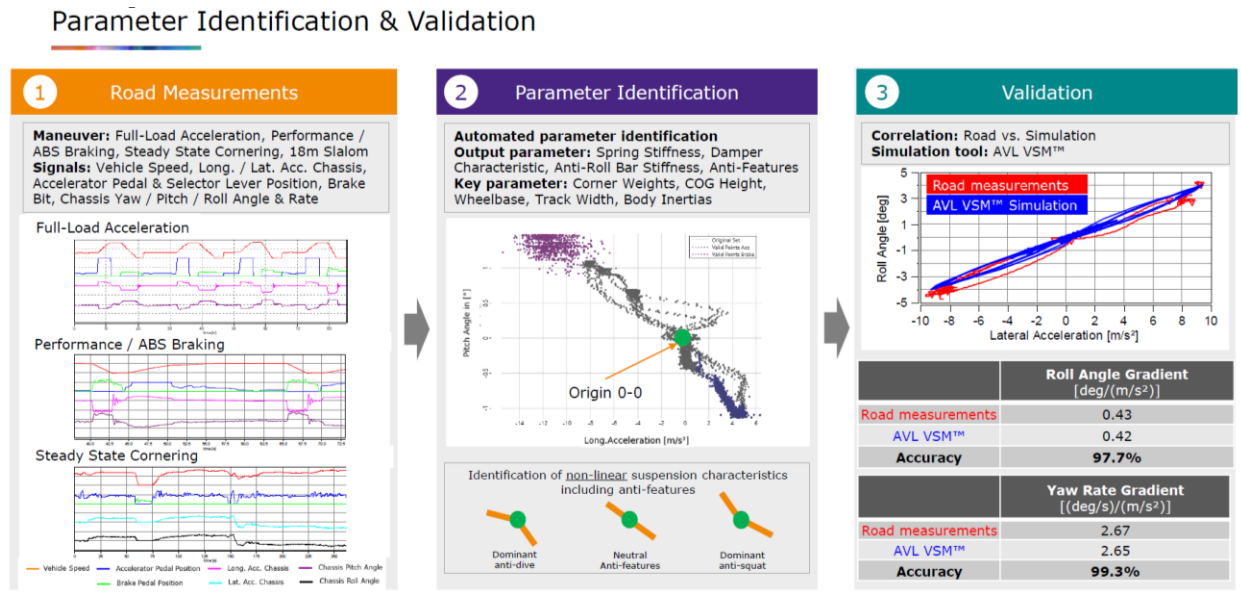


Figure 17: Example of suspension model validation flow

Validating the behaviour of a vehicle's suspension system through three primary stages:

Road Measurements: Real-world data collection from test manoeuvres such as full-load acceleration, performance/ABS braking, and steady-state cornering. Various signals, such as vehicle speed, longitudinal and lateral accelerations, chassis pitch/roll/yaw angles, and pedal positions, are measured to characterize the vehicle's dynamic response.

Parameter Identification: Automated tools are used to extract suspension parameters such as spring stiffness, damper characteristics, anti-roll bar stiffness, and anti-features like anti-dive and anti-squat. Key vehicle parameters (e.g., corner weights, centre of gravity height, wheelbase, track width, and body inertias) are also identified. This stage visualizes data, such as pitch angle versus longitudinal acceleration, to detect nonlinear suspension characteristics.

Validation: Correlation between road test data and simulation results using a simulation tool. The suspension model is validated by comparing simulation results with road measurements for metrics such as roll angle gradient and yaw rate gradient during lateral acceleration. The high accuracy percentages indicate a strong match between the real-world and simulated data.

This process is crucial for ensuring that the suspension model behaves accurately in both experimental and real-world conditions, enabling reliable simulation of vehicle dynamics.

Suspension – vehicle roll behaviour is shown on the figure below:

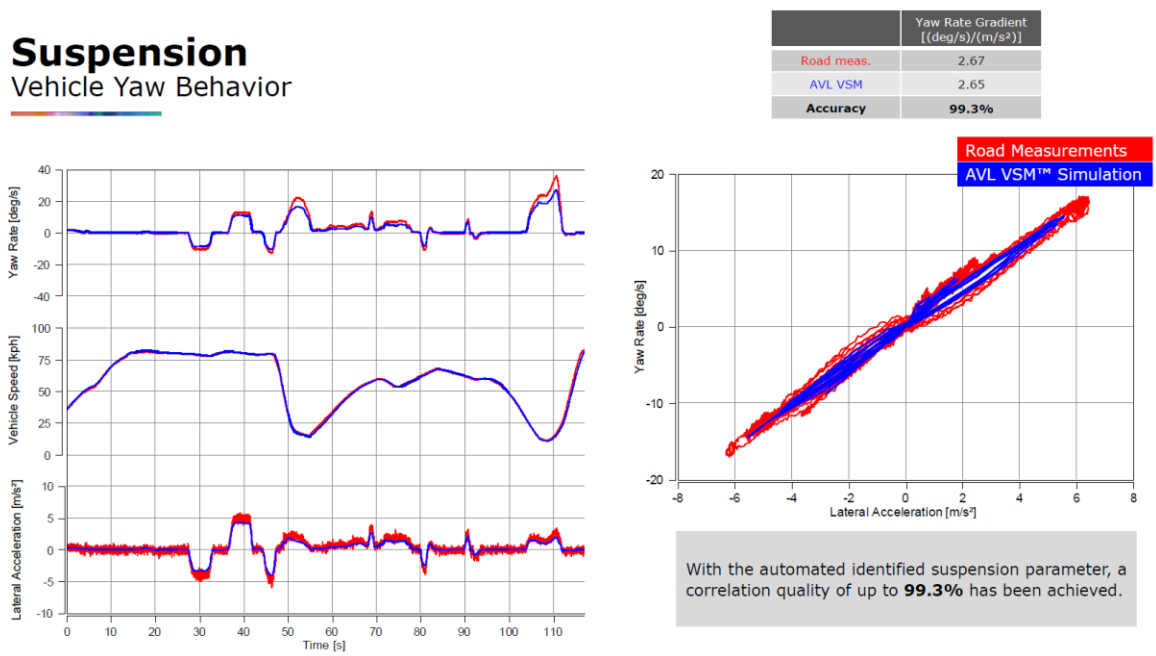
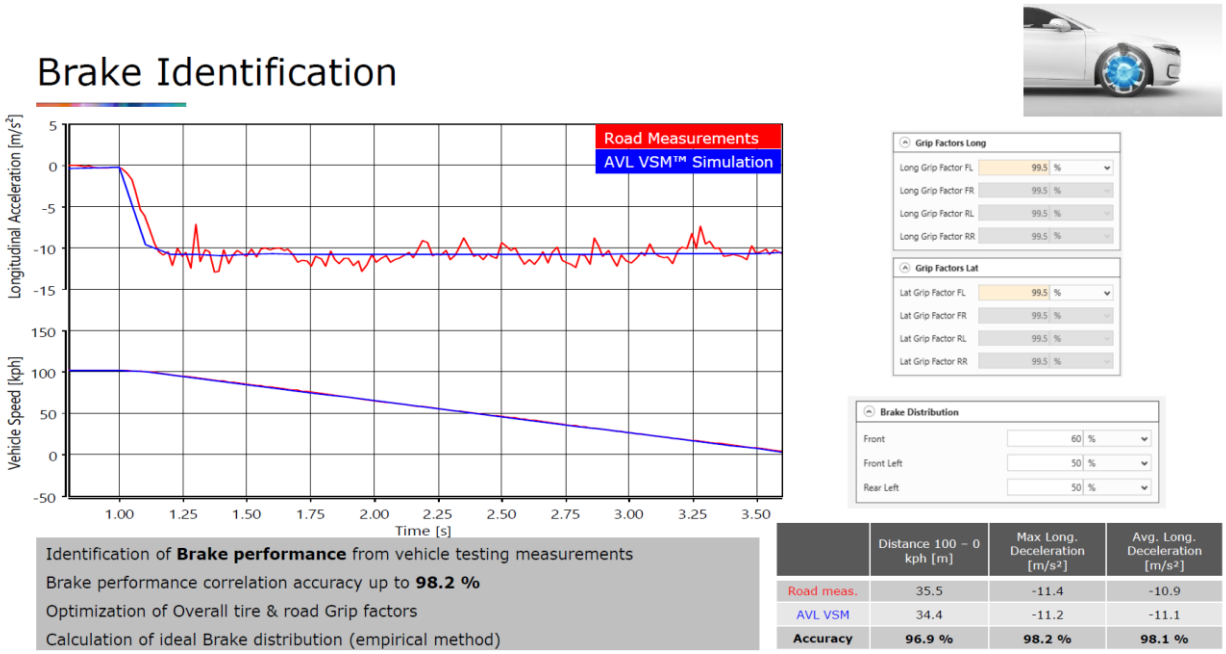


Figure 18: Suspension - Correlation of measured data with simulation output, time-based data

Brake



	Distance 100 – 0 kph [m]	Max Long. Deceleration [m/s²]	Avg. Long. Deceleration [m/s²]
Road meas.	35.5	-11.4	-10.9
AVL VSM	34.4	-11.2	-11.1
Accuracy	**96.9 %**	**98.2 %**	**98.1 %**

Figure 19: Brake - Correlation of measured data with simulation output, time-based data

This figure above represents the brake validation process for a vehicle, detailing how brake performance is assessed and correlated between real-world road measurements and simulation results. It highlights various aspects of brake system identification and validation, ensuring the braking system's accuracy in both experimental and simulated conditions.

The graph at the top-left corner illustrates the comparison between road measurements (red line) and simulation (blue line) for longitudinal acceleration and vehicle speed during a braking event. The close alignment of these curves indicates a high level of accuracy in the simulation's ability to replicate real-world brake performance.

3.2.2 Sub-model validation – sensor model

3.2.2.1 Camera sensor model

The focus of this section is on validation of camera models used in simulation environment such as Unreal Engine [17]. Here, we present a validation approach that is built upon two validation methodologies, namely, New Assessment/Test Method (NATM) [18] by United Nations Economic Commission for Europe (UNECE) and the comprehensive approach for validation of virtual testing toolchains [19] by SAE International. The high-level overview of this approach is presented in Figure 20. In Annex 1, we present sample results obtained from applying the camera validation approach presented in this section.

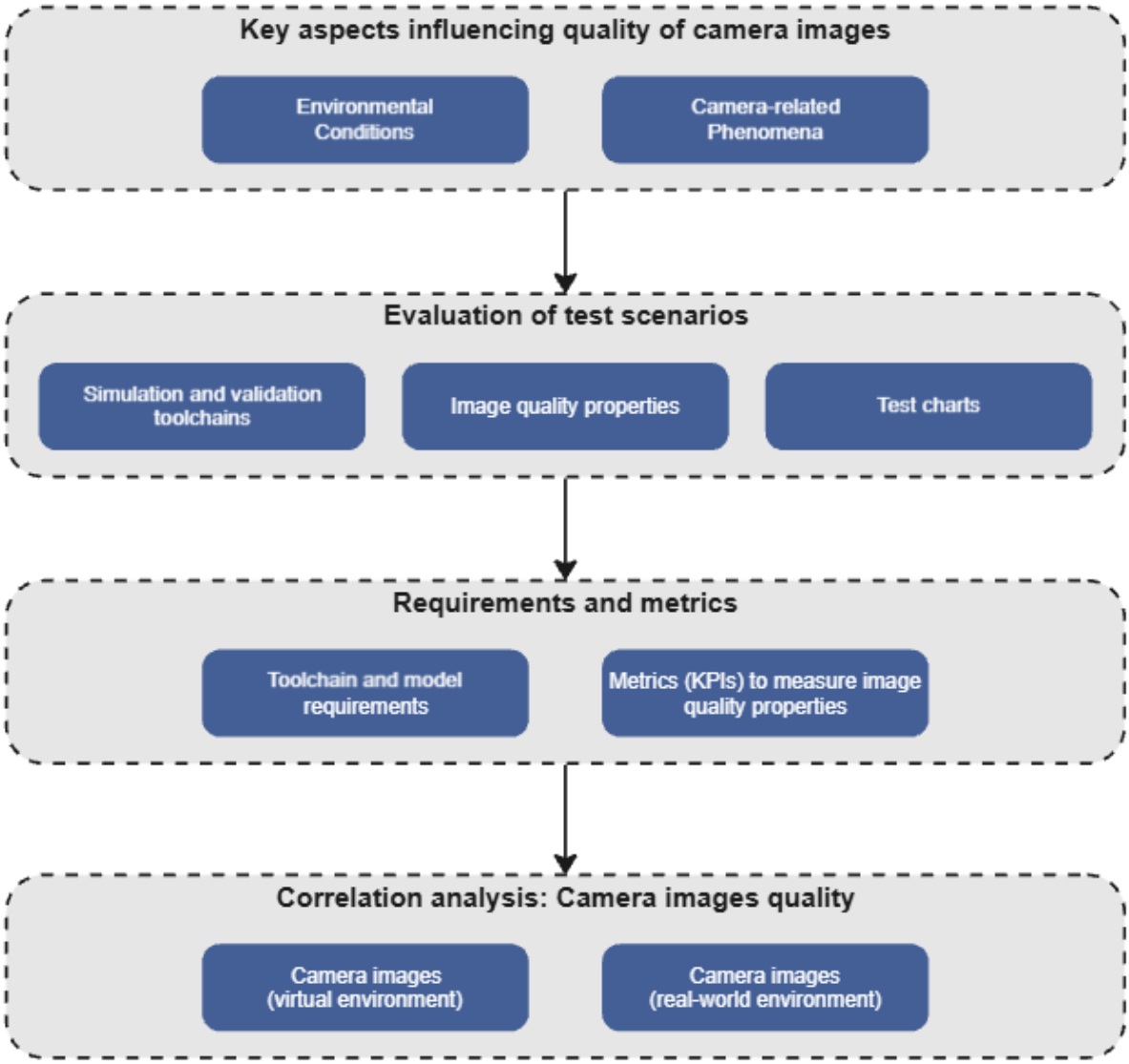


Figure 20. Overview of camera model validation approach, inspired by NATM [18] and SAE [19] validation methodologies.

The SAE methodology provides a structured approach to systematically preparing the simulation toolchain validation, which involves:

I. analysing operational design domain

II. identifying test scenarios

III. defining model and toolchain requirements

When it comes to analysing the ODD, perception systems may operate within a specific ODD. ***Environmental factors*** such as lighting conditions and weather aspects are essential to be considered as the varying light intensity can significantly influence the camera's ability to capture clear images and distinguish between objects in shaded or brightly lit areas. Apart from environmental conditions, there are ***camera-related phenomena*** that could influence the accuracy of images. Some of these phenomena as also listed by Ciuffo et al. [20] are: ***lens distortion***, ***vignette***, ***grain jitter***, ***bloom***, ***auto exposure***, ***lens flares***, ***depth of field***, **and** ***exposure time***. Note that here in this section, we focus only on validation of sub-models (in this case the camera model). This means that more than investigating variations in environmental conditions, we should focus on the phenomena that influence the quality of camera images. The influence of environmental conditions on the quality of camera images is something that should be studied as part of validation methodology investigating the interaction between the environment and sensor models.

After analysing the ODD and identifying key aspects that could influence the quality of camera images, ***test scenarios*** should be identified in a way to cover a wide range of operating conditions for the perception module. These scenarios should be designed to evaluate the camera model under environmental challenges as well as camera-related phenomena. The ***simulation and validation toolchains*** should also be clearly identified and their capabilities analysed. Moreover, a complete list of ***image quality properties*** should be identified. These are the properties that basically define what aspects should be the focus of the validations. Examples of these properties are *colour accuracy*, *sharpness*, *resolution*, *distortion*, and *chromatic aberration*.

To facilitate the assessment of the image quality properties, standard ***test charts*** and ***tools*** could be used within both virtual and real-world setup. A non-exhaustive set of such tools are presented by Ciuffo et al. [20]:

- *Macbeth Colour chart Test*: determine camera colour space, noise figures, and exposure characteristics.
- *Opto-Electronic Conversion Function (OECF) chart Tests*: evaluate the relationship between input luminance and output digital level.
- *Special Frequency Response (SFR) chart*: measure sharpness, contrast, and lens effects.
- *Lens Flare characterization*: determine lens sensitivity to flare and ghosting.
- *F-theta calibration*: checkerboard test to determine F-theta polynomial.

Based on the results obtained so far, concrete ***requirements*** should be defined on the **simulation models** and **toolchains** in addition to ***metrics***, also known as ***key performance indicators (KPIs)*** to measure the quality of camera images. These metrics are tightly connected to the image quality property under focus. For example, *could* be used to quantify the perceptual difference between two colours; moreover, *MTF50* could be used to measure how well a camera can reproduce varying levels of detail or contrast at different spatial frequencies.

After performing the above steps, the validation process can be completed by comparing the outputs of the virtual camera model with real-world data using the metrics defined to measure the image quality properties. Additionally, the ability of the virtual model to handle shadow rendering, and occlusions from natural obstacles can be tested against real-world results to ensure realistic sensor behaviour in the simulated environment.

3.2.2.2 Radar sensor model

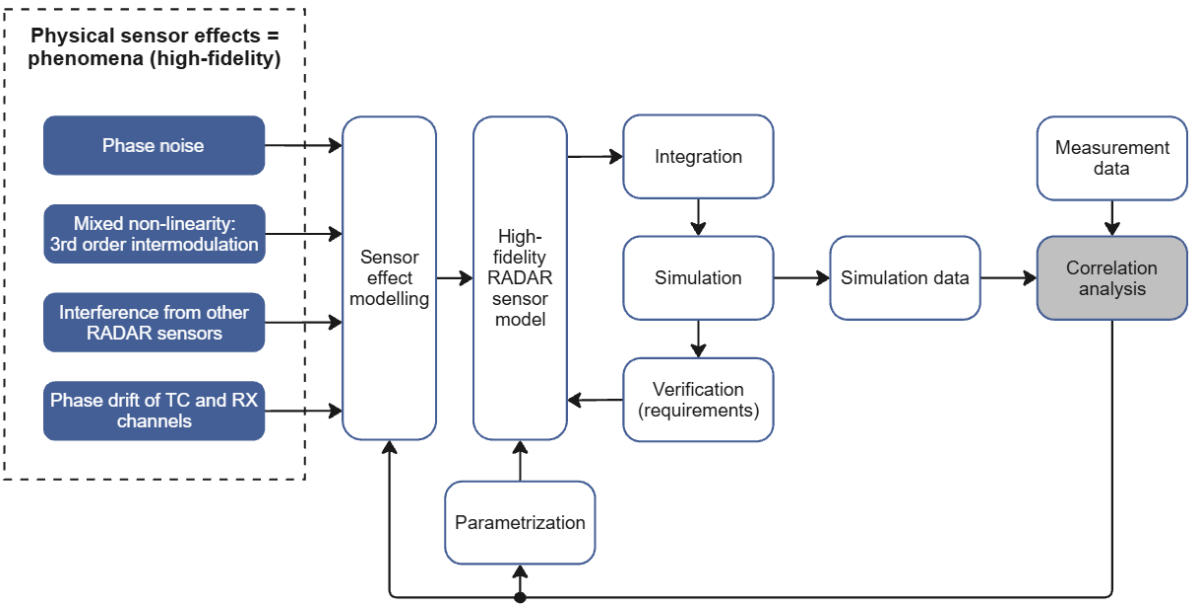


Figure 21: RADAR sensor sub-model validation flow – schematic overview

Figure 21 shows the overall principle for the model quality validation of the RADAR sensor model. Based on the chosen physical sensor effects, which are based on certain dedicated real-world phenomena (e.g., phase noise, mixer non-linearity), the respective sensor effects are modelled and integrated into the high-fidelity RADAR sensor model. Usually, in general, and for each modelled effect, there are various parameters to adjust the behaviour of the model. After the integration into the simulation toolchain, one can execute a simulation run. This is followed by a correlation analysis, where the generated simulation data is compared to the available measurement data. Based on the outcome of the correlation analysis, the parameters of the model mentioned can be adapted accordingly.

Hence, the conceptual validation, based on [10], consists of checking the consistency of the modelling hypothesis, including the chosen level of detail (model fidelity), in relation to the target objective, which is driven by the target OD and BC of the ADS-equipped vehicle.

Response analysis can be utilized to determine the degree of discrepancy between the simulated model and the physical realization [10]. Successful model validation is accomplished if the discrepancy is below a prescribed threshold level. Such a phase is typically accomplished by defining a selection of KPIs and the appropriate computational method to compare the recorded signals.

3.2.2.3 LiDAR sensor model

LiDAR (stands for Light Detection and Ranging) is known since long as geometrical measurement technique in land surveying and mapping. The principal of LiDAR is simple in theory: the system sends out light photons. When they hit objects in the field, the photons are scattered and some of them return to the source. Based on the elapsed time between emission and collections of photons and the constant speed of light, the distance between the LiDAR and the objects can be easily calculated.

Figure 22: Comparison of LiDAR specifications with respect to camera and radar [21]

LiDAR offers significant advantages over other sensing technologies in terms of range, accuracy, and reliability. It provides a consistent level of confidence across its sensing range, ensuring high reliability in detecting objects at various distances. Unlike cameras, which depend on ambient lighting and can struggle in darkness or glare, LiDAR operates effectively in any lighting condition by emitting its own laser light. This allows it to perform well in both daytime and nighttime scenarios. LiDAR is also capable of identifying a wide variety of objects, including vehicles, pedestrians, bicycles, stationary elements, uniquely shaped objects, and objects in dark environments. Its ability to create high-resolution, three-dimensional point clouds makes it particularly useful for capturing detailed information about the size, shape,

and position of objects. While radar is effective for detecting basic object shapes and movement, it lacks the high-resolution capabilities of LiDAR, making it less effective for distinguishing complex or uniquely shaped objects. LiDAR's independence from environmental lighting conditions and its precise object detection capabilities make it indispensable for applications like autonomous driving and advanced driver-assistance systems.

To validate a LiDAR sensor model involves comparing the model's outputs to theoretical calculations, analytical predictions, and manufacturer specifications in a controlled and idealized virtual environment. This ensures that the sensor's behaviour aligns with its intrinsic characteristics, independent of external influences.

The process starts with **defining the key parameters** of the LiDAR model that need validation, such as **range**, **resolution**, **field of view**, and noise characteristics. Each of these parameters must align with the specifications provided by the LiDAR manufacturer. For example, the range should be tested by simulating objects at various distances within the sensor's operational limit and verifying that the measured distances in the simulation correspond to the expected values based on time-of-flight calculations.

Geometric accuracy is another crucial aspect of validation. In this step, the LiDAR model is tested by scanning simple, well-defined virtual shapes such as planes, spheres, or cubes. The resulting point clouds are analysed to ensure that they accurately reconstruct the shapes and dimensions of the objects. Any discrepancies in distances, angles, or point density are indicators that the model may need refinement.

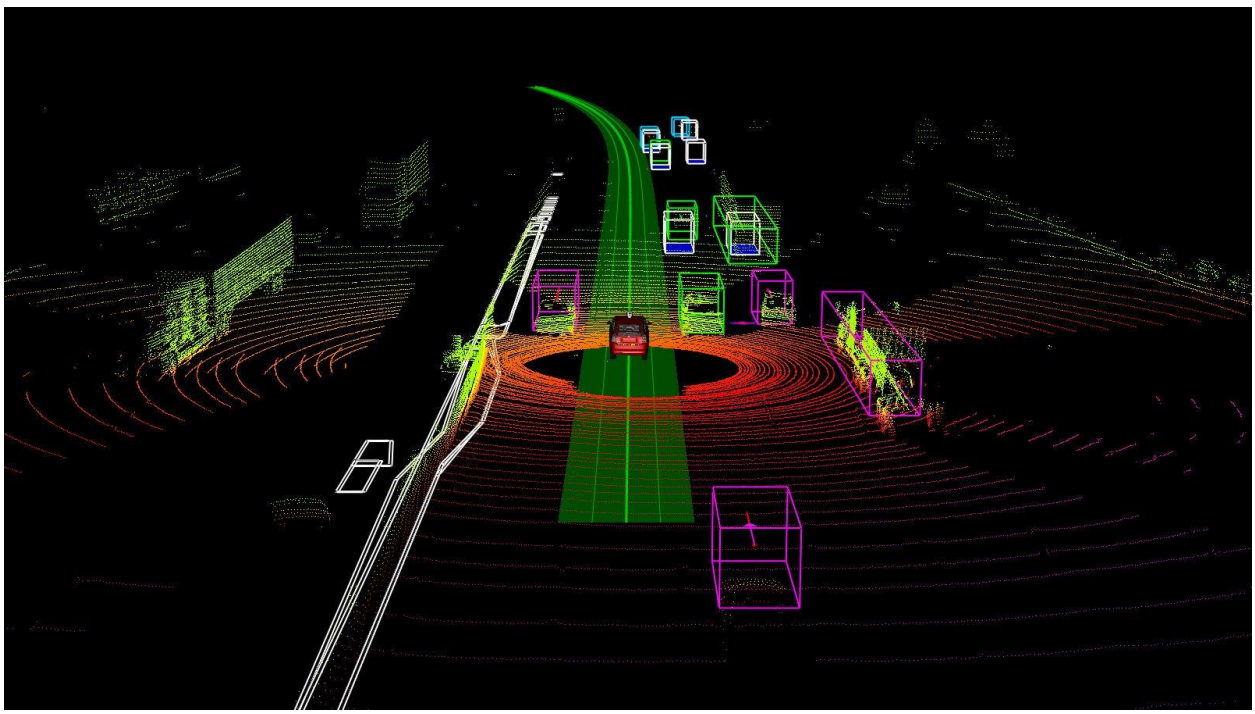


Figure 23: Visual representation of point cloud data produced by a LiDAR sensor [22]

The scanning behaviour of the LiDAR is validated by examining the field of view, angular resolution, and scan pattern. Virtual objects are placed across the sensor's horizontal and

vertical detection ranges to confirm that the model detects them consistently within the specified field of view. The angular resolution of the model is validated by measuring the spacing of points in the generated point cloud and ensuring it matches the real sensor's resolution. The scan pattern, whether rotational or solid-state, is also checked to confirm that the timing and sequencing of emitted laser beams are accurate.

Noise and artifacts are an important part of LiDAR modelling, as real-world sensors inherently produce noise in their data. The model is tested to ensure it replicates noise characteristics such as range noise, timing jitter, and angular error, as specified by the manufacturer. Virtual tests can also introduce controlled synthetic noise to confirm that the model handles it realistically, including producing artifacts like point dropouts or distortions when expected.

To validate intensity returns, the model is tested with virtual objects of varying reflectivity. The returned intensity values should correspond to the reflectivity of the objects, as well as their distance and angle relative to the sensor. This step ensures that the model accurately simulates how reflectivity impacts the sensor's output.

Finally, the overall performance of the LiDAR model is cross validated against analytical models and experimental data. Time-of-flight calculations, for example, can be compared to simulated results to confirm that the model correctly calculates distances. If experimental data from a physical LiDAR sensor is available, it can be used to validate the model further by replicating the same conditions in the simulation and comparing the outputs. This iterative process of testing and refinement ensures that the LiDAR model reliably represents the real sensor's behaviour under ideal and controlled conditions.

Since LiDAR outputs point cloud data, refer to the recommended correlation analysis approaches in section 3.1 for more information on correlation analysis used for validation.

3.2.2.4 Ultrasonic sensor model

An ultrasonic sensor is a device that uses high-frequency sound waves, typically above the range of human hearing (usually around 20 kHz to several MHz), to measure the distance to an object or detect its presence. The sensor works by emitting ultrasonic sound waves and then measuring the time it takes for the waves to travel to the target and return to the sensor after bouncing off the object.

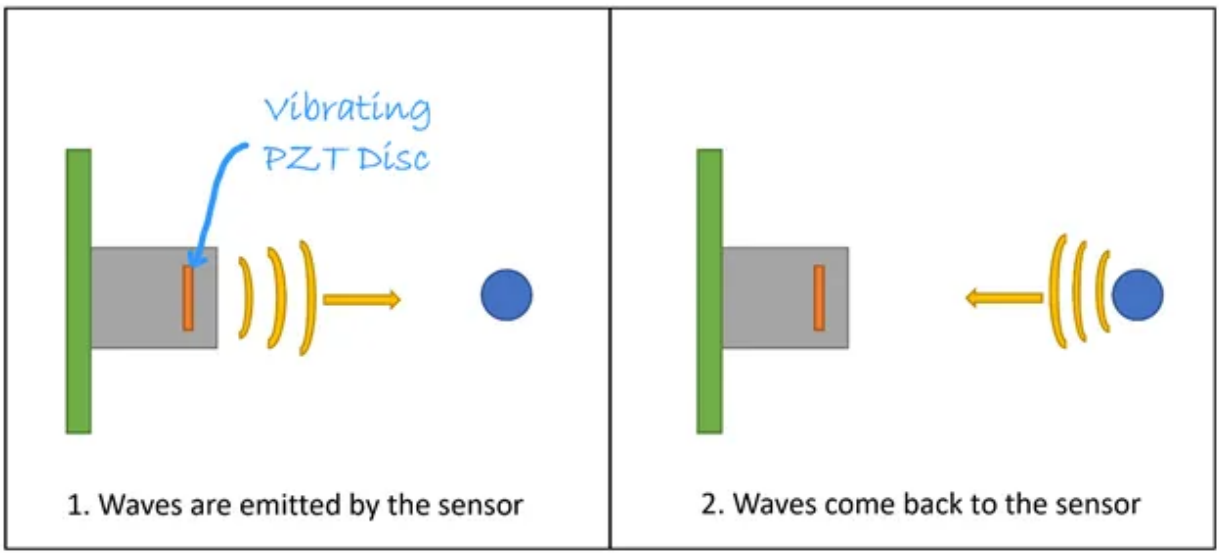


Figure 24: Visual representation of the ultrasonic principles of operation [23]

To validate an ultrasonic sensor model, various aspects of the sensor's behaviour must be compared to real-world observations. One of the main factors to consider is the **sensor's range** - the distance over which it can effectively measure. The model must be able to predict the sensor's performance accurately across both the minimum and maximum range limits. This involves assessing how the sensor's measurements may degrade at longer distances or in environments where conditions like temperature, humidity, and air pressure fluctuate. A properly validated model will reflect how these environmental changes impact the sensor's readings.

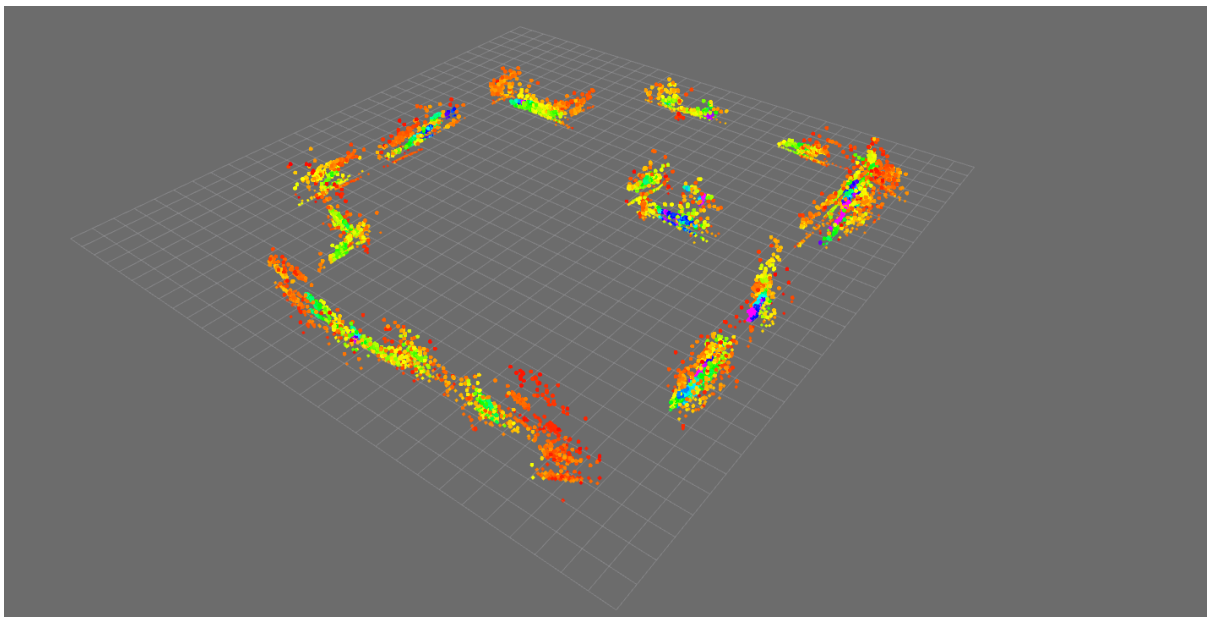


Figure 25: Example of point cloud data generated by a set of ultrasonic sensors [24]

Another critical element in validation is the beam **pattern of the sensor**, which describes how the ultrasonic waves propagate in different directions. The model should predict the shape and spread of the ultrasonic beam to ensure that the sensor's detection capabilities are accurately captured. The sensor's ability to focus its beam and detect objects at specific angles can significantly impact its performance in complex environments, so the model must account for the spread and intensity of the beam.

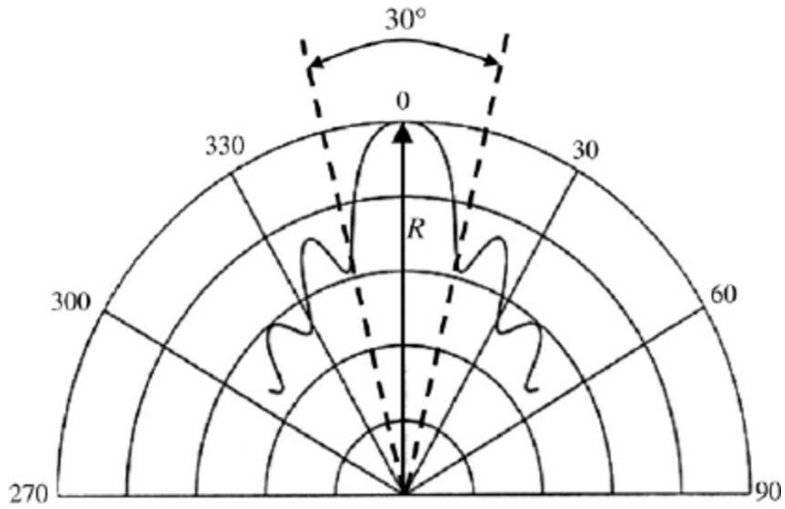


Figure 26: Typical beam pattern of the Polaroid ultrasonic sensor at 50 kHz [25]

Additionally, signal propagation and reflection characteristics need to be accurately modelled. Ultrasonic waves interact with objects differently based on their size, shape, and material composition. The model must incorporate how these factors affect wave reflection, absorption, and scattering, as these factors influence the sensor's ability to detect objects and measure distances correctly. For instance, soft materials may absorb sound waves more than hard, reflective surfaces, which would require adjustments to the model's predictions in real-world testing.

Model validation is often carried out through comparison experiments. These involve placing the ultrasonic sensor in a controlled environment where known distances and objects are used. The model's predicted values are compared against the actual sensor measurements to assess its accuracy. Any discrepancies between the model and actual data can reveal areas where the model needs refinement, whether in accounting for environmental effects or adjusting for sensor-specific characteristics.

Similar to LiDAR's model, ultrasonic sensor outputs point cloud data, refer to the recommended correlation analysis approaches in section 3.1 for more information on correlation analysis used for validation.

3.2.3 Sub-model validation – environment model

Environment model validation ensures that a perception system accurately represents the real-world environment. This process evaluates how well the model captures both static and dynamic aspects of the surroundings.

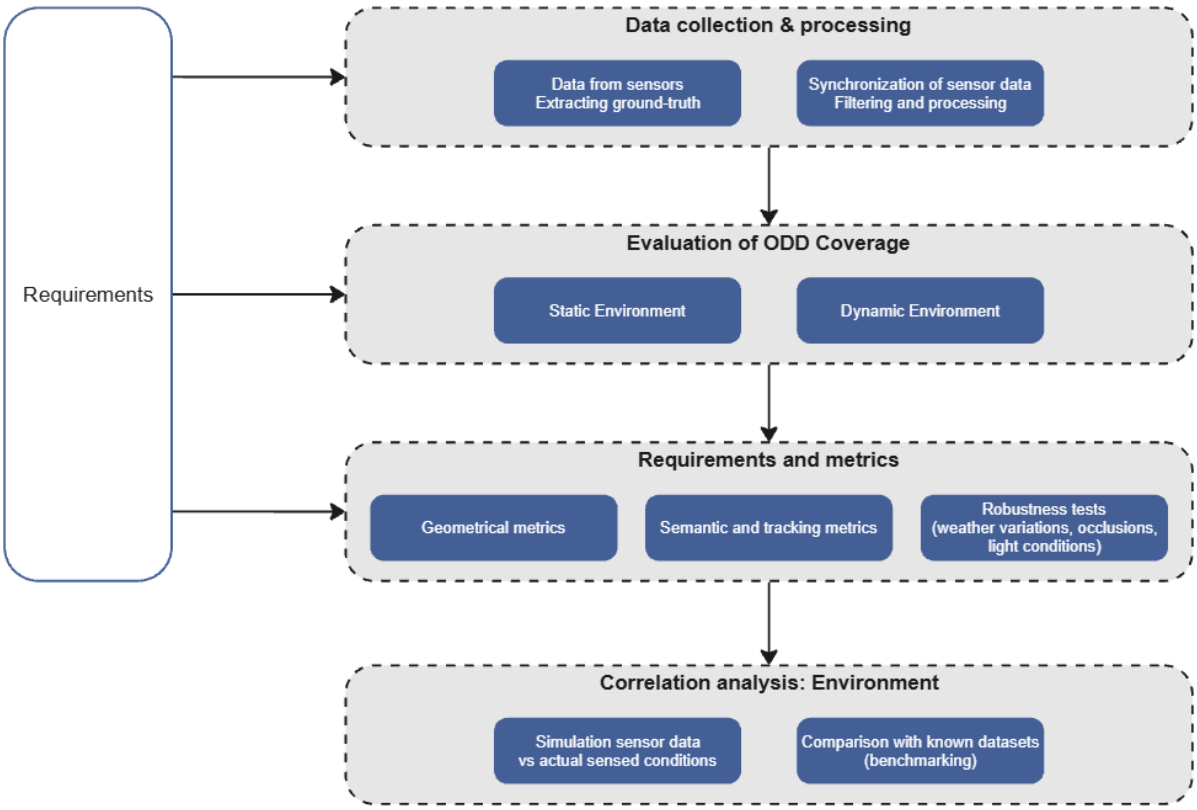


Figure 27: Validation flow of the environment sub-system model

3.2.3.1 Static Environment

The static environment consists of elements that remain unchanged over time, such as road geometry, lane markings, intersections, and surface conditions like asphalt, gravel, or ice. Roadside objects, including signs, guardrails, trees, and other infrastructure elements like tunnels and bridges, also fall into this category. Additionally, static environmental factors encompass weather conditions, such as fog, rain, or snow, which influence visibility and road friction.

One of the primary methods for validation is geometric accuracy assessment, which involves comparing detected road features with high-precision maps, LiDAR scans, or survey data. Position accuracy is measured by calculating the deviation between detected lane boundaries, intersections, or road edges and their actual locations. Lane width, curvature, and slope detection must also be validated to ensure the system correctly interprets road structure. Intersection-over-Union (IoU) calculations help evaluate how well the detected objects, such as road signs or barriers, align with ground truth data.

Another key aspect is surface condition recognition, where the perception system must correctly identify different types of road materials, wet or icy patches, and surface irregularities. Ground truth data from sensor fusion (LiDAR, radar, and cameras) or real-world measurements can be used to verify the system's ability to differentiate between dry, wet, and slippery conditions.

For roadside objects and infrastructure, the system must detect and classify static objects such as signs, poles, guardrails, and buildings. Validation involves checking the detection accuracy, classification correctness, and position consistency of these objects over time. False positives (detecting non-existent objects) and false negatives (missing actual objects) must be minimized. Temporal consistency ensures that objects detected in consecutive frames remain stable and do not shift unexpectedly due to perception errors.

Environmental condition robustness testing assesses how well static elements are perceived under different lighting and weather conditions. This includes testing lane detection accuracy in low-light environments, fog, and rain, as well as evaluating how well signs and road markings remain visible under varying conditions. Sensor redundancy, such as combining camera-based and LiDAR-based detection, can improve robustness.

Validation is typically performed using recorded datasets, real-world driving tests, and simulation environments. Benchmark datasets, such as HD maps or high-resolution aerial imagery, serve as ground truth references. Simulation-based validation allows controlled testing under diverse conditions without real-world risks, helping to evaluate how perception models handle variations in road infrastructure and environmental factors.

3.2.3.2 Dynamic Environment

The dynamic environment involves constantly changing elements, such as other road users, including vehicles, pedestrians, and cyclists. Traffic flow characteristics, like speed variations, lane changes, and density fluctuations, play a significant role in determining how well a model adapts to real-world conditions. Other dynamic factors include temporary road changes, such as construction zones or lane closures, and transitions in environmental conditions, such as the sudden onset of rain or fog.

One of the key aspects is object detection and classification accuracy, where the system must correctly identify vehicles, pedestrians, cyclists, and other road users. Performance is evaluated using ground truth data from high-precision sensors, annotated datasets, or reference systems like LiDAR-based tracking. Metrics such as **precision, recall, and Intersection-over-Union (IoU)** help quantify how well detected objects match real-world positions. False positives (detecting non-existent objects) and false negatives (failing to detect real objects) are critical factors in validation.

Tracking consistency is another important validation criterion. The perception system must maintain a continuous and accurate trajectory for each detected object over time, ensuring that moving vehicles or pedestrians are not lost or misidentified across frames. Metrics such as identity switching (when an object is mistakenly assigned a new ID) and tracking drift (gradual loss of accuracy over time) help measure tracking reliability.

Motion prediction accuracy evaluates how well the system anticipates the future positions and behaviours of dynamic objects. This is particularly important for collision avoidance and path planning. Prediction accuracy is validated by comparing the system's estimated trajectories with ground truth data from real-world movements. Errors in predicted vs. actual paths can be quantified using displacement error metrics, such as Average Displacement Error (ADE) or Final Displacement Error (FDE).

To ensure robustness in real-world conditions, environmental and scenario-based testing is conducted under various traffic densities, road types, and weather conditions. For example, the system must be validated in low-traffic suburban environments as well as dense urban scenarios with unpredictable pedestrian movement. Nighttime, rain, and fog tests assess how well perception and tracking systems function under adverse conditions.

Validation can be performed using real-world testing, sensor fusion-based ground truth, and simulation environments. Real-world driving tests use HD maps and sensor data to compare detections with actual traffic conditions. Sensor fusion, combining LiDAR, radar, and camera data, helps generate more accurate ground truth references. Simulation-based validation allows the system to be tested in rare or dangerous scenarios, such as emergency braking situations or sudden pedestrian crossings, without real-world risks.

Since dynamic environments are highly variable, scenario-based validation aligned with the Operational Design Domain (ODD) is critical. The system must be tested under conditions representative of its intended operating environment, such as highways, city traffic, or mixed-use roads. By validating detection, tracking, and prediction accuracy across different scenarios, the system can be refined to ensure safe and reliable operation in real-world traffic conditions.

3.2.3.3 ODD role in the validation of environment model

The ODD plays a crucial role in validation by setting the boundaries within which an autonomous system is expected to operate. It defines specific conditions such as road types, traffic densities, environmental factors, and speed limits. By aligning testing efforts with the defined ODD, validation ensures that the system is evaluated in relevant scenarios, making sure it behaves as expected within its intended environment.

A well-defined ODD also establishes clear pass and fail criteria, helping to determine whether the environment model meets the necessary accuracy and reliability requirements. If the system consistently performs within the defined parameters, it can be considered safe for deployment. Conversely, if it fails under conditions that fall within its ODD, further refinement is necessary before real-world implementation.

Ensuring safety is another key reason why ODD is critical. Autonomous and assisted driving systems must function reliably under the specified conditions, whether that includes urban streets, highways, or mixed-use roads. By validating the system within these constraints, developers can verify its robustness and identify potential failure points before deployment, reducing the risk of unexpected behaviour in real-world scenarios.

Regulatory compliance and standardization often require validation to be conducted in accordance with an ODD. Authorities and industry standards, such as ISO 21448 (Safety of the Intended Functionality - SOTIF), may mandate that systems are tested and proven to work safely within their declared operational boundaries. This structured validation approach ensures that the system is not only functionally effective but also legally compliant and aligned with industry safety expectations (more details in [26]).

METRICS

1. **Geometrical Metrics**
    - **Position accuracy**: Difference between perceived and actual object positions (e.g., Euclidean distance).
    - **Lane detection accuracy**: How well the model detects lane boundaries and their topology.
    - **Bounding box overlap (IoU - Intersection over Union)**: Measures how well the detected object overlaps with its ground truth.
    - **Curvature and slope accuracy**: Validates road geometry representation.
2. **Semantic and Temporal Metrics**
    - **Object classification accuracy**: Measures correct identification of vehicles, pedestrians, signs, etc.
    - **Tracking consistency**: Evaluates if dynamic objects are consistently tracked across frames.
    - **False positive/negative rates**: Measures misdetections of obstacles or lane markers.
    - **Short-term and long-term prediction accuracy**: Assesses how well the model predicts object motion.
3. **Environmental Condition Robustness**
    - **Performance under adverse weather conditions**: Fog, snow, rain, and low-light situations.
    - **Impact of occlusions**: Evaluates how well the system handles partially visible objects.
    - **Variability in road conditions**: Wet, dry, icy, or rough surfaces.

3.3 Methodology for Validation Setup for B.1

The validation of the integrated vehicle-environment co-simulation ensures that the vehicle model’s response to driver (or ADF) inputs matches the real-world behaviour for various

environmental conditions of interest (e.g., road geometry and surface, weather) with sufficient accuracy. What exactly “sufficient accuracy” means strongly depends on the automated mobility system’s ODD and more specifically the test scenarios and cases which are executed in simulation for its safety assessment. Although there are numerous types of vehicle models with different levels of detail, there will always be a discrepancy between measurements and simulation results. Thus, the vehicle-environment model interaction validation is only reasonable for the domain resulting from the test scenarios. The remaining part of this section describes a general process for the vehicle model validation based on the framework from [27], which is independent of ADF, vehicle model, and test case.

Table 7: Validation setup “B” sub-model methodology split

Validation Setup B:	Description:
B.1	Validation of the **integrated vehicle-environment** co-simulation **Method**: Open loop driving manoeuvre simulation to correlate vehicle dynamics **Criteria Examples**: Torque map, steering ratio, tire dimension, tire friction etc.
B.2	**Validation of integrated sensor-environment co-simulation** **Method**: Open-loop sensor simulation to correlate sensor data **Criteria Examples**: (e.g. Reflections, camera image, …)

Figure 28 illustrates the steps and shows the relationship between validation of sub-models (A.1) and vehicle-environment co-simulation (B.1).

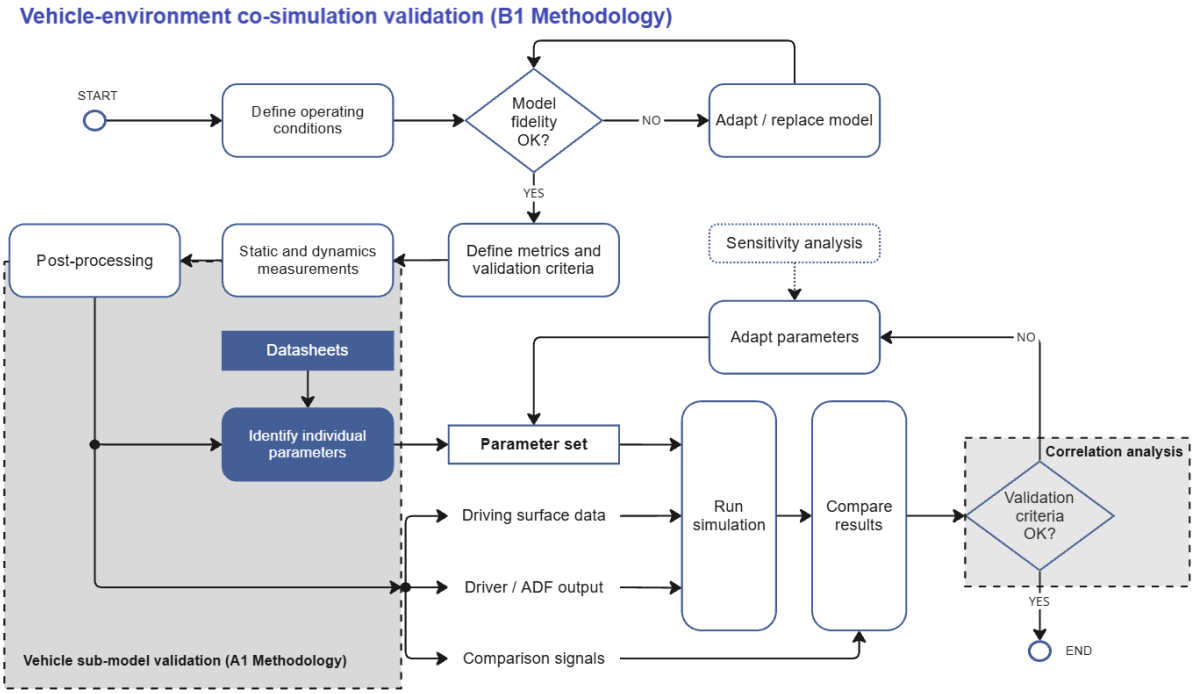


Figure 28: Generic vehicle model validation process based on the framework from [1].

The process starts with defining the operating conditions where the vehicle model should be applied. After that, it is verified that the model fidelity is sufficient for those conditions. Both steps influence the metrics and criteria which are utilized for the model validation. Next, static and dynamic measurements are designed for both parameter identification and (sub)model validation. Since some of these experiments are useful on sub-model as well as on the complete vehicle model level, the respective block is at the border of validation setup A1 and validation setup B1 (together with measurement signal post-processing). Based on these measurements and available datasheets the parameter identification is conducted and results in the vehicle model's initial parameter set. The actual model validation is done by comparing the previously selected metrics of signals from simulation with those from reality and verifying that the predefined criteria are met. Thereby, it is important to use the recorded signals from the driver or ADF (i.e., throttle, brake, and steering wheel angle) as model inputs. Apart from the road slope, the main interface between vehicle dynamics and the environment model is the tire-road contact. This includes possible weather influence through precipitation by varying the road friction coefficient (part of driving surface data). If the validation criteria check fails, parameters can be adapted iteratively until a better correlation with measurements is achieved. These adaptations are either done manually by an expert or automatically with an optimization framework. As high-fidelity vehicle models (e.g. multi-body models) contain many parameters, a sensitivity analysis on their influence on the validation criteria should be conducted beforehand to reduce the number of optimization variables and computation time. The following subsections provide further explanations for some of the mentioned tasks.

3.3.1 Definition of operating conditions

The operating conditions mainly depend on the test cases' parameter ranges and the expected behaviour of the SUT. They include longitudinal and lateral speeds, accelerations, and jerks, road properties such as curvature, slope, and friction coefficient as well as the vehicle load [27].

3.3.2 Model fidelity check and model selection

Typically, an ADF consists of several components (sense-plan-act paradigm) which can be tested independently as well as jointly and this influences the necessary level of detail for simulations. Separate testing of decision-making and behaviour planning routines is often done with simple vehicle models without any detailed sub-system models, whereas testing of vehicle dynamics controllers demands higher fidelity. In general, it must be ensured that the vehicle model represents the real-world behaviour accurately enough for a given set of operating conditions and assumptions. Challenging operating conditions require more detailed sub-models of certain vehicle components. For example, in some emergency braking test cases a nonlinear tire model based on the longitudinal slip may be sufficient, but for split-mu braking or evasive manoeuvres involving steering a combined (lateral and longitudinal) tire slip model is necessary. Another important aspect is the model's interface (input and output quantities) which not only must fit the rest of the simulation toolchain but also provide all necessary information to compute the KPIs specified in the test case. If needed, the model must be extended accordingly or another model fulfilling the fidelity and interface requirements must be selected.

3.3.3 Measurements and driving manoeuvres

The validation of a vehicle dynamics model is done by comparing simulation results with measurements at the same operating conditions. If the model contains sub-models the parameters should be identified from dedicated measurements or datasheets wherever possible (see sub-model validation A1 section 3.2.1). The measurements described in this section are categorized into static (performed at standstill / in workshop), longitudinal dynamics, and lateral dynamics. The following manoeuvre list serves as a basis for model validation and can be extended or reduced depending on the operating conditions.

3.3.3.1 Static measurements

Table 8: Static measurements

ID	Name	Short description
0.1	Tire loads, horizontal CG position	Normal force on four wheels, vehicle on plane surface
0.2	Vertical CG position	Lift car at one axle with angle and measure lifting force
0.3	Static steering characteristics	Measure angles at front wheels and steering wheel

0.4	Axle kinematics	Characteristic curves of springs and anti-roll bar, camber and toe angles

Static measurements are rather for model parametrization than for the validation process. Depending on the model fidelity some of them may be skipped.

3.3.3.2 Longitudinal dynamics

Table 9: Longitudinal dynamics manoeuvres

ID	Name	Short description
1.1	Straight line acceleration	Accelerate from standstill to with constant throttle, high µ, low µ
1.2	Straight line braking	Decelerate from to standstill with constant brake pedal, high µ, low µ, split µ
1.3	Straight line roll-out	Switch to neutral gear at and measure until car comes to standstill, high µ
1.4	Road bumps and road holes	Measuring spring deflection during run over with different velocities

Manoeuvres 1.1 and 1.2 are performed for several fixed pedal positions at least three times each. One variation of 1.2 can examine engine drag torque. The pedal positions must be set either by the onboard drive-by-wire system or an external pedal robot to get reproducible results. Measurements shall take place on a surface with a known friction coefficient µ. Cool-down phases for tires must be considered.

3.3.3.3 Lateral dynamics

Table 10: Lateral dynamics manoeuvres

ID	Name	Short description
2.1	Steady state cornering ISO 4138	Driving on a circle with fixed radius, increasing speed until dynamic limit
2.2	Sinus steering ISO 7401	Sinusoidal steering wheel angle profile with fixed frequency, amplitude: (stationary)
2.3	Step steering ISO 7401	Step-like steering wheel angle change, steering wheel rate at least, amplitude: (stationary)

2.4	Double lane change ISO 3888-1/2	Constant velocity, lane change to the left, then lane change to the right
2.5	Braking in a turn ISO 7975	Driving on a circle with fixed radius, constant speed, braking while steering wheel remains fixed

The suggested experiments involve steady-state lateral dynamics, as well as transient behaviour and real-world driving situations. All measurements 2.X should be performed at least three times on a surface with high µ. Manoeuvres 2.2 and 2.3 must be executed by the onboard steer-by-wire system or an external steering robot. More details can be found in the referenced ISO norms.

3.3.3.4 Measured signals

To compare the simulations with the real-world driving manoeuvres 1.X and 2.X the following signals should be measured (assuming a high-fidelity vehicle model):

Table 11: Measured signals

ID	Name	Comment
S0	Time	Common time base for all other signals
S1	Brake pedal	
S2	Throttle	
S3	Current gear	
S4	Steering wheel / actuator angle	
S5	Roll angle	
S6	Roll rate	
S7	Pitch angle	
S8	Pitch rate	
S9	Yaw rate	
S10	Longitudinal velocity	Not derived from wheel speed sensors (slip calculation)
S11	Lateral velocity or side slip angle	
S12 - S15	Wheel speeds front left (FL), front right (FR), rear right (RR), and rear left (RL)	
S16	Longitudinal acceleration	
S17	Lateral acceleration	

S18	Vertical acceleration	
S19	Engine speed	
S20 - S23	Brake pressure FL, FR, RR, and RL	Optional: for detailed brake subsystem models
S24 - S27	Wheel position and orientations relative to chassis	Optional: for detailed suspension subsystem models

For analysis and metrics computation all signals must use the same time base (at least after the post-processing step).

3.4 Methodology for Validation Setup for B.2

The model interface and fidelity check represent a prerequisite to validating the model interaction (validation setup B1 - vehicle & environment and validation setup B2 – sensor & environment) within the V&V process, as displayed in Figure 29. Hence, the next step is to specify the model interaction validation method. In particular, this section tackles environment–sensor interaction validation.

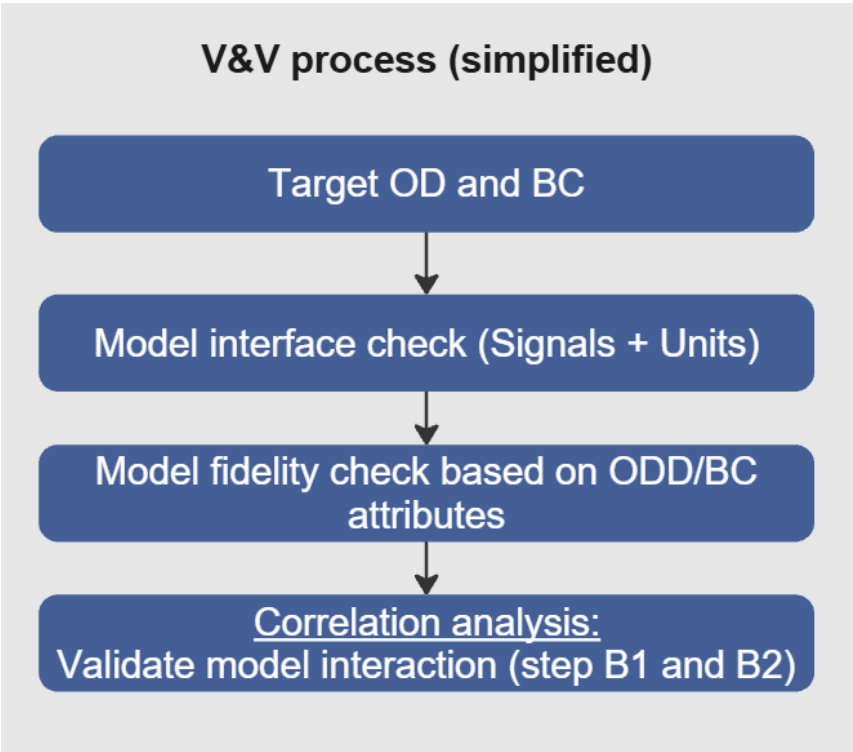


Figure 29: Simplified V&V process regarding the model interaction validation

3.4.1 Validation of simulated radar-environment interaction

Similar to validation setup A.2, however, the cause-effect justification is included as a prerequisite, as there is usually not one single cause for an effect or phenomenon but rather

a cause-effect chain that leads to a deviation from the GT [28]. An important note: If an environment simulation cannot simulate a cause, then no effect can be visible in the sensor model output – even if the real-world data includes such an effect. If an effect is observable at the output of a sensor (model), it is called a phenomenon [28]. Figure 30 shows the various layers of the so-called cause-and-effect tree. Starting from the signal propagation, including reception and pre-processing, various other aspects such as detection, feature, and object identification is included.

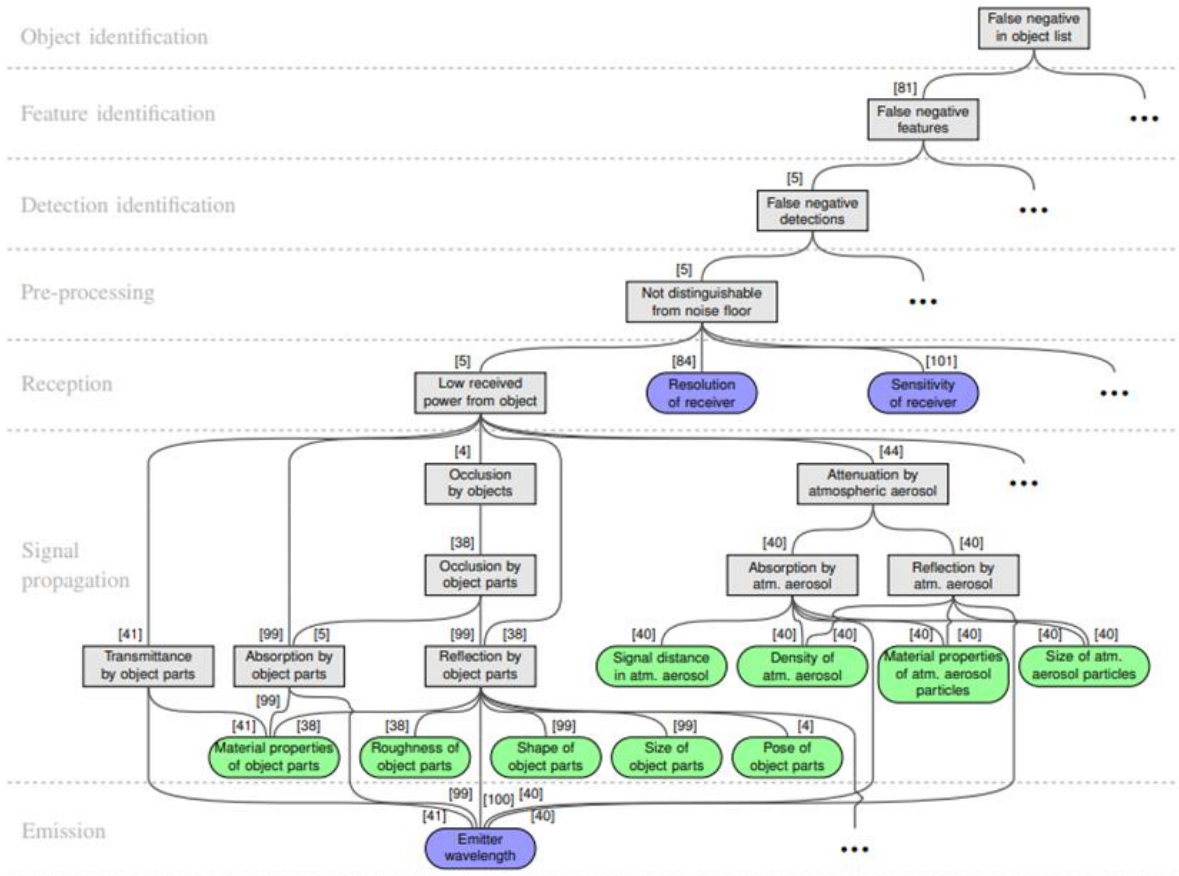


Figure 30: Exemplary perception sensor collaborative effect and cause tree [28]

3.4.2 Validation of simulated camera-environment interaction

This section covers the validation of the interaction between the camera model and the environment model. This interaction is the basis for the camera sensor simulation. The environment model is composed of the simulation engine and the 3D world model. The simulation engine is responsible for 3D world rendering, including illumination of the scene from different light sources (e.g., sun, car headlights), accurate reflectivity of objects in the scene, and rendering of different weather conditions such as fog, rain, and snow. The camera model simulates a physical camera in this rendered 3D world. High fidelity models include representations of the different physical layers of a real camera, for example, the lens and the colour filter array. The camera model outputs images, which are the input for the ADAS/AV perception software similar to images from a physical camera on a vehicle.

The validation of the camera-environment interaction assumes that the validation of the camera model and the environment model have already been individually performed as described in Section 3.2. For example, lab measurements have been performed for the physical camera to understand behaviour such as distortion, colour reproduction, and vignetting. The simulation model behaviour is compared against the lab measurements. Figure 31 shows an example of the validation of a physics-based camera model with respect to vignetting characteristics.

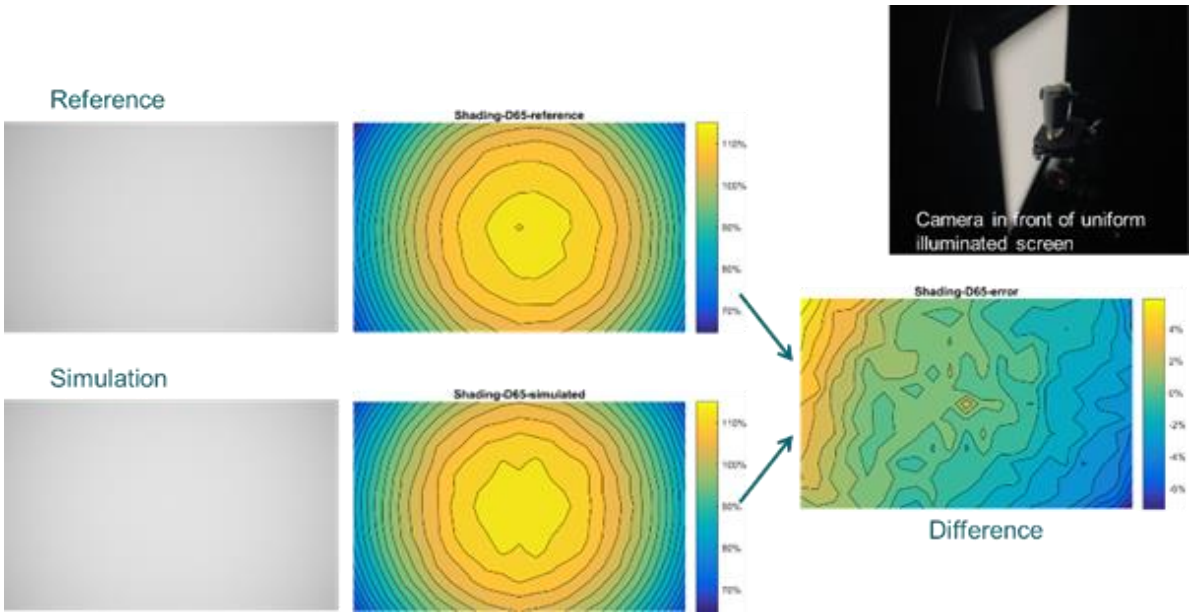


Figure 31: Example of camera model validation with respect to vignetting characteristics of the camera. The example shown is for the Simcenter Prescan simulation tool

Likewise, the environment model is validated. For example, light sources in the scene are modelled accurately and the propagation of light is modelled correctly and includes real-world influences such as reflectivity of surfaces, indirect illumination, and influence of weather conditions such as fog, pollution, and rain. Figure 32 shows how physical measurements are made for vehicle headlights, such that these light sources can be accurately modelled in a simulation tool.

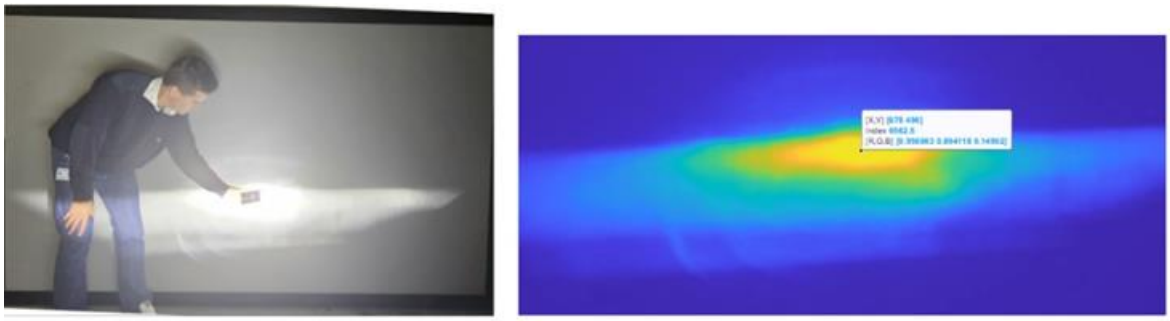


Figure 32: Left: Physical measurements for vehicle headlights modelling. Right: light map of the vehicle headlight in simulation. The example shown is for the Simcenter Prescan simulation tool

The ODD of the system-under-test and safety standards should be considered during the validation of the camera-environment interaction. Outside ODD conditions are irrelevant and may be ignored, while the validation should provide sufficient coverage of within inside ODD conditions. There are many studies on achieving coverage of the ODD, for example [29], however it remains an open challenge. Safety analyses, such as in the SOTIF standard [26], can be used to identify particularly important parts of the ODD to be considered during the validation. The SOTIF standard processes include Identifying weaknesses of the system (termed in SOTIF as functional insufficiencies) and conditions where the weakness poses a safety risk (termed in SOTIF as triggering conditions), e.g., extreme weather conditions. Therefore, identified triggering conditions is one area of the ODD to be focused on during validation.

It is well known that common triggering conditions for camera based ADAS functions are difficult weather conditions or low lighting conditions (see for example [30], [31]). Thus, it is pertinent that the camera-environment interaction Is validated in these conditions. Figure 33 shows the blooming effect in images, due to bright light sources. Such effects may occur for example, when the camera is pointed directly towards the sun, or due to bright car headlights in night conditions. Such effects may significantly affect the performance of ADAS function and therefore must also be representative in simulation.

Figure 33: Dash camera footage prior to a Tesla autopilot crash. The blooming effect due to headlights of the emergency vehicles parked in front may have been a contributing factor [32] .

Figure 34: Blooming effect due to the sun [33]

The validation of camera-environment interaction may be performed with a correlation analysis between simulation and real-world with respect to camera-environment interaction. This can be done in different ways as shown in Figure 35. For a digital twin world and camera model, we may compare the output images in simulation against those in real-world at a pixel level. However, this requires a high precision digital twin world model, as every detail is being compared. Instead, the validation may consider behaviour of the camera ADAS function. For example, attention maps of neural network-based algorithms may be compared between simulation and real world. Through this method, it becomes clear if there is a domain gap between simulation and real-world such that the ADAS function is affected. Finally, the validation of camera-environment interaction may also be performed considering the outputs of the camera-based perception algorithm, for example, list of detected objects. Here, metrics for object detection, such as precision and recall, may be considered.

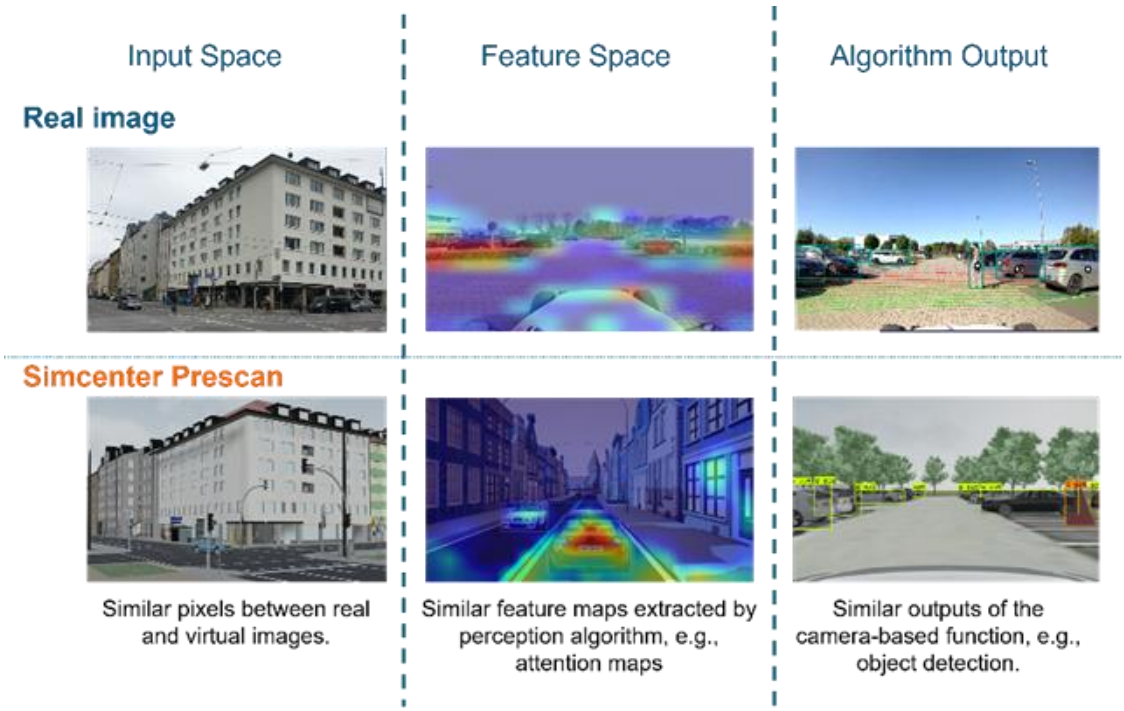


Figure 35: Different methods may be considered for the camera-environment interaction validation. Performing validation considering algorithm behaviour enables validation to focus on factors which affect the algorithms

A proper correlation analysis between physical tests and simulation requires the data from real-world to be gathered in a controlled environment, such that the real-world environment can be reproduced in simulation easily. For example, [34] performs physical testing for a camera perception algorithm in an indoor proving ground, capable of generating different illumination and weather conditions. Then, they reproduce the tests in simulation and compare the test results. Their findings show that the difference between simulation and real-world varies for different environment conditions. Such an analysis can provide a good indicator of whether the camera-environment interactions are sufficiently well modelled in simulation for different environmental conditions.

3.5 Methodology for Validation Setup C.1 and C.2

The output generated by sensors of methodology C1 is a detection list which is often called “point cloud” in case of lidar or “cluster points” in case of radar. Cluster points are defined by attributes such as position, speed, and radar cross-section, similar to osi3::Radar Detection in the ASAM Open Simulation Interface (OSI) [35]. For instance, a single vehicle may be represented by multiple clusters. These models generate raw or partially processed sensor data. For example, a radar sensor model may output cluster points with position, speed, and radar cross-section, while a lidar sensor model produces a point cloud. Sensors models of validation setup C.2, on the other hand, produce an object list as output. They typically contain information such as position, dimensions, velocity, and type. While they simulate perception and tracking processes, the actual details of these processes (e.g., the use of real perception software or stochastic generation) are often not included within the model.

However, as shown in the table below, the validation method for both sensor model types uses an open-loop approach with sampled test cases. Hence, a unified validation strategy is presented here. The final validation process is explained in two subsections.

Table 12: Validation setup “C” sub-model methodology split

Validation Setup C:	Description:
C.1	**Raw data replay and correlation** without perception and control function **Method**: Open-loop simulation for sampled test cases and comparison of raw sensor data **Criteria Examples**: Raw sensor data correlation, chassis movements with sensors, simul. performance, repeatability
C.2	**Replay and correlation** with perception system but without control function

	Method: Open-loop simulation for sampled test cases and comparison of object sensor data **Criteria Examples**: Object list correlation, position and classification of objects
C.3	**Replay and correlation** with perception and control function **Method**: Closed-loop simulation for checking the behaviour of the vehicle compared to real-world **Criteria Examples**: Control output correlation, e.g. driven trajectory, min. TTC, controller reaction times etc

As shown in Figure 36 the unified validation strategy is divided into five steps:

Figure 36: Compact overview of the validation steps presented in [36]

While the first three steps can be applied to validation setup C.1 and C.2 in general, steps 4(Metrics, KPIs) and 5(Validation) have to be considered individually. The five steps are outlined here:

Measurement Campaigns: Reliable reference data on object positions, movements, reflectivity, and environmental conditions form the basis for validating sensor models. This requires a well-structured and thorough approach to planning and conducting measurement campaigns.

Data Preparation: This step entails the meticulous selection and processing of trajectories from the measurement data for re-simulation in the co-simulation. Accuracy in data preparation is crucial, as even small errors can greatly affect the validation outcome.

Simulation: This step relies on precise 3D assets, especially for ray tracing in lidar and radar simulations. Open-loop simulation is adequate for sensor model validation, as it involves re-simulating recorded trajectories. This process produces synthetic sensor data, with further setup details outlined below

Metrics and Key Performance Indicators (KPIs): Establishing appropriate metrics and KPIs is a critical component of the validation process, as different sensor models and modelling techniques may necessitate tailored metrics and KPIs.

Validation: The final step entails calculating relevant KPIs from metrics using both synthetic sensor data and real measurements. The selection of scenarios is critical and closely tied to the effects being modelled by the sensor. Scenarios should be carefully chosen to ensure the modelled effects are clearly observable in the datasets. Although defining a specific validation process is challenging due to its highly customized nature, a general recommendation is to start with simple, static scenarios featuring reference targets suited to the sensor being evaluated.

As the focus is the validation methodology a closer look on the simulation setup and the construction of the trajectory is given here.

Simulation Setup for Sensor Model Validation:

Simulation environments are essential for validating sensor models, especially when handling diverse model types. A co-simulation toolchain enables seamless integration of various sensor models, proving to be an efficient approach. In [36] such a simulation toolchain is presented. It comprises three primary subsystems: the scenario player, which moves all objects; the 3D assets that form the virtual simulation environment; and the validated subsystems, the sensor models. The figure below illustrates this architecture and its data flow, adhering to ASAM standards. It is similar to the Harmonised V&V Simulation Framework in D4.4 [1] but reduced to environment and subject vehicle – sensor simulation.

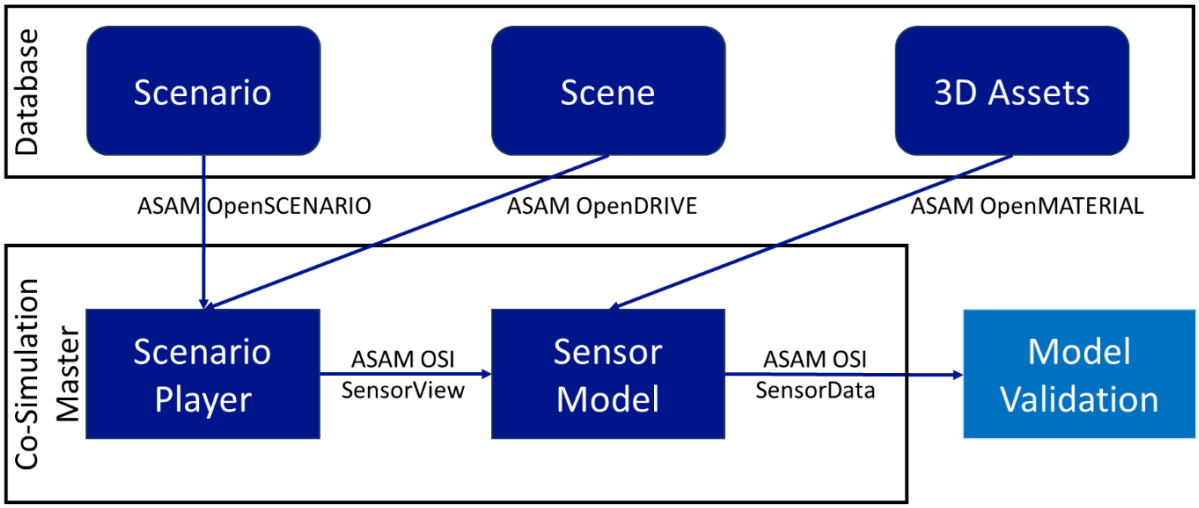


Figure 37: Overview of the co-simulation toolchain presented in [36]

Additionally, the reflectivity and geometries of all included 3D assets must be validated prior to evaluating the sensor simulation. It is important to note that this setup represents an open-loop simulation, without incorporating vehicle dynamics or traffic simulation for model validation. As a result, it excludes any ADAS/AD functions or driver inputs, relying solely on predefined trajectory re-simulation with no feedback loop from the sensor models to the scenario or environment simulation.

The need for these four subsystems varies based on the focus and validation requirements of the sensor model. The detail required in each subsystem is shaped by the modelled effects and the sensor model's ODD. For instance, a lidar sensor model with detection output often demands a highly detailed 3D virtual environment, including wavelength-specific material reflectivity, as the environment and sensor simulations closely influence each other. On the other hand, a stochastic sensor model with object list output may only require basic simulations of parameters such as vehicle positions and velocities, allowing for a simpler 2D birds-eye view simulation. Regardless of the method, "validating the environment simulation is essential; otherwise, unreliable data could compromise simulation-based safety validation." [36].

An essential element is the co-simulation master which is equivalent to test case manager in D4.4. There are various tools such as [37] or [38] to setup the co-simulation. While the first co-simulation tool supports a lot of commercial and non-commercial simulation tools, the latter one is limited to Functional Mock-up Units (FMUs) only. A major challenge is ensuring proper interface configuration between subsystems for efficient and accurate integration and data exchange. Thus, also here the ASAM OSI [35] standard is highly recommended for simulation data exchange, as it facilitates comprehensive communication of essential information. Sensor model integration largely depends on their implementation. A practical solution is using the Functional Mock-up Interface (FMI) [39], a widely adopted, programming language-independent standard for integrating subsystems in co-simulations. According to the OSI Sensor Model Packaging (OSMP) specification, FMI is the preferred format for simulation models that "speak" OSI within a co-simulation.

Discussions within major publicly funded projects on CCAM systems simulation-based testing, such as ENABLE-S3 [40], PEGASUS [41], SETLevel [42], and VIVID [43], have led to the establishment of international standards for simulation-based testing and validation in recent years. These standards, governed by ASAM e.V. as OpenX standards, include OpenSCENARIO for scenario description, OpenDRIVE for the scene and driving environment, and OpenCRG for road conditions and topology. Each standard intersects with sensor simulation, addressing elements like the movement of detectable objects, environmental conditions, and static features affecting sensor signals, such as guardrails, lane markings, or road roughness. Since early 2024, a new ASAM standardization project, OpenMATERIAL, has been addressing the description of geometries and material properties for 3D assets in the virtual static environment. Accurate descriptions of physical properties in the virtual scene are critical for ensuring valid sensor simulation

Trajectory Reconstruction

The re-simulation process, essential for validation, requires preserving trajectories and other elements from real-world scenarios, such as environmental conditions, reflectivity, and object geometries, derived from sensor data. To achieve this, the use of OpenSCENARIO or OSI TraceFiles is recommended, as both formats enable precise re-simulation of recorded trajectories.

The main goal is to estimate the true trajectories of detected vehicles with maximum accuracy to ensure reliable scenario information. In real-world scenarios, GNSS data for detected vehicles is often unavailable, necessitating trajectory estimation based solely on the perception sensors of the ego vehicle. The approach in [44] uses measured object data without relying on advanced perception software. A sophisticated post-processing filtering technique enables accurate reconstruction of relevant trajectories in most cases.

Figure 38 illustrates a reconstructed trajectory compared to a reference measurement and radar-based data, demonstrating the improved realism of the reconstructed trajectory through intelligent estimation. However, discrepancies between the reconstructed and reference trajectory, as seen in the Figure 38, are challenging to address since they fall outside the perception sensor’s measurements, such as front or rear radars.

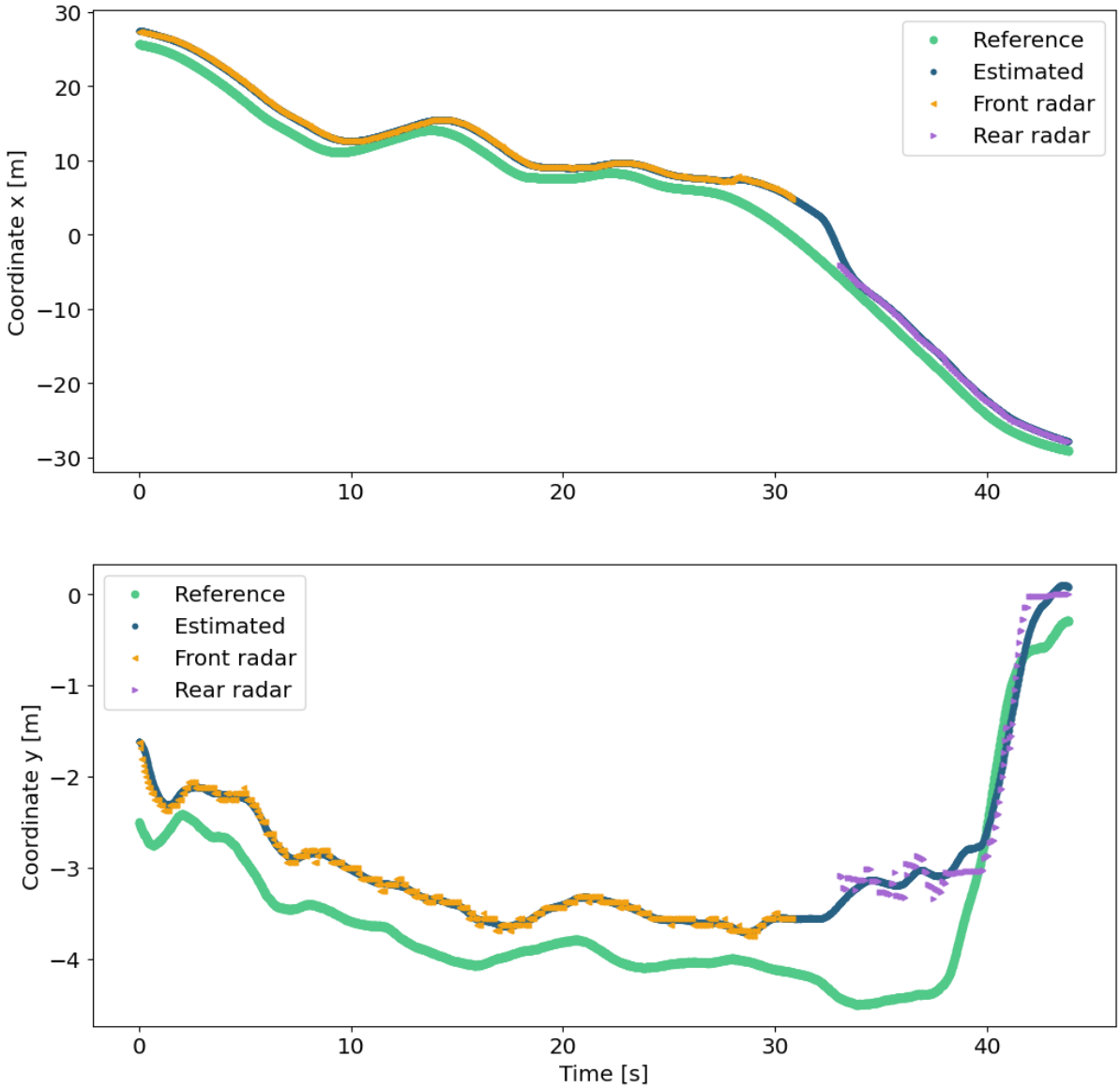


Figure 38: Example by [36] of reconstructed trajectory based on radar measurements only; the trajectory is from a classical overtake manoeuvre on a motorway

Using reconstructed trajectories along with the ego vehicle's trajectory, OpenSCENARIO files are created. These files are then incorporated into the co-simulation toolchain discussed earlier, enabling robust sensor model validation.

3.5.1.1 Metrics for C.1

Validation setup C.1 concerns model correlation on raw RADAR sensor data in an open loop for the integrated system validation.

Based on the definition of the different types of simulations given in [10], C.1 carries out an explicit open-loop simulation (E-OL). An E-OL is a re-simulation to obtain the sensor output of previously recorded driving scenarios. The validation is carried out after the signal processing

stage (e.g., using raw point clouds) but before the actual data processing (e.g., fusion, segmentation, tracking). Furthermore, performing the simulations open loop means that the raw data is replayed and correlated with the simulation output without further perception and control functions at play. These open-loop simulations are used for the sampled test cases to compare the raw sensor data.

Overall, this includes defining a reference manoeuvre (benchmark scenario) and tuning a simulation environment to reproduce the driving task virtually.

The presented methods so far represent general methods usable on raw data (e.g., point cloud) level. However, high-fidelity sensor models often include very specific effects, which are important for various cause-and-effect chains in the respective OD/BC of an ADS-equipped vehicle. Hence, specific methods are required to validate the accuracy of the modelled effect compared to real-world data.

The following propose possible criteria for the sensor effects phase noise, mixer non-linearity, and phase drift—all relevant for high-fidelity RADAR sensor models.

Phase noise: Phase noise (PN) is caused by random fluctuations in the phase of a signal due to non-ideal behaviour of the oscillators and the phase-locked loop (PLL). The PN limits the received signal-to-noise ratio (SNR). Hence, the weak object signal is buried under the PN of an adjacent strong object. Therefore, a range fast Fourier transformation (FFT) map with phase noise (distance over magnitude) is a valid option to determine the level of correlation between simulation and the real world. Concretely, for the phase noise, the peak shape and location (in terms of distance) are essential, in addition to the noise level. Therefore, a quantitative correlation and validation criteria considers the maximum value and the level of matching of the mean value to take the noise level across the whole range into account. Referring to Section 3.1, concrete criteria are the Frequency Spectrum Error Criterion (Section 3.1.2), as the signal are in the frequency domain. If the signals are transferred back into the time domain, Dynamic Time Warping (Section 3.1.2) can be suitable for the analysis. In addition, if the signals are discretized, the Discrete Fréchet Distance (Section 3.1.2) can be relevant. In principle, for determining the level of matching maximum values, the stated point data metrics in Section 3.1.1 can be used (using the two maximum values as respective points).

Mixer non-linearity: Any non-ideal mixer will output generally undesired, third-order intermodulation components. Due to their high amplitude, these frequency components can lead to false positive signal detections. The range over velocity (range-Doppler map (RDM)) can be a practical approach to quantifying the correlation level of this effect, as it is mainly relevant in the velocity domain. Concrete quantitative aspects include detected signals' accuracy (range/velocity).

Phase drift: The phase drift of transceiver (TX) and receiver (RX) channels describes the change of the output phase of one TX channel, mainly over temperature. Such a phase drift causes an angular estimation error and a degradation of sensitivity/SNR in the angular domain. To determine the correlation levels for this effect, the azimuth angle FFT is a valid representation, as the phase drift rate can be measured there.

3.5.1.2 Metrics for C.2

While metrics for validation setup C.1 for point data from Table 4 (section 3.1.1) can be applied to data points of single/static measurement, e.g. to the x-values of each point in the point cloud of a lidar measurement, for validation setup C.2 these metrics can be applied to a time series of the object list if the following approach is followed. The length in x-direction of a bounding box of a perceived object as shown in the figure below is considered [11].

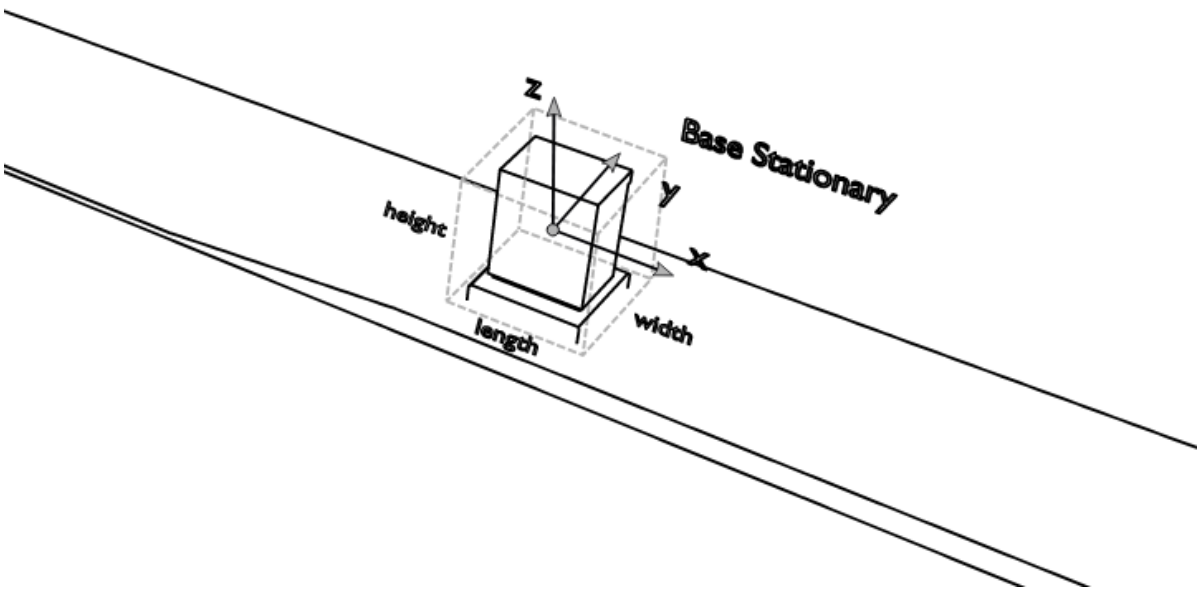


Figure 39: Bounding box of a perceived object

Assuming a scenario with 10s and sampling time of 0.1s would result in a time series with 101 length values (for sake simplicity is assumed that the object is detected all the time). The probability density function of the length can be calculated for both, the real data from the measurement and synthetic data of the simulation. As shown in Figure 8 by comparing the two density functions the metrics the model bias and scattering error can be evaluated. The same can be done with the width values in y direction and so on.

The disadvantage of this method is that there is no longer any temporal context, which is (more) important for attributes which are not a constant e.g.: the center positon (x,y,z) of the bounding box of a moving object or the velocity. Therefore, the error at each point in time can be calculated in additon. In example the x component of the center position of the bounding box: e_x(t) = x_pos_measurement(t) - x_pos_simulation(t). The result is again a vector with 101 elements, if we consider the same setting as before. The density function of this error vector can be calculated. Finally, the peak value of that density function corresponds to the bias error and the width to the scattering error.

3.6 Methodology for Validation Setup C.3

The validation setup C.3 has a combined validation methodology for the whole simulation toolchain that is used for the CCAM virtual validation itself, meaning all virtual components (models) that are needed for CCAM validation are validated at once (see Figure 6). This includes the validation of the combination of environment, vehicle and sensor models and the CCAM AD function which consists of the perception, path planning and control. The main difference to all the other validation approaches before and especially compared to validation setup C.2 is the integration of planning and control function as part of CCAM AD function. This means the main objective of this validation setup C.3 is the validation of the CCAM planning and control function model as part of the simulation toolchain which cannot be validated encapsuled from all the other models of the simulation toolchain. Since all the models are interacting with each other this validation methodology represents a closed-loop validation, means during simulation the environment is changing which affects the vehicle, sensors and perception. Based on the output from perception the CCAM AD function plans the path to be followed and controls the vehicle which affects the vehicle model and the environment (surrounding of vehicle is changing by its movement).

Most of the models or combinations of those can and should be validated by one of the previous validation methodologies, such as the vehicle dynamics model in combination with the road surface from environmental (e.g., by making use of validation setup B.1) or even with weather effects such as precipitation. The main reason is if there are many more non-validated models except of CCAM planning and control function a mismatch between virtual and real data could be caused by one of the other models and the identification of the wrong models gets worse with increasing number of non-validated models. Additionally, even if there would be a nearly perfect match between virtual and real results there might be errors in single models resulting in effects cancelling each other. But, except of these limitations validation methodology for C.3 can be used instead of all combined validation methodologies (not for validation setup A with standalone models) before depending on the test case design.

Since the methodology is validating the whole simulation toolchain the data for validation from real world must include all real components and therefore can only be validated with proving ground data (including black box) or field test data. Because the SUNRISE SAF describes a scenario-based safety assurance of CCAM proving ground data is more in focus. The same data or proving ground tests used for safety assurance of CCAM can be used for the validation of the CCAM planning and control function by simulation of the same test case and comparison of the results.

The methodology described in Figure 40 below starts with the purpose of validation, means which kind of models or which behaviour shall be validated. For each purpose a specific scenario cluster was defined that can be taken as reference to validate the according models or behaviour. For each of the scenario clusters some examples as test cases below are listed. However, these are just examples. Any scenario or test case can be chosen which are usually performed on proving grounds or for which already proving ground data exist such as test cases from regulations or Euro NCAP protocols. The data that usually can be provided from testing on proving grounds or from field tests are describing the lateral or longitudinal

behaviour mainly consisting of translational and rotational motion data including positions, velocities and accelerations. Other data can include list of classified detected objects as a result of the Perception layer. These data and the metrics from section 3.5 are used in a correlation analysis to come to a validation result of the underlying models.

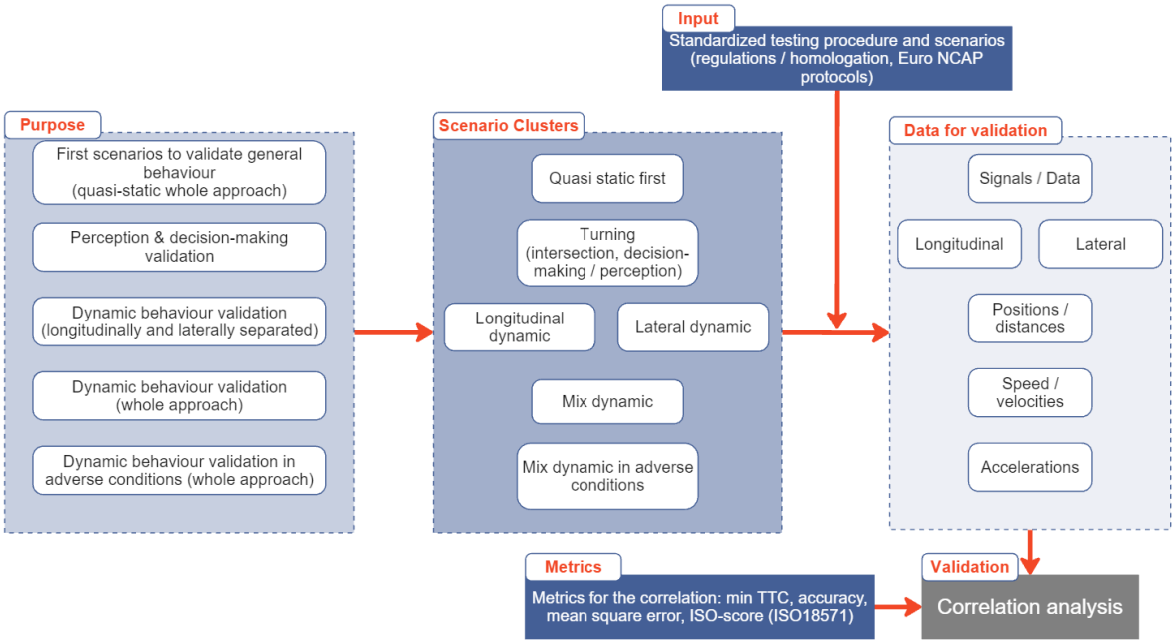


Figure 40: Model validation methodology C.3 (structure)

Quasi static test cases can be used to validate the combination of vehicle and environment model (low speed longitudinal or lateral movement, see B1 methodology) or at standstill the combined environment and sensor (see methodology for validation setup C.1, section 3.5) or perception models (see methodology for validation setup C.2, section 3.5). But also, these quasi-static test cases can be used to validate the AD function model compared to its real counterpart (such as in parking situations). The quasi-static scenarios are described by the criteria of longitudinal scenarios with speed < 15km/h and static objects. As examples CCRs [45] and CPRA [46] scenarios from Euro NCAP protocol are provided in the following Figure 41 and Figure 42.

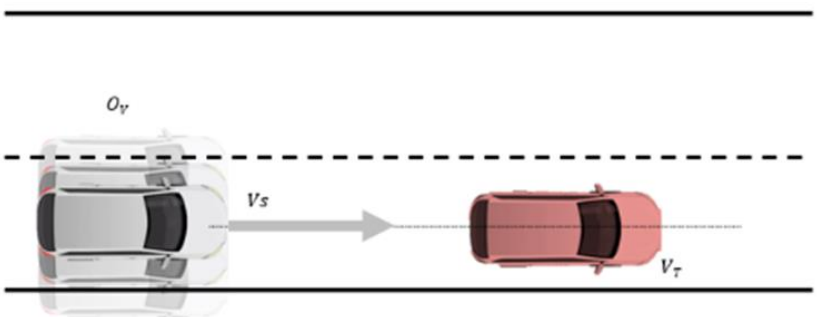


Figure 41: CCRs scenario [45]

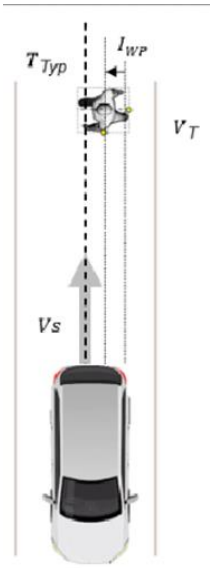


Figure 42: CPRA scenario [46]

The next cluster describes turning or intersection scenarios which are used to test the AD function behaviour at junction conflict situations and can also be used to compare the AD function model to the decision-making and control of the real system. These scenarios are described by the criteria of lateral scenarios, moving objects and a speed of up to 15km/h. As examples the scenarios CPTA (Figure 43) and CBTA (Figure 44) are provided from Euro NCAP protocol [46].

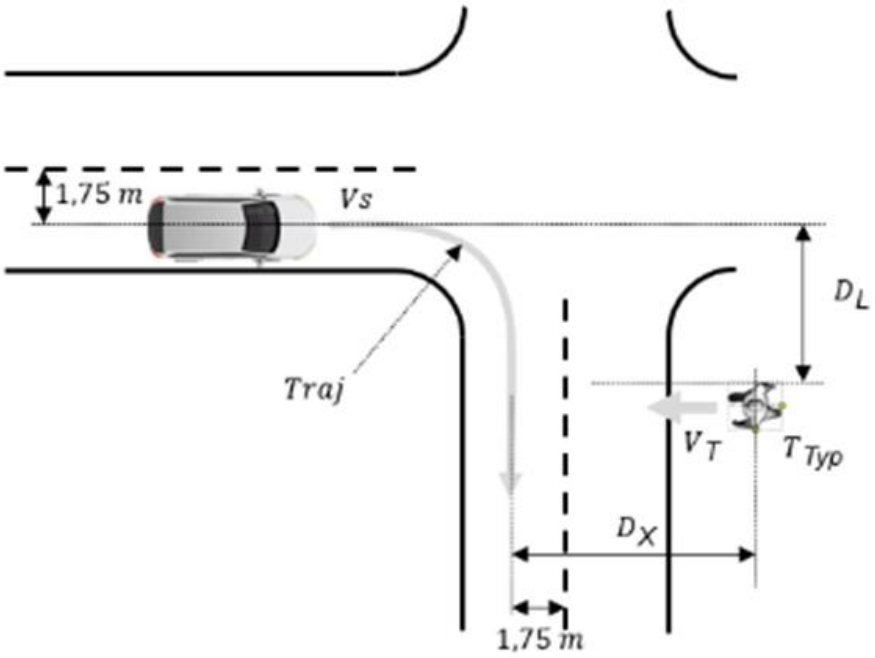


Figure 43: CPTA scenario [46]

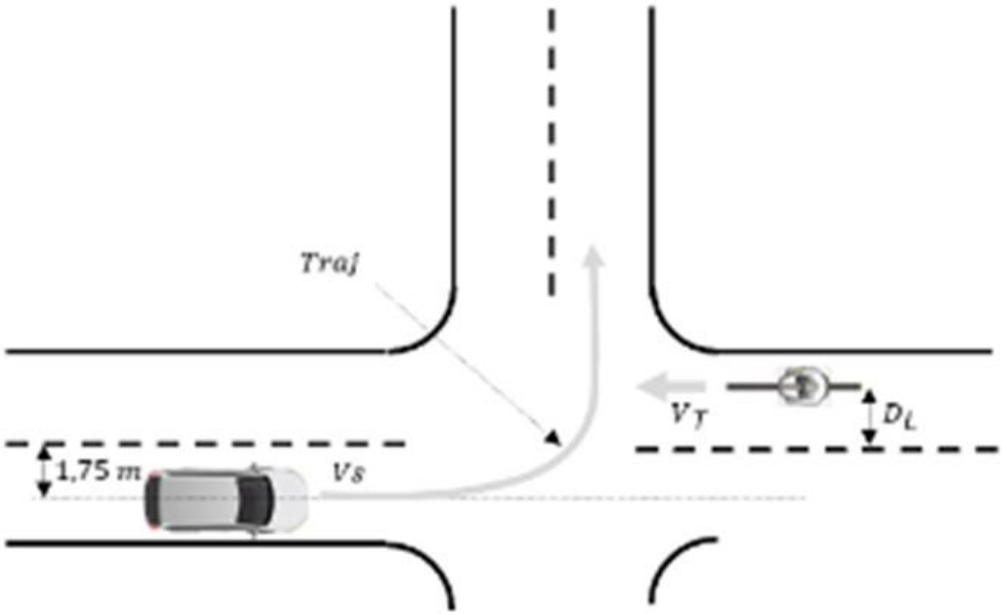


Figure 44: CBTA scenario [46]

The next cluster is divided into two separated sub-clusters and consists of the dynamic longitudinal (e.g., high accelerations, decelerations) and dynamic lateral (e.g., lane changes at high speed) scenarios. If this dynamic behaviour is more related to ego movement, then this kind of scenarios can be used to validate the vehicle model interaction with the environment model (see methodology for validation setup B1, section 3.3). But, if the dynamic behaviour is more related to the surrounding traffic (e.g., deceleration or cut-in/cut-out scenario) then this kind of scenarios can be used to validate the environment model in combination with AD function model or just the sensor and perception models (see methodologies for C1 or C2, section 3.5). The dynamic scenarios are described by the criteria to be longitudinal or lateral, by a speed of up to 90km/h and moving objects. As examples for dynamic longitudinal clusters are CCRm [45] (see Figure 45) from Euro NCAP protocol provided and (A)LKS scenarios [47] (see Figure 46) for dynamic lateral behaviour model validation.

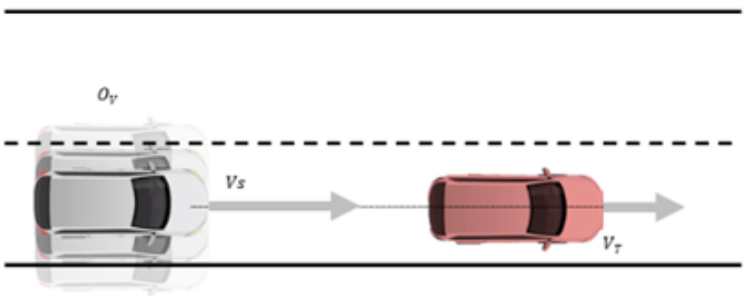


Figure 45: CCRm scenario [45]

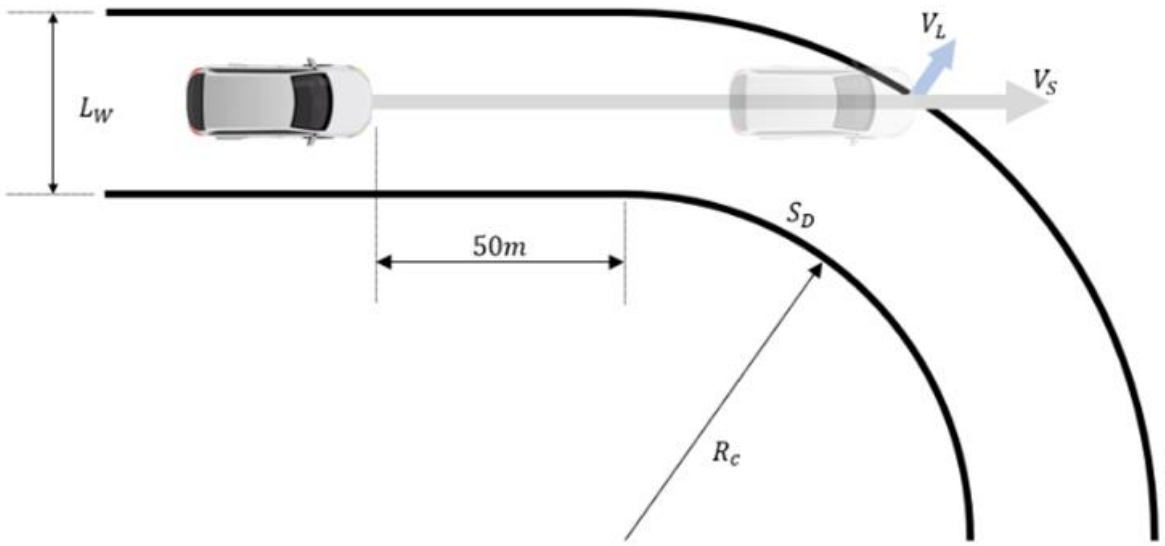


Figure 46: Lane keeping test scenario from [47] (Annex 5 section 4.1.1)

Also, a mixture of the dynamic scenarios can be used in order to validate the models' dynamic longitudinal and lateral behaviour combined (e.g., lane change with deceleration). Again, if the combined dynamic behaviour is more related to ego it would replace the methodology for validation setup B.1 and if it would be related to other agents as part of the environment then it would be used to validate the AD function model or just the sensor (see methodology for C.1) or perception models (see methodology for C.2). These scenarios are described by the criteria to be longitudinal and lateral, to be at any speed and moving objects. As example a S-bend CCRs is provided (see Figure 47).

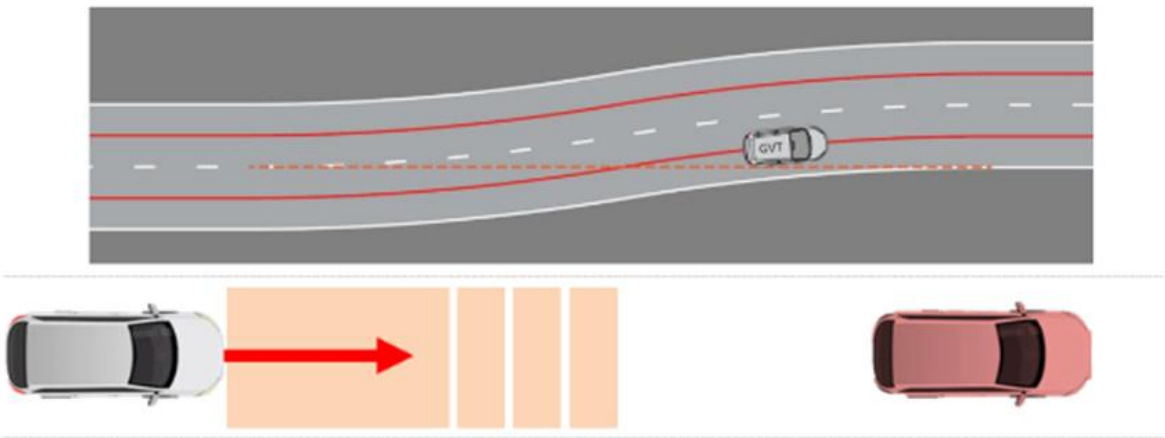


Figure 47: S-bend CCRs scenario

All of the upper mentioned clusters of scenarios would be not at adverse weather conditions to exclude any specific bias of one of the subsystems.

The last cluster describes mix dynamic scenarios at adverse weather condition in order to validate the models' behaviour at ODD boundaries or even outside the ODD. Except of the adverse weather conditions this cluster does not differ from the cluster before. The criteria to describe these scenarios are the same as before but added by the additional criterium to be at adverse weather conditions. As example an S-Bend CCRs at raining precipitation (Figure 48 below) is provided.

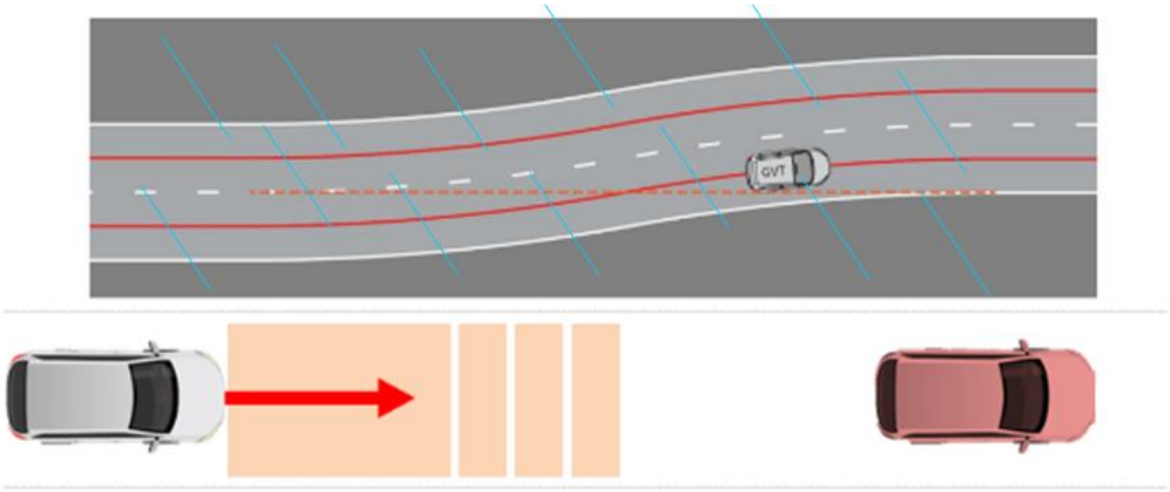


Figure 48: S-bend CCRs scenario at rain conditions

Please notice that not all scenario or test case clusters are relevant for all AD functions. For example, the turning or junction scenarios are not in ODD of an HWP or longitudinal or lateral dynamic scenarios maybe of less relevance for a turning or junction AD function.

3.6.1 Metrics for C.3

The correlation of the digital vehicle is a crucial step in the process of system validation through simulation. It enables the evaluation of the maturity of models in their ability to predict the behaviour of physical vehicles in various driving situations that may occur in real life. It is important to note that a numerical model should not be used outside its domain of validity. The domain of validity defines the conditions under which tests must be conducted to obtain reliable and usable simulation results, whether for design, validation, or even homologation, in compliance with the applicable regulatory requirements.

To analyse the correlation between real-world driving tests and simulations, various techniques could be employed to calculate indicators or metrics that measure the similarities and differences between time series data (see section 3.1.1).

3.7 Toolchain Functionality Methodology for Determinism and Repeatability of Simulation Results

This section will discuss on the factors to consider when it comes to determinism and repeatability of the simulation results. While the exact metrics are a function of the ODD & behaviour capabilities of the SUT, high level examples are provided in this section. This section will discuss the motivation on why determinism matters, followed by looking into determinism from two perspectives (scenario, and SAF), and it ends with the implication for WP3 T3.5 on allocation and re-allocation.

Determinism or repeatability can be subjective to begin with, a simple understanding of it is the ability to repeat the same execution in x number of times, so that the result is reliable, if nothing external changes, results are expected to be repeatable. Below Figure 49 shows a real-world use case on why determinism or repeatability is important: a customer wants to use a supplied model in a toolchain for the safety assessment of a system under test, after integrating with the supplied model with the customer's model, customer's model introduces randomness to the integrated system. For safety assurance process, being able to minimise and identify the source of such non-repeatability is crucial to the fair and trustworthiness of the assurance outcome of the system under test.

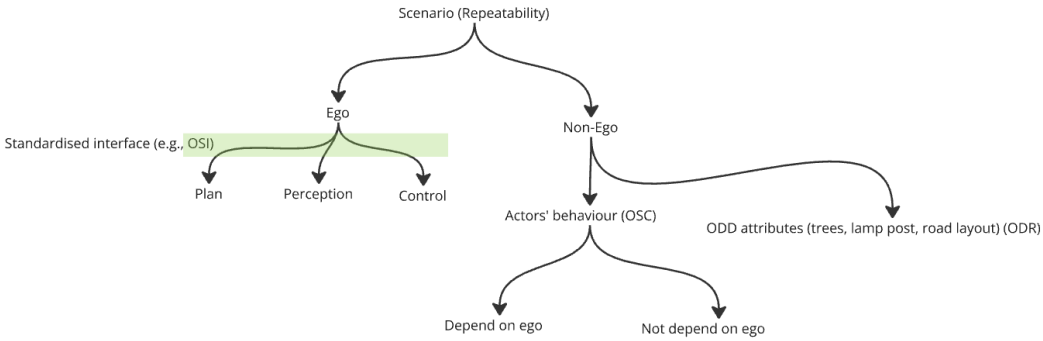


Figure 49. Potential sources of non-determinism from scenario perspective

To further dive into the sources of any non-determinism, two different perspectives are taken:

1. Scenario repeatability
2. Repeatability along the SAF components

As shown in Figure 49, within the components that form a scenario, at the highest level, they can be divided into Ego related aspect, and non-ego related aspect. The ego-related aspects can be further divided into plan, control, and sense. Simulation best practices such as the on-going SAE ORAD simulation taskforce are helping to address considerations including achieving determinism. The usage of common standards such as the ASAM open simulation interface (OSI) is a key enabler.

Within the non-ego related aspect, it can be divided into **behaviour aspect**, and **ODD aspect**. The behaviour part is mainly those that are captured within the ASAM OpenSCNEARIO 1.x, and the ODD aspects are mainly those within the ASAM OpenDRIVE. Please note that environmental conditions are part of the ODD but are included in the ASAM OpenSCENARIO 1.x. Areas that may arise non-determinism include the consistent and correct execution of the non-ego behaviour, and the consistency of the execution of the road layout. Non-ego behaviour can be further divided into ego dependent behaviour (such as using ego behaviour as trigger), and non-ego dependent behaviour.

The non-ego dependent behaviour can be majorly impacted by individual simulator's execution logic, especially for action-based behaviour description. For the non-ego behaviour,

as well as the other ODD elements representation, the scenario description language, together with the simulation execution logic play key roles. Going from functional, to abstract, logical, and concrete, each of the scenario format levels contains different level of flexibility, which will result into allowable non-repeatability.

For example, one functional scenario could result into infinite numbers of concrete scenarios, technically all those variations are considered as the correct execution of the original functional scenario, but they are not repeatable at the concrete level. Even at the concrete scenario level (e.g., ASAM OpenSCNEARIO XML), describing behaviour in actions versus trajectories will result in different execution. Therefore, having the right scenario abstraction level, and utilising common standards, and common execution logic are key aspects for maintaining repeatability.

When mapped to the SUNRISE SAF from

Figure 3, considerations for achieving repeatability sit mainly at the format and execute block. The former deals with the scenario format abstraction level and potential ambiguities (e.g., 'turn right' vs a concrete trajectory of a right turn), the later deals with the execution logic, model interfaces.

Another form of inherent non-determinism in execute block has to do with the limitations and constrains of the underlying simulation SW used in game engines typically used for driving simulations. As proposed in [48] different configurations and utilisations of the software and hardware can be explored to determine a 'virtual' operational domain specification where the simulation precision is sufficiently high for CCAM safety assurance testing.

3.8 Toolchain Functionality Methodology for Performance and Scalability

Simulation toolchains have become an integral part of developing Advanced Driver Assistance Systems (ADAS) and autonomous vehicles. These tools enable manufacturers and developers to test and validate algorithms in a controlled and safe environment while accommodating a vast range of scenarios. Although real-world testing is essential to validate the final product, simulation environments offer the advantage of evaluating numerous variations and situations that would be too challenging or costly to replicate in real world.

The **realism** of these simulation systems is a critical factor, as it ensures that the results obtained are transferable to real-world applications. However, achieving high realism comes at the expense of performance. Poor performance not only reduces the productivity of testing but may also fail to meet the minimum requirements necessary for specific tests, such as those involving co-simulation with real vehicles. Therefore, a key aspect of evaluating a virtual toolchain is its performance level, encompassing both its capabilities under specific configurations and its scalability. High performance and scalability enable increased

connectivity, the integration of automated agents, an expanded number of sensors, and enhanced realism in simulations.

Assessing the performance and scalability of a simulation toolchain is a hard task, as numerous factors influence simulation outcomes. Qualitative factors, such as simulation realism, cannot be measured quantitatively and require subjective evaluation based on project needs. In contrast, measurable elements like the number of agents, simulated sensors, and resource utilization provide objective metrics for evaluation. A thorough assessment must consider both qualitative and quantitative factors.

Table 13 outlines the primary variables affecting simulation performance. It is important to note that specific project requirements may necessitate variations in these factors. Consequently, an internal analysis of the simulator's requirements should precede any formal evaluation of its performance and scalability.

Table 13: Main variable aspects to have into account when measuring performance and scalability

Name	Affects	Measurable
Setup		
Licenses	Scalability	Yes
Complexity learning	Scalability	No
Complexity design	Scalability	No
Simulation		
Realism	Performance	No
Number of sensors	Performance & Scalability	Yes
Number of agents	Performance & Scalability	Yes
Number of connections	Performance & Scalability	Yes

A clear methodology and precise metrics are essential for effectively assessing the simulation toolchain. The following sections detail the approach adopted for the SUNRISE project, including the specific evaluation criteria and methods utilized to assess the toolchain's performance and scalability.

A fundamental aspect of evaluating a simulation toolchain is understanding its intended use. The requirements for a simulator vary significantly depending on the type of sensors being tested. For instance, a simulator designed for camera sensors must produce highly realistic images, while for radar sensors, visual details such as colours and textures are less critical. It is essential to identify the specific features required from the simulator to align with the project's objectives.

Once the use case is established, a set of scenarios of interest must be defined. These scenarios should be developed based on the system's specific requirements to ensure a

balance between critical edge cases and common use cases. In the SUNRISE project, the standard scenarios are outlined in [2]. The edge cases, however, should focus on situations that place high demands on the simulation, such as increased connections, a large number of agents, and complex simulated features.

From a scalability perspective, the evaluation should encompass two key areas:

- **Implementation Aspects**: This includes assessing the learning curve of the simulation toolchain and the requirements for expanding its capabilities (e.g., licensing or hardware needs). These aspects, however, are not evaluated in the scope of the SUNRISE project.

- **Scalability in Operation**: This involves evaluating the simulator's ability to handle increasing numbers of agents, sensors, connections, and users during execution.

The performance and scalability of the simulation toolchain will be assessed for individual sensors by systematically varying the number of agents and the resources allocated. These variations will help identify the toolchain's capacity to handle diverse configurations.

Metrics for evaluating the performance of simulation toolchains are essential for ensuring their efficiency, accuracy, and scalability across various applications.

Three key aspects must be assessed:

- **execution time**

- **resource usage**

- **scalability**

These metrics provide quantitative insights into computational efficiency, memory consumption, and overall system performance. By systematically measuring these factors, researchers and engineers can identify bottlenecks, optimize performance, and ensure that simulation toolchains meet real-world demands. Without these metrics, comparing different toolchains and selecting the most suitable one for specific project goals would be challenging.

To measure the execution performance of a process, time-based metrics are commonly used. Table 14 presents different metrics for this purpose. Since simulation runs can vary significantly depending on the scenario, the execution time metric for the SUNRISE project will be the time required to generate one scenario using the Scenario Runner format.

Table 14: Metrics examples to evaluate the performance and scalability

Metric	To measure	Description
Runtime	Performance	It measures the time taken to simulate on step

Throughput	Performance	Number of simulation steps completed for a given time
Latency	Performance	Time taken to start the simulation
Resources Usage	Performance & scalability	Percentage of resources used (CPU, GPU, RAM, memory, etc)
Speed up	Scalability	Increase of performance when more resources are given
Weak scalability [49]	Scalability	Performance retention when increasing both problem size and computational resources
Weak scalability [50]	Scalability	Performance retention when increasing computational resources while keeping problem size constant

The resources consumed by a simulation toolchain play a crucial role in estimating the computational power required to achieve project goals. Additionally, resource usage metrics provide insights into the scalability potential of a toolchain. The key resource utilization metrics to be tracked in this study include:

- **CPU/GPU Utilization** – Measures the percentage of processing power used during simulation.
- **Memory Usage** – Tracks the total memory consumed.
- **RAM Usage** – Evaluates the working memory required for efficient execution.

Scalability metrics assess how well a toolchain handles increasing workloads, ensuring it remains effective for large-scale simulations. In the scope of the SUNRISE project the speedup will be measured by increasing computational resources and comparing performance under different configurations. This will provide insights into how efficiently the toolchain can leverage additional resources to enhance performance.

3.9 Toolchain Functionality Methodology for Useability

The rapid advancement of autonomous driving technology has brought about a significant demand for efficient and reliable simulation toolchains.

These toolchains provide a virtual environment for testing and validating various aspects of autonomous driving software, ensuring its safety, performance, and adherence to regulatory requirements. However, as the complexity of autonomous systems continues to grow, ensuring the usability of simulation toolchains becomes very useful for the end-user. The **usability** of a simulation toolchain refers to its **ease of use**, **efficiency**, and **overall user**

experience. A user-friendly toolchain empowers developers and researchers to quickly and effectively create, test, and iterate AD algorithms.

Evaluating and comparing the usability of simulation toolchains can provide valuable insights into their strengths, weaknesses, and suitability for specific use cases. **Objective** and **subjective** evaluation methods can be employed to assess the useability of a simulation toolchain. **Objective** evaluations involve quantifiable metrics, such as **coverage of environment models**, **crash rates during usage**, and so on. On the other hand, **subjective** evaluations capture the **user's perception of the toolchain** through personal experiences, such as ease of **access to software support**. By combining both objective and subjective evaluations, a comprehensive understanding of a simulation toolchain's usability can be obtained.

This section focuses on evaluating the usability of a generic autonomous driving (AD) simulation toolchain under six subsystems defined in SUNRISE Harmonised V&V Simulation Framework and one additional subjective criterion defined below:

- **Environment**: The ability of the toolchain to simulate diverse driving environments, such as urban, rural, or highway scenarios.
- **Sensor**: Simulation toolchain's capability to emulate various sensors used in autonomous vehicles, such as LiDAR, radar, and cameras.
- **AD Function**: The toolchain’s support for simulating and testing different autonomous driving functions, including perception, planning, and control algorithms.
- **Vehicle Dynamics**: Fidelity of the toolchain in representing the behaviour and dynamics of vehicles in simulation, considering factors like acceleration, braking, and steering.
- **Traffic Agents**: The toolchain’s ability to simulate traffic scenarios, including other vehicles, pedestrians, or cyclists to evaluate performance of autonomous vehicles in complex environments.
- **Connectivity**: The toolchain’s capability to support communication and interaction between various agents/components of the simulated autonomous driving system, such as vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication.
- **Daily Usage**: The overall usability and user experience of the simulation toolchain in terms of ease of setup, configuration, support, and integration into existing development workflows.

The given analysis in this case was chosen to follow subsystems as described by D4.4 since authors believe that it provides more context on given subject as it is not a validation setup that follows the approach described in section 3.

Further below in this section each subsystem is elaborated by providing analysis of the evaluation criteria and methodologies to consider for the useability of a simulation toolchain.

ENVIRONMENT

Environment in the context of simulation execution refers to the virtual representation of the environmental conditions, as well as the scenery elements mentioned in a scenario. From an attribute decomposition perspective, scenario covers scenery, environmental conditions, dynamic agent, as well as behaviours, whereas ODD covers scenery, environment, and dynamic agent [51].

Figure 50 illustrates a snapshot of an example scenario (a lane changing cut-in scenario). By definition [52] in the ISO34503 ODD standard, scenery elements ODD attributes consist of spatially fixed objects of the operating environment (e.g., roads, traffic lights, etc), while their state may change such as bridges open and close; the environmental conditions shall consist of weather and atmospheric conditions (including information technology connectivity); and dynamic elements shall consist of the movable elements of the ODD (e.g., traffic, subject vehicle).

Please note that behaviour is not part of the scope of the ODD attributes. By mapping the ODD attributes to a scenario scope, scenery elements are usually covered by format such as ASAM OpenDRIVE, and environmental conditions and behaviours are usually covered by ASAM OpenSCENARIO xml. In addition to the attributes, within the scenario context, the scenery attributes will also need to consider the spatial relations among them to form a virtual environment. Similarly, for the environmental conditions which are normally covered in the ASAM OpenSCENARIO xml, any transitions from the temporal perspective needs to be reflected.

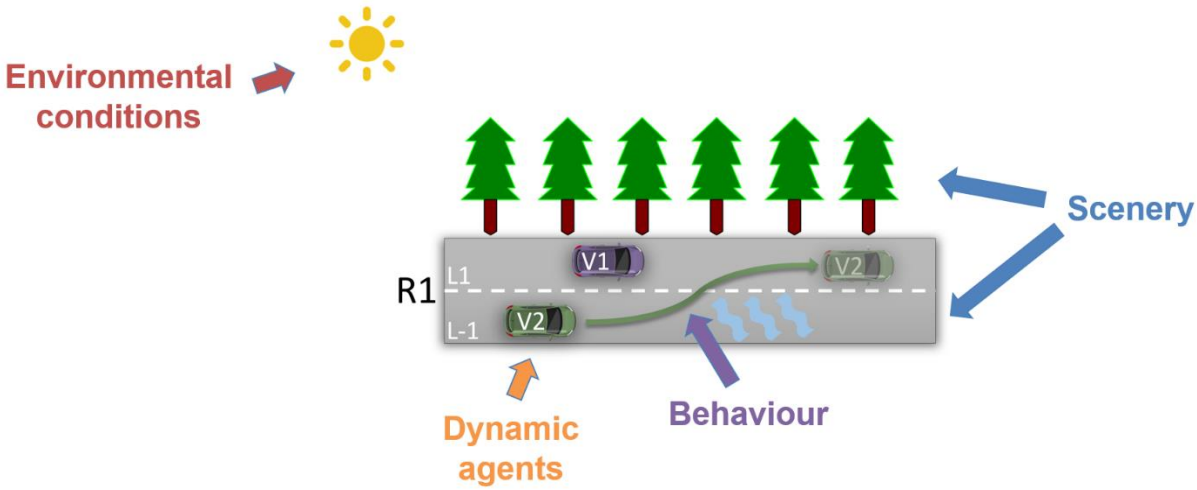


Figure 50: Snapshot of an example scenario

With the concepts of ODD, scenario in mind, combined with the understanding of temporal and spatial relations that need to be further represented based on the scenario, the usability

from the simulation perspective can be examined. The goal for using the simulation in the first place is to execute scenarios for virtual evaluation, this means that the simulation needs to:

1. Represent the attributes listed in the ODD taxonomy (e.g., asset library diverse enough to cover the complete ODD attributes range)
2. Accurately represent the spatial relations as defined in the scenario file
3. Accurately represent the temporal relations as defined in the scenario file

The first point can result into a coverage-based metric which indicates percentage wise how much the simulation can represent the ODD list.

The second and thirds can be measure in the forms of:

1. How much of such spatial and temporal features can the simulation support (e.g., OpenDRIVE road network notation)
2. If the simulation can support, how accurate can it support

To sum up, the Table 15 indicates the usability from environment simulation:

Table 15: Environment metrics for toolchain functionality

Metrics:	To measure:
ODD taxonomy attributes coverage	Performance & scalability & realism
OpenDRIVE function coverage	Performance & scalability & realism
OpenDRIVE function accuracy	Performance & realism
OpenSCENARIO function coverage	Performance & scalability & realism
OpenSCENARIO function accuracy	Performance & realism

SENSOR

Sensor in autonomous driving (AD) simulation toolchains is crucial, as sensors are the "eyes and ears" of an autonomous vehicle. They enable the vehicle to perceive its surroundings, providing data for decision-making and safe navigation. When evaluating the usability of simulation toolchains in terms of sensor simulation, several factors come into play. These factors help determine how effective and useable the sensor simulation module is for developers and testers working on autonomous systems.

The following metrics are used for this subsystem:

Table 16: Sensor metrics for toolchain functionality

Metrics:	To measure:
Sensor models are available at different levels of fidelity	Ease-of-use
Parameters can be configured to different physical sensors.	Realism
Outputs are in various standardized formats	Ease-of-use
Visualization tools are inherently supported or easily integrated.	Ease-of-use

Realism and Configuration of Sensor Models

Autonomous vehicles use a range of sensors, including cameras, LiDAR, radar, ultrasonic sensors, and GPS. An effective simulation toolchain must model these sensors from low to high fidelity in order to satisfy realism and hardware requirements.

Realistic sensor models allow developers to test and validate AD functions in a virtual environment that is as close to the real world as possible. Capabilities such as sensor noise, environmental interactions including lightning, weather conditions may be critical for some applications.

On the other hand, low-fidelity sensor models offer quick setup and don't require powerful computing machines; however, they may not be sufficient for validating an actual AD function in real-world scenarios.

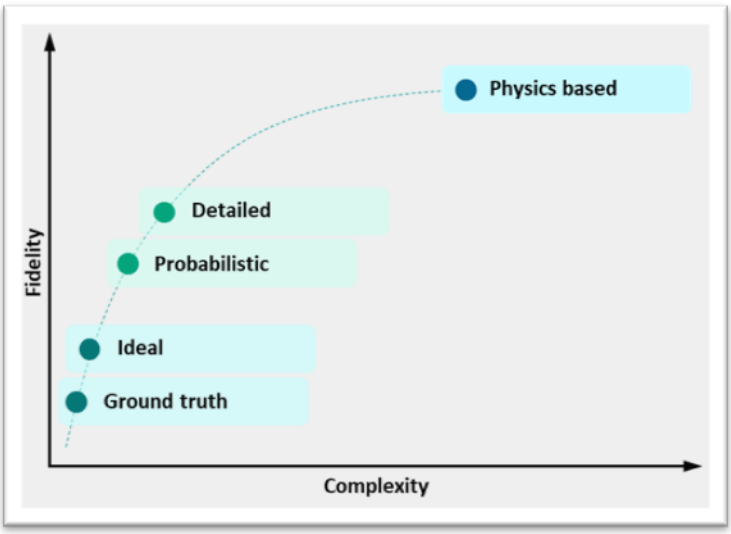


Figure 51: Overview of sensor fidelity with respect to complexity. The example shown is from software tool Simcenter Prescan

Additionally, a toolchain should allow for customization of sensor parameters (e.g., range, resolution, frame rate) to match specific hardware or experimental needs for both low and

high-fidelity models. For instance, an adjustable LiDAR sensor model could allow developers to replicate both high-density (64-beam) and low-density (16-beam) LiDARs.

Data Output Format and Compatibility with Processing Pipelines

The format in which simulated sensor data is outputted is critical. An ideal toolchain should support standard formats like ROS (Robot Operating System), OSI, or PCD files, which are commonly used in autonomous driving software stacks to provide a generic, standardized format to represent data from various sensors like cameras, LiDAR, radar, GPS, and IMU within simulation environments.

Data compatibility minimizes the time developers spend converting data formats and enables smoother integration with existing perception and processing pipelines. This is especially beneficial for teams working with a range of ADAS/AD toolchains.

Visualization Tools and Debugging Aids

Visualization tools can significantly enhance the usability of sensor simulation by enabling users to view sensor outputs (e.g., point clouds from LiDAR or image feeds from cameras) in a visualizing environment. This aids in debugging, calibration, and understanding how perception systems interpret their surroundings.

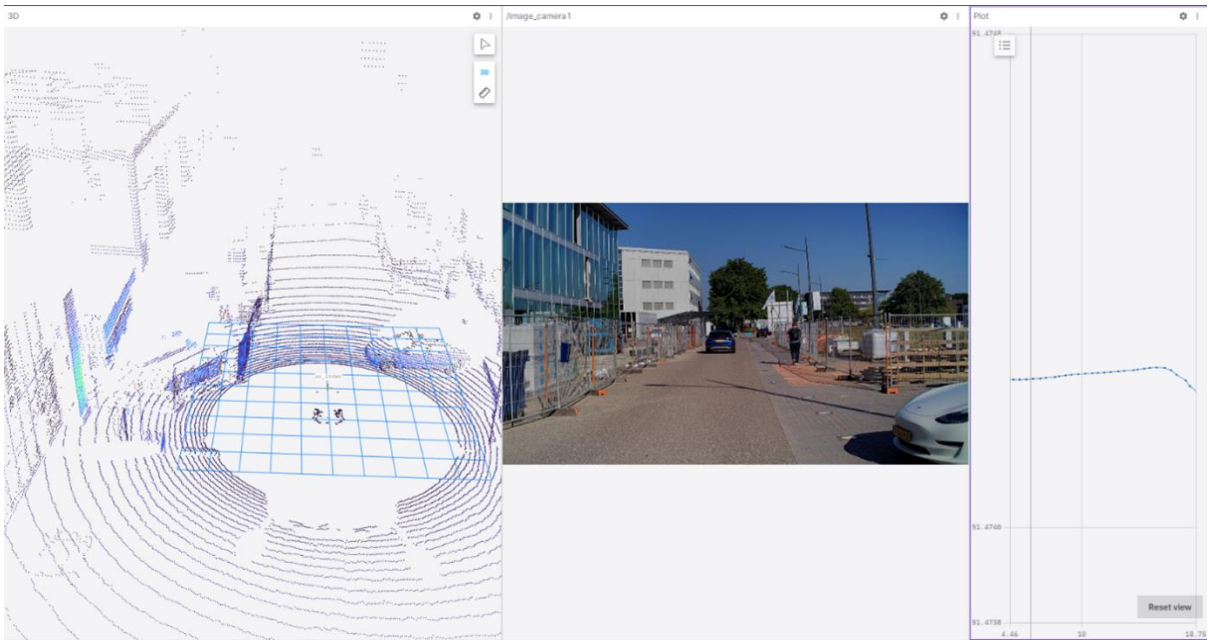


Figure 52: Sample sensor visualization of ROS2 data in Foxglove

Real-time visualization features, such as interactive 3D views, adjustable camera perspectives, and overlay tools allow developers to visually inspect how sensors respond to environmental changes, validate sensor coverage, and identify potential blind spots or artifacts in sensor data.

AD FUNCTION

The development of AD functions such as perception, planning, and control is central to autonomous vehicle software. Simulation toolchains designed for Autonomous Driving (AD) systems must offer specific features to facilitate the end user's ability to design and test these functions efficiently. The usability of a simulation toolchain for AD function development can be evaluated based on how well it supports the following key aspects:

Table 17: AD function metrics for toolchain functionality

Metrics:	To measure:
Modularity and extensibility	Ease-of-use

Modularity and Extensibility

A user-friendly simulation toolchain should support modular design, allowing developers to plug in or replace individual AD functions (like perception, path planning, or control algorithms) without needing to rework the entire software stack. This modularity enables users to test each function in isolation or in combination with other functions, accelerating the iterative development process.

Extensibility also allows developers to integrate custom algorithms and libraries easier, which can be essential for preparing simulations for requirements or testing innovative approaches.

VEHICLE DYNAMICS

Simulating vehicle dynamics is crucial for accurately modelling and predicting how a vehicle responds to different driving conditions. Vehicle dynamics in a simulation toolchain must encompass various aspects of real-world physics that impact vehicle behaviour, allowing developers to test, fine-tune, and validate autonomous driving systems in a virtual environment. This section discusses the features that are essential for vehicle dynamics within an AD simulation toolchain and how these features contribute to better useability for developers.

Table 18: Vehicle dynamics metrics for toolchain functionality

Metrics:	To measure:
Vehicle dynamics models are available at different levels of fidelity	Ease-of-use
Parameters can be configured to different vehicle types and specific models.	Realism
Visualization and data analysis tools are inherently supported or easily integrated.	Ease-of-use

Customization and Flexibility in Physical Modelling

The foundation of any vehicle dynamics simulation is an accurate physical model that reflects real-world physics. This includes models for **tire friction**, **suspension**, **braking**, and **steering**, as well as how these elements interact dynamically.

Usability improves significantly when users can quickly switch between high-fidelity (3D-15 DOF models) simulations and faster, less computationally demanding models (2D Bicycle models), which aids in the iterative development process. Simulation toolchains that offer these options in an intuitive, accessible manner, such as through graphical user interfaces (GUIs), make it easier for developers to manage trade-offs between simulation speed and detail.

Toolchains that include pre-built vehicle templates also improve usability by reducing setup time and allowing developers to focus on testing rather than building each vehicle model from scratch.

Additionally, a user-friendly toolchain should provide adjustable parameters for these models to enable customization, helping developers to simulate specific vehicle characteristics accurately, such as those of sedans, SUVs, or trucks, without having to delve into complex code modifications.

Data Logging and Analysis Tools

Developers need detailed insights into the results of their simulations to assess vehicle dynamics performance. Simulation toolchains that offer data logging and analysis tools, such as real-time telemetry or data playback can make it easier for developers to interpret issues with dynamics of simulated vehicle.

In the context of useability, these tools should be readily accessible and allow users to export data into common formats for external analysis or visualization tools including Python or MATLAB. Integrated analysis tools, such as plotting capabilities for acceleration, steering angle, or wheel slip may provide quick feedback, reducing time needed to export and use third-party software for debugging.

TRAFFIC AGENTS

Traffic agents can be divided into two aspects:

1. **agent appearance**
2. **agent behaviour**

Furthermore, traffic can be divided into:

1. **macroscopic traffic**

2. **microscopic traffic**

Therefore, the usability will be derived from the simulation capabilities for represent the above mentioned three aspects:

1. **appearance**
2. **behaviour at macroscopic level**
3. **behaviour at the microscopic level**

Within the ODD taxonomy ISO34503, traffic agent (from the appearance perspective) is a key attribute, which includes motor vehicle, non-motor vehicle, VRUs, animals, horse riders, and also special vehicles such as ambulance, police vehicle, work vehicle, traffic management vehicle, fire engines etc. Furthermore, due to the extensibility nature of the ODD taxonomy, user might want to extend the categories further to incorporate more diverse types. Such representations of the traffic agent types are required when creating both the microscopic traffic, as well as macroscopic traffic. To assess the usability of the simulation, metrics such as ODD coverage can be used to indicate.

For the macroscopic traffic behaviour, the ODD specification (ISO 34503 [52]) only indicates high level properties such as density of agent, volume of traffic, and flow rate. The choice of two of the three attributes can enable the specifier to define a unique specification. Density is the number of agents per unit distance, volume is the number of agents passing a reference points for a specific period of time, and flow rate is the rate at which agents pass a given point, expressed as agents per hour. These three key parameters as defined in the ODD specification provides a good metric to assess the simulation's usability to represent macroscopic traffic.

Furthermore, user might be able to define further requirements at the macroscopic level, such as percentage of the vehicles perform certain actions such as cut-in, such further requirements beyond the ODD attributes may be added to the required coverage when determining the usability.

For the microscopic traffic behaviours, these are the individual vehicle manoeuvres, usually defined within the ASAM OpenSCENARIO xml. They can be defined in several abstractions: trajectory level (e.g., waypoints-based information), action level (e.g., perform a stop), mission level (e.g., reach a destination). To measure the usability of the simulation to reflect such information, a two-step process can be used to define the metrics:

- Step 1 – how much of the behaviour definition from the target scenario format can be supported
- Step 2 – how accurate can those supported behaviour be executed.

To sum up, Table 19 below illustrate the metrics from the traffic agent perspective

Table 19: Traffic agents metrics for toolchain functionality

Metrics:	To measure:
ODD taxonomy attributes coverage	Performance & scalability & realism
OpenSCENARIO macroscopic traffic function coverage	Performance & scalability & realism
OpenSCENARIO macroscopic traffic function accuracy	Performance & realism
OpenSCENARIO microscopic traffic function coverage	Performance & scalability & realism
OpenSCENARIO microscopic traffic function accuracy	Performance & realism

CONNECTIVITY

Connectivity refers to the toolchain's ability to emulate interactions between the vehicle and its surrounding environment. This includes Vehicle-to-Everything (V2X) communication capabilities, where vehicles can exchange data with infrastructure, other vehicles, road infrastructure, and networks. V2X simulation capabilities are essential for testing and refining connectivity-based AD functions, especially for safety-critical features like collision avoidance, traffic flow optimization, and cooperative adaptive cruise control.

To evaluate useability of a simulation toolchain, several key features should be considered:

Table 20: Connectivity metrics for toolchain functionality

Metrics:	To measure:
Support for wide range of V2X communication protocols and common message types	Ease-of-use, availability
Parameters can be configured to test varying levels of latency, dropping of messages, and other realistic characteristics of V2X communication	Realism
Smart infrastructure agents for V2I testing scenarios	Availability
Visualization and data analysis tools are inherently supported or easily integrated.	Ease-of-use

<u>V2X Communication Protocol Support</u>

Simulation toolchains designed for autonomous driving should support a wide range of V2X communication protocols, such as Vehicle-to-Infrastructure (V2I), Vehicle-to-Vehicle (V2V), and Vehicle-to-Network (V2N).

A user-friendly simulation environment provides built-in libraries for protocols like Cellular V2X (C-V2X) or frequently used message types like ETSI CAM-CPM-DENM, allowing developers to easily configure, test, and evaluate connectivity scenarios without extensive manual setup.

Fidelity of Communication Models

For some connectivity testing, the simulation toolchain (in this case a co-simulation toolchain that integrates a network simulation too) should include high-fidelity modelling of communication channels, covering variables like latency, packet loss, bandwidth limitations, and range restrictions. Such realism may be crucial for simulating network conditions experienced in real-world urban, suburban, and rural environments.

A user-friendly toolchain should allow developers to easily adjust these parameters to create various testing scenarios. This flexibility enables accurate validation of how an AD system might react to poor or disrupted communication, which is essential for assessing connectivity robustness.

Infrastructure Integration for Smart City Simulations

V2X-based AD systems often rely on interactions with smart infrastructure like traffic lights, road signs, and construction warnings. A usable simulation toolchain should allow the user to configure and simulate smart infrastructure elements and test how the AD system uses this information to make decisions. For example, when approaching a simulated traffic light, smart vehicle should be able to receive real-time updates through standardized SPATEM messages and adjust its behaviour based on traffic signal changes.

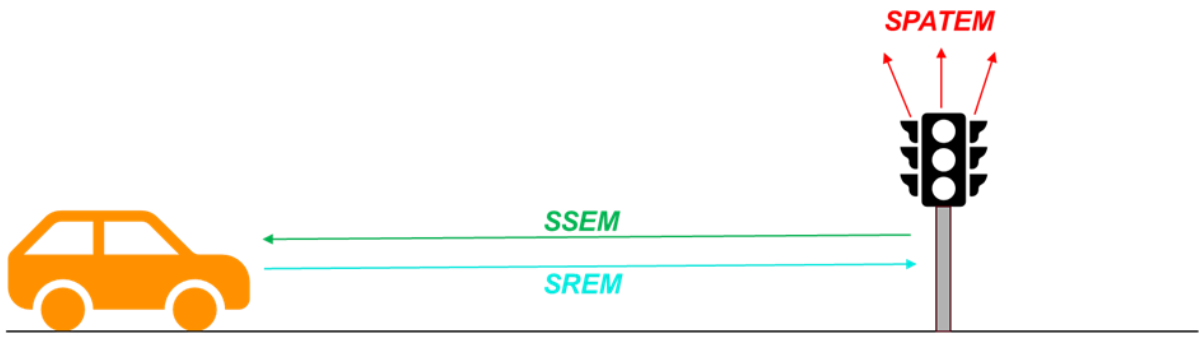


Figure 53: Messages used for V2X Traffic Light Priority Applications

It would be very user-friendly for simulation toolchain to allow end-users to easily add, remove, or modify smart infrastructure elements.

Connectivity Performance Logging

For effective development and testing, a toolchain should provide logging on connectivity performance, such as latency indicators, error rates, and message delivery success rates. This feedback may help teams to quickly identify and troubleshoot issues with connectivity-based functions, improving the efficiency of the simulation process.

Moreover, in a user-friendly toolchain, performance metrics should be accessible through an intuitive interface. Ideally, these metrics would be visualized in a dashboard format, allowing developers to monitor connectivity health alongside other key metrics like vehicle dynamics to avoid use third-party software for debugging.

DAILY USAGE

The following metrics are considered to evaluate how effective a simulation toolchain is for ADAS/AV testing:

Table 21: Daily usage metrics for toolchain functionality

Metrics:	To measure:
Seamless transition from real-world driving scenarios to test scenarios in simulation	Ease-of-use
Scalability and distributed testing	Availability, meeting demands of testing processes and standards

Seamless Integration with Real-World Data

One of the major challenges in AD development is bridging the gap between virtual simulations and real-world performance.

A simulation toolchain should offer the ability to incorporate real-world data—whether it's from previously recorded drives or sensor data, to make simulated scenarios as realistic as possible. The ability to import generic sensor data and be able to export scenarios in standardized formats such as Openscenario and OpenDrive format increases ease of integration. This integration can help developers to validate AD functions against real-world data, ensuring that the algorithms are robust and capable of handling real-world conditions. Meaning that support for some software functions such as importing real-world driving and sensor data, tools for benchmarking simulation data with real-world counterpart or fusion of real-world data with virtual elements can significantly increase useability of a simulation toolchain in context of AD functions.

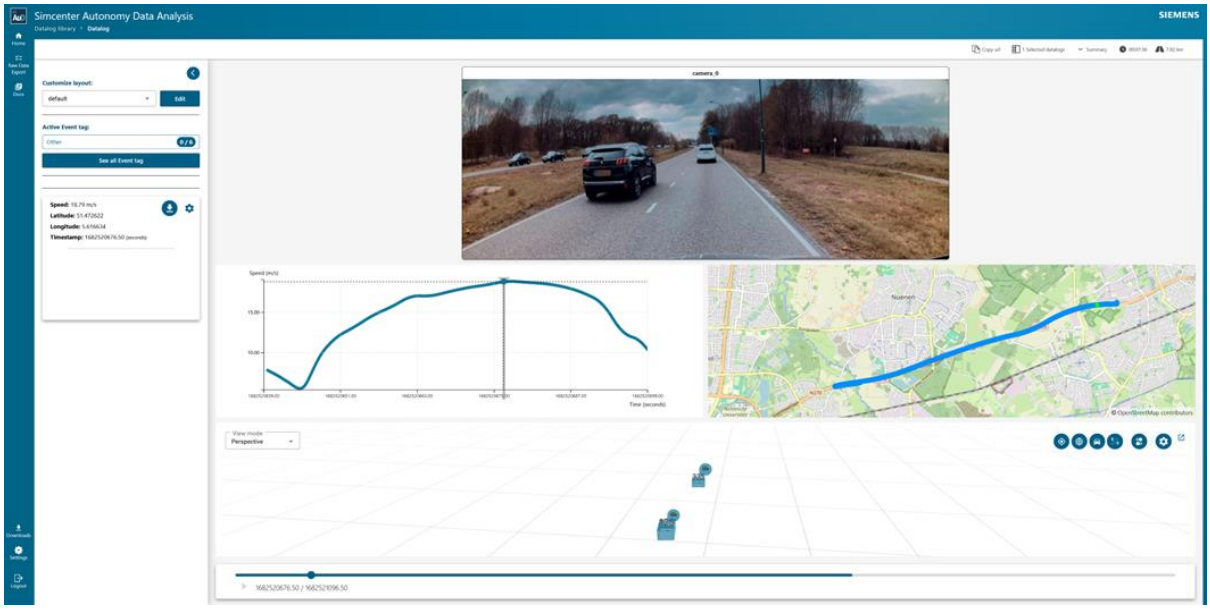


Figure 54: Example of Simcenter Autonomy which ingests raw sensor data from real world drives and generates scenarios in OpenScenario format for testing in simulation

Scalability and Distributed Testing

As AD functions mature and regulations get stricter, more effort is required to extensively validate AD functions across a wide range of scenarios and environments. A simulation toolchain that supports scalability, such as cloud-based or distributed in hardware, may allow engineers to run multiple tests in parallel, covering a broader range of conditions more efficiently.

This capability can be crucial for teams looking to simulate millions of miles of driving data to ensure safety and robustness before deploying AD functions in physical vehicles. For instance, introduction of SOTIF standard has forced teams to have a scalable way to discover unknown-unsafe scenarios to assess weaknesses on applications that possess thread to system safety. Therefore, the ability of running simulations at scale and using AI to find edge cases more efficiently is important.

4 ANALYSIS OF UC REQUIREMENTS RELATED TO VALIDATION SETUP

Based on use-case requirements defined in D7.2 and further refined in D4.2, section 4 will outline the requirements relevant to different validation setup elaborated in section 3. They will be categorized as to which setup they are relevant. This can assist partners in T7.3 by providing easier determination of appropriate validation setup for given simulation toolchain and requirements.

4.1 Urban AD perception validation (UC 1)

The scope of UC 1 - Urban AD perception validation is to validate the environment perception for SAE L3+ vehicles in urban and/or suburban areas, using a hybrid validation approach (virtual simulations and physical tests). Also, aspects of connected driving and collective perception are considered in this use case (as described in [2], [4] and [8]).

UC 1 - Urban AD perception validation includes three main sub-UCs as follows:

- sub-UC 1.1 - perception testing: covers sensor models used in the perception AD subsystem of an urban chauffer.

- sub-UC 1.2 - connected perception testing: builds on sub-UC 1.1 and covers the integration of information from other vehicles/VRUs coming from external sources via V2X and the use of C-ITS services, such as Cooperative Awareness Messages (CAM), Distributed Environmental Notification Messages (DENM), etc.

- sub-UC 1.3 - cooperative perception testing: builds on sub-UC 1.2 and covers the integration of information from other vehicles/VRUs coming from external sources via V2X and the use of C-ITS services, such as Collective Perception Systems (CPMs). More specifically, in UC 1.3 the focus is on the testing of collective perception service. This UC assumes the interaction of connected traffic agents with a local off-board Collective Perception (CP) system. The requirements relate to the validation of the combined connected CCAM system under test within a predefined urban ODD context (intersection, roundabout, darting out pedestrian). Focus is cast on the validation of a multi-agent system of systems that relies on V2X communication and for this reason the assumed simulation toolchain considers a co-simulation setup where AD driving simulation is combined with a network simulation and optionally a traffic simulation too. The specific simulation pipeline employed by each partner are detailed in [1].

In the following subsections 4.1.1, 4.1.2 and 4.1.3 simulation tooling specifications will be provided, based on the defined subsystem requirements in D4.2, in order to validate the CCAM systems presented in the above sub-UCs.

4.1.1 Perception testing (sub-UC 1.1)

Sub-UC 1.1 is intended to assess the perception modules when the operational design domain of the ADS includes urban environments. Each partner conducts tests on distinct AD designs, with each operational design focusing on the evaluation of a single sensor (camera, radar, or lidar) at a time. The specific simulation toolchains employed by each partner are detailed in D4.4.

Among validation setups, Sub-UC 1.1 focusses exclusively on validation setup A. Validation at the other levels requires a combination of real and virtual data, whereas Sub-UC 1.1 only utilizes either virtual or real setups, depending on the partner's approach.

Table 22 outlines the requirements for validating the simulation toolchain, based on the requirements defined in D4.2 for Sub-UC 1.1.

Table 22: Requirements relevant for sub-UC 1.1 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 1.1	R1.1_02 R1.1_03 R1.1_03_1 R1.1_03_4 R1.1_03_5 R1.1_03_7 R1.1_03_8	Radar sensor model	Relevant for validating the radar model at level A
UC 1.1	R1.1_04_01 R1.1_04_02 R1.1_04_03 R1.1_04_04	Camera sensor model	Relevant for validating the camera model at level A
UC 1.1	R1.1_05	LIDAR sensor model	Relevant for validating the lidar model at level A
UC 1.1	R1.1_11_01 R1.1_11_02	Perception DF required detections - Camera	Relevant when sensor models provide object lists
UC 1.1	R1.1_12	Generated scenarios requirements	Simulator should be similar to real cameras. Relevant for validating the camera model at level A.
UC 1.1	R1.1_22	Environment - landscape category	Simulator should be similar to real cameras. Relevant for validating the camera model at level A
UC 1.1	R1.1_17 R1.1_21	Vehicle model	Relevant for the vehicle model
UC 1.1	R1.1_18 R1.1_19	Traffic agents	Relevant for the behaviours included on the other actors

4.1.2 Connected perception testing (sub-UC 1.2)

Within UC 1.2 the focus is on the testing of connected perception. The requirements relate to the validation of GLOSA (Green Light Optimal Speed Advisory) and C-ACC (Cooperative Adaptive Cruise Control) within a predefined ODD context and which relies on V2V acquisitions, which is the main objective of Sub-UC 1.2. Therefore, a major emphasis is on the validation of V2V communication systems.

Based on the previously defined and refined requirements relevant for sub-UC 1.2 and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 23: Requirements relevant for sub-UC 1.2 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 1.2	R1.2_01	Validation metrics and KPIs	The validation metrics and KPIs shall comply with the Euro NCAP and GSR requirements.
UC 1.2	R1.2_03_04 R1.2_08_01	Connectivity	A realistic V2X connectivity simulation is required is relevant to C2.
UC 1.2	R1.2_03_01 R1.2_08_01	Environment/ ODD	Weather effects like illumination conditions, wind and adverse weather have to be validated according to B.2 and C.2.
UC 1.2	R1.2_05	ADS behaviour manoeuvres	Relevant for C.3, the simulation has to be able to replicate longitudinal forward driving manoeuvres and should correlate to reality.
UC 1.2	R1.2_06_01 R1.2_06_02 R1.2_06_03 R1.2_06_04	SuT required safe behaviour	Relevant for C.2 and C.3. In these scenarios, the perception shall work similarly in simulation and on proving ground. In C.3, additionally the behaviour of the ADS shall be similar.
UC 1.2	R1.2_07	Perception DF	Perception including ODD boundaries, static and dynamic objects, as well as V2X is relevant to B.2 and C.2.

4.1.3 Cooperative perception testing (sub-UC 1.3)

Based on the previously defined and refined requirements relevant for sub-UC 1.3 and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 24: Requirements relevant for sub-UC 1.3 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 1.3	R1.3_01	Validation of the test framework	Combination of physical testing and virtual testing into a hybrid setup that allows for C.2 and C.3. Ground truth data extracted from simulation and available real-world measurements can be used to analyse correlations between simulation and real world.
UC 1.3	R1.3_02	Validation of the test framework	Validating the interfaces used in the co-simulation toolchain that includes network and driving/traffic simulation components is relevant to C.2.
UC 1.3	R1.3_06 R1.3_12	Perception DF	Compare real vs sim perception results (object-level) without assuming common sensor models is relevant to B.2 and C.2.
UC 1.3	R1.3_04 R1.3_05	Connectivity	V2X network delays during CPM exchange can be compared between real and virtual CP testing setups when ETSI-compliant message exchange is used. This is relevant to C.2.
UC 1.3	R1.3_09	ODD/Dynamic elements	Integrate pedestrian motion from real world into CARLA simulator and compare motion model of CARLA in simple walking and in running. Relevant with A.3. Other road users can be passenger cars, busses, trucks and pedestrians.
UC 1.3	R1.3_07	ODD/Scenery	Manually inspect the produced digital map of both ICCS and VED premises with respect to lanes, slope, buildings. Relevant to A.3.
UC 1.3	R1.3_08	ODD/Atmospheric conditions	Effects of rain on perception can be studied in comparison between real and virtual worlds. Relevant to B.2 and C.2.
UC 1.3	R1.3_13	Scenario generation	Accuracy of representing real GPS locations into CARLA scenario can be considered. Relevant to C.1, C.2, C.3.

4.2 Traffic jam AD validation (UC 2)

As described in the deliverables D7.1, D4.1, D4.2 and D4.3 the scope of the **UC ID 2 "Traffic Jam AD validation"** is to validate the automated lane keeping system (ALKS) for SAE L3+

automated vehicles on motorways and motorway-similar roads via the implementation of a combined validation testing, including virtual simulations and physical tests.

The focus is on AD behaviour validation and optimisation of the workflow from test case generation to model creation and integration, as well as to test execution and assessment.

In the following subsection 4.2.1, we will briefly go through only the relevant requirements defined in the D4.2 and categorize them with relevant validation setup developed within this deliverable.

4.2.1 Traffic Jam Chauffeur speed limit adaptation

Based on the previously defined and refined requirements relevant for specific UCs and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 25: Requirements relevant for sub-UC 2.1 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 2.1	R2.1_15 R2.1_16	Vehicle Dynamics Data	Feature requirements relevant only for Methodology A, B1 and C1-3, all of the methodologies that have vehicle relevant subsystem.
UC 2.1	R2.1_24 R2.1_25 R2.1_26	Sensor Data	These sensor data requirements are relevant only for those validation setups that include object-based sensors and perception. Focus is on controls, not on raw-data perception that fits C3 validation setup.
UC 2.1	R2.1_38 R2.1_39 R2.1_40 R2.1_41 R2.1_43 R2.1_45 R2.1_50	Test Framework	Test framework RQs are relevant for almost all mentioned methodologies in T4.5 since the framework and used toolchain have the largest impact on the validation principles.
UC 2.1	R2.1_51 R2.1_52	User Requirements	Methodology for useability and simulation toolchain.

4.3 Highway (co-operative) AD validation (UC 3)

The scope of **UC 3 - Highway (co-operative) AD validation** is to validate semi/highly automated vehicles (SAE L2/L3+) on motorways (and similar roads) via the implementation of a hybrid validation approach (virtual simulations and physical tests).

UC 3 - Highway (co-operative) AD validation includes two main sub-UCs as follows:

- **sub-UC 3.1 - map-based perception & decision-making & control testing:** focuses on demonstrating how the vehicle's safety and awareness can be improved based on information coming from maps, sensors or connected services about road characteristics or road dynamic events.

- **sub-UC 3.2 - cooperative perception & decision making & control testing:** focuses on demonstrating how safety and surrounding awareness can be improved on motorways by including cooperative V2X functionality (with other vehicles in the neighbourhood) in the Highway Pilot (HWP) system (e.g., by leveraging and upgrading the driver assistance functionality developed previously in C-ACC from sub-UC 1.2).

In the following subsections 4.3.1 and 4.3.2 we will briefly go through only the relevant requirements defined in D7.1 and categorize them with relevant validation setups (see section 3) developed within this deliverable.

4.3.1 Map based perception & decision making (sub-UC 3.1)

Based on the previously defined and refined requirements relevant for specific UCs and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 26: Requirements relevant for sub-UC 3.1 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 3.1	R3.1_01	Validation metrics and KPIs	Relevant for all validation setups. Applicable quantities and criteria from Euro NCAP and GSR protocols shall be used for correlation analysis.
UC3.1	R3.1_02 R3.1_03 R3.1_04	ODD descriptions	Relevant for all validation setups since model quality shall be high inside ODD. Validation setups involving sensor and environment model (B.2 and C.*) shall test whether the required elements and ODD violations are sensed and processed similarly in simulation and on proving ground.
UC3.1	R3.1_05	ADS behaviour manoeuvres	Relevant for B.2 and C.*. In B.2 and C.1, the described items shall be detected similarly in simulation and on proving ground. In C.2, the available map data shall be compared additionally. And in C.3, the ADS shall detect and react to the described items similarly in simulation and on proving ground.
UC3.1	R3.1_06	SuT required safe behaviour	Relevant for B.* and C.*. In B.1 vehicle trajectories mimicking the manoeuvres of the SUT shall be

			considered. In B.2 and C.1, speed limit signs shall be detected similarly in simulation and on proving ground. In C.2, the available map data can be compared additionally. In C.3, processing of map data and the behaviour of the ADS shall be compared.
UC3.1	R3.1_07	Perception DF	Relevant for C.2 and C.3. The perception model must detect the described elements similarly in simulation and on proving ground.

4.3.2 Cooperative perception & decision making & control (sub-UC 3.2)

Based on the previously defined and refined requirements relevant for specific UCs and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 27: Requirements relevant for sub-UC 3.2 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 3.2	R3.2_01	Validation metrics and KPIs	Relevant for all validation setups. Applicable quantities and criteria from EuroNCAP and GSR protocols shall be used for correlation analysis.
UC3.2	R3.2_02 R3.2_03 R3.2_04	ODD descriptions	Relevant for all validation setups since model quality shall be high inside ODD. Validation setups involving sensor and environment model (B.2 and C.*) shall test whether the required elements, V2X messages, and ODD violations are sensed and processed similarly in simulation and on proving ground.
UC3.2	R3.2_05	ADS behaviour manoeuvres	The simulated model must achieve the capabilities defined.
UC3.2	R3.2_06	SUT required safe behaviour	Relevant for C.2 and C.3. In these scenarios, the perception shall work similarly in simulation and on proving ground. In C.3, additionally the behaviour of the ADS shall be similar.
UC3.2	R3.2_07	Perception DF	Relevant for C.2 and C.3. The perception model must detect the

			described elements similarly in simulation and on proving ground.

4.4 Freight vehicle automated parking validation (UC4)

The scope of **"UC ID 4 – Freight vehicle automated parking validation"** is to validate the environment perception and connected cyber-security perception for highly automated freight transport vehicles in confined areas via the implementation of a hybrid validation testing, by combining virtual simulations and physical tests. In SUNRISE project, UC 4 includes two main sub-UCs as follows:

- Sub-UC 4.1: Testing the perception & decision making of the SuT in a truck at low speed.
- Sub-UC 4.2: Testing the connected perception cyber-security of the SuT in a truck at low speed. More specifically, in UC 4.2 the focus is on the testing of a collective perception service under a cyber-attack context (physical or remote attack on the RSU camera) by combining the SuTs of UC1.3 and of UC4.1 and augmenting them with capabilities of falsified CPM creation (however, aspects already studied in UC1.3 or UC4.1 will not be in focus here). The specific simulation pipeline employed by the two collaborating partners is detailed in D4.4.

In both cases, starting from a pre-defined area, the truck will reverse into a loading dock. A sensor mounted on the loading dock will monitor the area behind the truck and communicate its observations to the truck.

In the following subsection we will briefly go through only the relevant requirements defined in the previous work packages and categorize them with relevant methodologies developed within this deliverable.

4.4.1 Truck low-speed perception & decision making (sub-UC 4.1)

Based on the previously defined and refined requirements relevant for specific UCs and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 28: Requirements relevant for sub-UC 4.1 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC4.1	R4.1_01_3	Environment	Feature requirements relevant for A, B, C1-C3. The ODD where the automated truck is operating must be properly simulated.
UC4.1	R4.1_03	Subject Vehicle: Control and Act	Feature requirement relevant only for C3 closed loop validation

			setup. Once the truck has planned its path, it would need to be able to manoeuvre precisely into the docking bay.
UC4.1	R4.1_08	AD system accuracy	Feature requirement relevant only for C3. The reverse function should be able to drive the vehicle from the start position to the end parking position.
UC4.1	R4.1_14	Vehicle dynamics	Feature requirements relevant for A, B, C1-C3.
UC4.1	R4.1_16	Simulation model validation	Feature requirement relevant only for methodology C3 where the validity of the simulation model must be shown.

4.4.2 Truck low-speed connected perception cyber-security testing (sub-UC 4.2)

Based on the previously defined and refined requirements relevant for specific UCs and the generic requirements, the following table outlines the relevant requirements for simulation toolchain validation developed within this deliverable.

Table 29: Requirements relevant for sub-UC 4.2 with respect to validation setup

Use Case	Requirement number ID	Requirement Category	Comment on Relevant Validation Setup Category
UC 4.2	R4.2_06	ODD/Environment	Co-simulation of cyber-attack features on top of what developed in UC1.3 is relevant to be validated through B2, C3
UC 4.2	R4.2_08 R4.2_09	ADS functional safety assessment	Safety and cybersecurity co-engineering should be supported by the SAF. Relevant with C3.
UC 4.2	R4.2_12	ADS functional safety assessment	Realism in CPM falsification in virtual environment testing is relevant with C3.
UC 4.2	R4.2_14 R4.2_17	Test framework (methods/tools/data)	Co-simulation features in utilized network simulation shall support realistic cyber-attacks. To be validated through C3.
UC 4.2	R4.2_19	Test framework (methods/tools/data)	Network simulation timing aspects when cyberattacks are integrated shall be analysed. Relevant with C3 and relevant with co-simulation determinism (see sec. 3.7).

5 GUIDELINE FOR SIMULATION TOOLCHAIN VALIDATION

The methodologies described in section 3 provide explanation to validation in an “ideal” case, meaning, no limitation of measurement data, tool and equipment accessibility or any other realistic limitation is considered.

The purpose of this section is to provide guidance to T7.3 within the SUNRISE project where limitations of particular use case is potentially impacting the generally recommended approach to validation methodology and metrics described in section 3.

The given guidelines are use-case specific, since most of the limitations and validation approaches depend on the simulation toolchain developed specifically for that use-case.

It has to be noted that the implementation of this guidance within T7.3 depends on the circumstances and limitations within T7.3 and involved partners, and it does not entail them to follow this approach exactly, but serves again as a blueprint, a recommendation of the approach to take in validating their particular simulation toolchain.

5.1 Guideline for Urban AD perception validation (UC1)

5.1.1 Validation Guideline for Use-Case 1.1

In UC 1.1, different partners employ different simulation toolchains, each using different sensors. Since no scenario within UC 1.1 involves both a real and virtual vehicle in the same setup, the validation process will focus exclusively on the sensors and the environment. Validation will involve comparisons with real data and virtual data when both a real prototype of the sensor and a reproduction of the environment are available. If these are not available, the validation process will ensure adherence to the requirements outlined in this section.

The primary focus of UC 1.1 is perception, and therefore only sensor and environment validation will be performed.

Radar model: The radar model validation begins by ensuring that all requirements specified in Section 4.1.1 are met. These requirements detail the general parameters and specifications the sensor must adhere to. If a real radar prototype is available, the methodology outlined in Section 3.2.2 will be followed, allowing for an estimation of the level of fidelity by comparing samples from the real and simulated radar sensors. Metrics such as the Euclidean distance between points, as specified in Section 3.1, should be utilized in this process.

LiDAR model: As LiDAR sensors generate point clouds, their validation follows a similar approach to radar, using the same metrics. However, a more controlled setup is required. In accordance with Section 3.2.2, validation should be conducted in a predefined environment

containing objects with varying reflectivity. The reflectivity values of real-world objects must be known to enable accurate comparison with the sensor's interaction data.

Camera model: The validation of the camera model follows the same approach as the radar model, but with additional focus on image-related phenomena such as distortion, lighting variations, and noise. When comparing with a real camera, these factors will be carefully examined. The tests described in Section 3.2.2 will be performed on both the real and virtual camera prototypes to facilitate direct comparison and ensure consistency.

Environment model: The validation of the environment model will leverage OpenDRIVE and OpenSCENARIO files to ensure that the Operational Design Domain (ODD) aligns with project requirements. The ODD will be verified through visual inspection. When real-world representations of the OpenDRIVE and OpenSCENARIO formats are available, these will be compared directly. Specifically, the OpenDRIVE files will be used to verify track and surrounding dimensions, while the OpenSCENARIO files will validate traffic user behaviors within the environment.

5.1.2 Validation Guideline for Use-Case 1.2

The simulation toolchain for Use-Case 1.2 is a composition of state-of-the-art simulation models that are coupled together. In addition to the general simulation toolchain requirements, Use-Case 1.2 requires a strongly validated V2X communication simulation. The transmission behaviour of V2X messages, such as delays, jitter, and signal range, must match those observed in real-life V2X solutions. Moreover, the full perception system must be validated since the propagation of perception errors through V2X is a key aspect of this use case.

The validation of the tool chain for Use-Case 1.2 can be partitioned by validating the core submodules through the following steps:

- **Environment Model**: The environment model shall be validated according to the validation setup A (to be exact, A3, section 3.2.3). A visual inspection should confirm that road geometries and configurations are correctly represented, and the behaviour and timing of traffic lights are accurately modelled.
- **Vehicle Model**: The vehicle model should be constructed from sub models that comply with the A.1 requirements. Given that this use case is complemented by physical testing, the vehicle model should undergo further validation using a validation process for validation setup C.1.
- **V2X Communication**: The V2X communication model should be validated at the object-list level using a methodology for validation setup C.2. The simulated object lists generated by the communication model are to be compared with communication data recorded from real-world tests, ensuring that metrics such as transmission delays and effective signal ranges are accurately reproduced.

After the virtual testing, the developed algorithms are tested in the physical world, on a proving ground with traffic infrastructure. Finally, a methodology for validation setup C.3 shall be followed between the results obtained in the virtual tests and the results obtained in the physical tests to confirm the robustness and representativeness of the simulated approach.

5.1.3 Validation Guideline for Use-Case 1.3

In UC 1.3, two partners employ distinct virtual and hybrid setups. In the virtual setup, the proposed co-simulation toolchains always include a network simulation. With respect to the hybrid setup, similar (ETSI-compliant) V2X messages are studied but different sensors, different real vehicles and different communication interfaces are used. The fact that also hybrid experimentation is considered allows for collection of proving ground data that can be also used for simulation toolchain validation. UC 1.3 hybrid setup involves a connected real and a connected virtual vehicle sharing a common virtual scene, therefore the validation may involve comparisons between real data and virtual data when both a real prototype of the sensor and a reproduction of the environment are available. If these are not available, the validation process should ensure adherence to the requirements outlined in Section 4.1.3

The primary focus of UC 1.3 is collective perception, and hence mainly ADS perception layer (object-level, this validation setup B.2) and V2X messages exchange (validation setup C.2) validation can be performed taking also in parallel consideration aspects of scenery and environment (validation setup A.3).

The following steps are considered useful for simulation environment validation of UC1.3:

- **Environment/Map Model** (see validation setup A.3): visual inspection of lanes, slopes, buildings in a small area selected as the test area.

- **Object-level perception Model** (see validation setup B.2 and C.2): per sensor or per perception layer; compare virtual only with hybrid (correlation analysis for object detection, see section 3.1) including weather/lighting effect if possible. For validation of perception results, it is recommended to use quasi-static setups where one agent is moving and one of the surrounding agents is static (this can be for example the parked vehicle in UC 1.3A).

- **Connectivity/CPM exchange Model** (see validation setup C.2 and C.3): Log messages exchange latencies and compare network delays between virtual only with hybrid when ETSI-compliant message exchange is applied, through correlation analysis (Kendal coefficient for rank order correlation can be used if the hybrid and virtual setup is not directly comparable, see section 3.1).

- **CP effect on ADS** (see validation setup C3): closed loop testing and compare virtual only with hybrid. This step assumes that the SuT also integrates control functionality in both real world and virtual testing environment and hence it is not mandatory for this UC that focuses on testing the perception layer.

- **Hybrid scenario accuracy**: Compare trajectories in real world versus those represented in simulated scenario through time series correlation analysis (see section 3.1). Visually inspect motion of real agents with motion models used by simulation toolchain traffic agent assets.

Repeatability in virtual scenario execution: Determinism offered by the proposed co-simulation toolchain should be studied (section 3.7).

5.2 Guideline for Traffic jam AD validation (UC2)

5.2.1 Validation Guideline for Use-Case 2.1

As scenarios which are derived from UN-R 157 in the scope of UC 2.1 main model quality checks include vehicle, environment, sensor and control function (SUT). Perception software is not included in the SUT package, therefore UC 2.1 simulation toolchain will generate object list data from virtual sensor models as described in [1].

Pure virtual testing environment described in [D4.4] is focusing on SiL environment with the focus on scalability and modularity which is achieved by CI/CD cloud execution environment.

The simulation models within simulation toolchain for UC 2.1 that are relevant for validation include **vehicle**, **environment** and **sensor model**. It is to be noted that SUT is the same as will be tested with the prototype vehicle, the validation of the SUT software will be done through physical testing, using black box testing approach as described in D4.6.

Considering the proposed validation methodologies in the section 3 and given the above conclusions we can summarize virtual validation approach for UC 2.1 simulation toolchain:

- SUT and perception software not validated for SiL
- SUT software is prepared to work with object-list data, from which we can conclude that no raw data sensor is modelling is needed or possible, but an object-list validation shall be carried out
- Vehicle + environment + sensor simulation sub-models' outputs are to be validated, making validation setup C.1 most appropriate. For more information on C.1 validation setup please refer to section 3.5.
- Still, sensor sub-models have to be validated for their mounting position, field of view and general specification of the sensor itself, but no output performance is evaluated since the models are ideal object based.

As shown in Figure 6, it is often recommended to start with validation of each separate simulation sub-model and then move towards right to more complex validation setups, but in this case, validating vehicle + environment is often easier then validating those two sub-

models separately, since capturing measurements in physical world is much easier where these domains overlap (e.g. Vehicle on a racetrack).

Validation of the above-mentioned simulation sub-models should be done according to description below:

- **Environment model**: Most appropriate validation methodology used for environment sub-model is the A validation setup described in section 3.2, specifically the part A.3 regarding environment model validation. Measurement data captured during proving grounds and/or real-world testing should be processed to extract trajectories of all relevant participants of the given test case. Additionally, a sufficient level of road description has to be extracted from measurements as well considering the requirement for environment model fidelity. The required level of detail for environment for UC2.1 is covered by OpenSCENARIO (trajectories for all relevant participants) format and OpenDRIVE (road information and network) format. Correlation analysis is to be carried out in order to compare ground-truth from simulation with the ground-truth data extracted from measurements.

- **Vehicle model**: In case of vehicle sub-model validation, an interactive vehicle-environment validation is most appropriate encompassing the methodology for validation setup B.1, additionally, only on-road measurements will be conducted, thus separate vehicle sub-model measurements will not be available for separate validation of the vehicle outside of the environment model (e.g. vehicle test-bench in controlled conditions). Similar as to the environment sub-model, ground-truth data from simulation is to be compared with the measured trajectory and dynamic behaviour of the Ego vehicle on the road. It is important to note the environment/ODD conditions under which the measurements are takes so the equivalent simulation ground-truth data can be extracted for correlation analysis.

- **Sensor model**: Validation of the sensor sub-model shall be carried out by executing the simulation in validation setup C.1 (vehicle + environment + sensor) producing simulated ground-truth data of detected object lists. It is important to note, that before the main correlation analysis according to methodology for C.1 can be executed, sensor specification metrics (mounting position on the vehicle, field of view, range, frequency, etc.) have to be separately validated and checked that they are correctly accounted for before proceeding with the correlation analysis for C.1. The simulated object list is to be compared with the object-list derived from the measurements ground-truth. To improve the correlation of the object-list sensor models, various error injection models (e.g. equation-based, statistical-based) shall be implemented to improve correlation. Same as with vehicle and the environment sub-model, road data shall be described in OpenDRIVE format, and trajectories in OpenSCENARIO format.

5.3 Guideline for Highway (co-operative) AD validation (UC3)

5.3.1 Validation Guideline for Use-Case 3.1

As detailed in section 3.5 of D7.2 [53], use case 3.1 will be executed in two setups, on a proving ground and in virtual simulation. To ensure that both methods yield comparable results for same input, different models used in the simulation toolchain will be validated using different validation setups. The models in the simulation toolchain that have to be validated are the environment model, the vehicle model, the sensor models and the interfaces in between. The AD system including the perception part itself and the provision of the map data information are the same as in the prototype vehicle. Thus, these two models do not need to be validated. The general approach is to start on the left-hand side of the validation setups instead of using directly methodology for validation setup C.3 to validate all models and the interactions at once. For validating the models of the simulation toolchain, the following data from proving ground testing will be available:

- Map of the road in OpenDrive format with the information of the curves and speed limits.
- Perception sensors output of the tests, both, in raw format and processed with the detections.
- Ground truth data of VUT based on GPS sensors installed in the vehicles.
- Inputs and outputs of the ADS in addition to the vehicle behaviour. This means that not only the vehicle behaviour is available but also the instructions and reference of the ADS.

To validate the models mentioned above the following validation setups and steps will be followed:

- **Environment model**: The validation setup A.3 (see section 3.2.3) shall be applied in order to validate the environment model which represents the simplest validation approach that can be chosen. This shall mainly cover the test tracks on proving ground and the movements of the target vehicles. To do so collect the ground truth data of the trajectories of the target vehicles and implement these in OpenSCENARIO format. The digitalised test track data in OpenDRIVE format will be used for all simulations from the test tracks where all tests will take place. These OpenDRIVE and OpenSCENARIO data will be used in simulations in order to validate the environment model, means to compare the visualised road network with the real one and to compare the trajectories of the target vehicles (ground truth data from simulation) to the implemented ground truth data from OpenSCENARIO files. By the ODD description requirements, trajectories that could have stemmed from highway scenarios shall be included in the tests. Vehicle model in interaction with environment model: The validation setup B.1 (see section 3.3) will be applied to validate the vehicle model representing the real prototype vehicle. The prototype vehicle will be not mounted on any test bench and no single components will be tested separately. Thus, it is only feasible to validate the

vehicle model in interaction with the environment which excludes approach A. To do so, collect driving dynamics and comfort data from proving ground testing in combination with ground truth information of the driven trajectory and implement this trajectory in OpenSCENARIO format. Together with the underlying road network in OpenDRIVE the trajectories will be re-simulated and afterwards the driving dynamics data compared to the collected data from proving ground tests. Since the original trajectory shall be followed this setup validates the vehicle model in interaction with the environment and trajectory following model, no influence by sensor, perception or ADS function needs to be considered. It shall be reported under which conditions of the ODD descriptions the correlation is checked and by R3.1_06 curved trajectories with varying speeds shall be included.

- **Sensor models** in combination with vehicle model and environment: The sensor models used in simulation will be idealised and return object-list data and will be validated by validation setup C.1 (see section 3.5). Therefore, no validation setup based on raw data would be feasible to use, which excludes validation setup A. Because, idealised sensors will be used only the sensor setup needs to be validated such as mounting position (including angles), FOV and range. In order to apply the validation setup C.1, collect the sensor object-lists from proving ground together with the trajectories' ground truth data and the road network. The road network will be implemented in OpenDRIVE format, the trajectories in OpenSCENARIO. By simulation the sensor object list data will be compared to the collected object lists from real tests in order to validate the sensor setup with positions (including angles), FOVs and ranges. As in the previous paragraph, it shall be reported under which conditions of the ODD descriptions apply. Trajectories of target vehicles shall include manoeuvres and positions relative to SuT described in R3.1_05. The C.3 validation setup (see section 3.6) will be applied to validate all models at once and especially the interaction of the models (including the interfaces). For this as much data as possible will be collected from proving ground including ground truth information of test tracks and moving objects, but also sensor object list data, inputs and outputs of ADS and driving dynamics data. Some of these information will be used for the simulation as input to generate a kind of digital twin which mainly consist of OpenDRIVE data describing the road network and the trajectories of the agents as part of OpenSCENARIO. Finally, the map data needs to be included the same way as for proving ground tests. All other data is taken for the correlation analysis between simulation and test track results, which includes the ADS inputs and outputs, sensor data and vehicle dynamics data. It shall be reported which combinations of items from requirements R3.1_02 to R3.1_05 and R3.1_07 were tested (e.g., adverse weather might not be possible on proving ground). Conditions triggering the behaviours of SuT from R3.1_06 shall be included in the tests.

By R3.1_01, the correlation of simulated and measured safety metrics (such as time gaps or TTC between pairs of vehicles) from test protocols shall be considered, at least if trajectories do not match perfectly. Since the AD function in simulation and on proving is equal, vector norms or the discrete Fréchet distance from section 3.1 are expected to be suitable metrics for the correlation analysis. In case of vector norms, result curves from simulation and proving

ground testing may have to be interpolated onto synchronized time instances. If other metrics from section are better suited will have to be decided based on the obtained results.

5.3.2 Validation Guideline for Use-Case 3.2

Since the most parts of the simulation toolchain are identical to the use case before (section 5.3.1) only the differences will be described in this section. All map data from section 3.5 should be omitted before applying it on this use case 3.2. Except for the missing map data from use case 3.1, the main difference to this use case is the addition of the V2X messaging and the corresponding models (receiver, transmitter and messages). Below the validation approach for the V2X messaging part is more detailed.

For all models except of the V2X models the guideline in section 5.3.1 should be followed. Notice should be taken to ignore all map data information in the guideline from section 5.3.1 and consider the changes below for the C.3 validation setup:

- **V2X messaging in interaction with vehicle model and environment**: The V2X messages (type and content) are strongly dependent on the movements of both interacting agents. Therefore, the corresponding V2X models can only be validated by considering relevant elements interacting with each other and the simplest methodology for validation setup B.2 was chosen. The perception and ADS function are not necessary for the validation of the V2X models if all possible types of V2X messages and the corresponding scenarios (trajectory constellations) are covered. In order to validate the V2X models the ground truth trajectories (implemented in OpenSCENARIO) need to be recorded from real tests, as well as the road network (implemented in OpenDRIVE) needs to be provided. Additionally, the exchanged V2X messages (sent and received) must be recorded with the according timestamps (which must match the timestamps of the trajectories) and re-simulated. For the re-simulation of the V2X messages one can make use of the OpenSCENARIO triggers and the UserDefinedActions. Finally, the exchanged messages from simulation need to be compared to the ones from real tests. It shall be reported which items of the ODD description requirements and which conditions of R3.2_05 for triggering V2X messaging were tested.

- For the C.3 validation setup mentioned in section 3.6 it must be ensured that the V2X messages from real tests including timestamps (which match the ones of the trajectories) are available in order to be able to re-simulate the scenario. Additionally, to the other data to be compared the V2X messages from simulation must be compared to the ones from proving ground test. It shall be reported which combinations of items from requirements R3.2_02 to R3.2_05 and R3.2_07 were tested (e.g., adverse weather might not be possible on proving ground), analogous to use case 3.1. From R3.2_06, triggers for messaging or the behaviours of SuT shall be included in the tests.

Analogous to the sensors (see Section 5.3.1), V2X messaging in simulation is idealized so that only type, content and time stamp of messages can be compared not raw data. Regarding R3.2_01 and metrics the correlation of other quantities, see the final remark of section 5.3.1.

5.4 Guideline for Freight vehicle automated parking validation (UC4)

5.4.1 Validation Guideline for Use-Case 4.1

In section 3.7 of D7.2 [53], it is detailed how RISE will implement Use Case 4.1 in both virtual simulations and a scaled physical testing environment using a 1:14 scale truck with semitrailer. The scaled physical test environment is used to validate the simulation-based test environment and the simulation-based test results. In physical test environment, the test object is a scaled model of the original truck. To ensure that both simulation-based and physical test approaches produce comparable results for the same input conditions, various models including the environment model, sensor model, and vehicular dynamics model used within the simulation toolchain will be validated in physical testing using the methodology for validation setup C.3. This methodology serves as the model quality validation framework, ensuring consistency and reliability across all models.

For RISE, the most suitable approach is to adopt the methodology for C.3 validation setup to validate all the simulation models and their interactions simultaneously in CARLA simulations. This comprehensive validation process ensures that the models and their integrated behaviour align with the desired accuracy and according to the given requirements.

The main simulation tools that are used and developed for simulation-based testing and verification of the freight vehicle (truck with trailer) automated parking system is CARLA and Waywise. Carla provides perception simulation, visualisation, and vehicle dynamics. Carla ROS bridge for controller interface and data logging. WayWiseR supports ROS2 based communication to control Carla and objects in Carla. ControlTower is part of Waywise and used as test setup and orchestration system.

These tools and their interaction are presented in the simulation tool chain diagram below in Figure 55. The aim is to perform the feasibility of safety validation for the complete system in simulation by comparing the simulation results with the results obtained from the physical testing (miniature/scaled truck model). From the simulation functionality validation perspective, the focus is on repeatability and determinism.

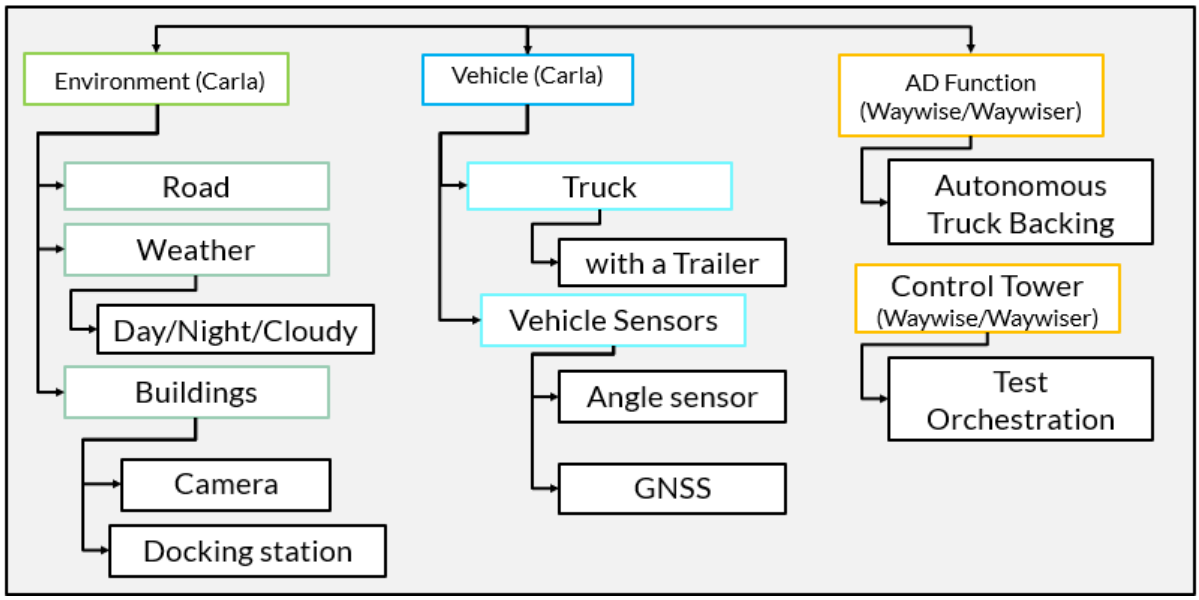


Figure 55: Overview of RISE simulation toolchain for UC4.1.

It should be noted that due to the unavailability of the actual truck, a scaled truck model was used for validation. While this approach provides valuable insights, the behaviour of a scaled model may not perfectly translate to a full-scale vehicle such as differences in weight distribution, tire dynamics, and other physical properties could influence the accuracy of the test results.
For Use Case 4.1 following guidelines are considered useful for validation of the simulation models following the methodology for validation setup C.3:

Environment model: The environment model is validated such that all relevant geometries are properly modelled and scales correctly between physical set-ups and simulation model, e.g., the driving area, buildings, start and stop positions for the truck with trailer. Other real-world conditions such as lighting and friction are also relevant and should if possible be validated, but unfortunately, they are known to not precisely modelled in Carla.

Vehicle model: The truck with semitrailer model is validated by comparing simulated trajectory following with physical tests. It is important to note that the model in Carla basically is two separate vehicles that are mathematically connected that is expected to behave similarly to a kinematic model and not as a full dynamic model. Physical parameters like, e.g., friction and tire dynamics will not be correctly simulated.

AD function model: The validation of the AD function model is strongly connected with the validation of the vehicle model as it is validated by comparing simulated trajectory following with physical tests.

5.4.2 Validation Guideline for Use-Case 4.2

This UC builds upon the virtual test environments of UC4.1 and UC1.3 and focuses on integrating cybersecurity aspects on the level of a local collective perception system and hence simulation toolchain validation will focus on the additional simulated components needed for cyber-attack support (through CPM falsification).

Important note: As UC 4.2 is only tested in virtual environment as the physical setup is hard to obtain, most of the parts reported in this section cannot be validated in the context of the project.

Since the SuT refers to a vehicle and an RSU involving connectivity and cybersecurity aspects, the most suitable approach is to adopt the methodology for validation setup C.3 to validate the combined simulation models (co-simulation approach) and their interactions simultaneously. For this purpose, a CCAM-enabled proving ground (where V2X is supported by the tuck and a camera- equipped RSU) is assumed to be available. Validation is enabled by comparing the behaviour in the virtual co-simulation world versus the physical confined environment.

The following steps are considered useful for simulation environment validation of UC4.2:

- **Cybersecurity aspects** - CPM falsification Model: Two types of CPM attacks are supported by this UC, namely camera spoofing via physical attack or CPM falsification via channel tampering. The realism of both types of emulated attacks (especially the one that involves a real camera HW) need to be compared against real-world data (e.g. image and CPM recorded data) either by using public datasets or by reproducing the scenario in a proving ground.

- **Cyber-attack effect on ADS** (see validation setup C.3): Perform closed loop testing in the virtual setup and compare effects in virtual versus proving ground. This step assumes that the SuT also integrates control functionality in both real world and virtual testing environment. The effects of both camera obfuscation though a light source (validation setup B.2 is relevant here) and CPM falsification through the network in simulation (here methodology for validation setup C.3 shall be used that apply to the whole co-simulated system of systems) have to be compared against real-world similar situations in controlled environment when ETSI-compliant CPM communication is tested. For this purpose, standard correlation analysis techniques can be used (see section 3.1).

- **Latencies and determinism**: Network simulation timing aspects when cyberattacks are integrated shall be analysed. Relevant with validation setup C.1 and relevant with co-simulation determinism (see section 3.7). Log and compare CPM exchange latencies in real world versus those represented in simulation under the two types of cyber-attack scenario through time series correlation analysis (see section 3.1).

Safety and cybersecurity co-engineering: The simulation toolchain should seamlessly support simultaneous testing of safety and cybersecurity aspects. Validation of the simulation toolchain with respect to this, involves manual inspection of the KPIs adopted which should include KPIs for cyber-attack effects or mitigation.

6 CONCLUSIONS

Deliverable 4.5 provides a set of validation methodologies based on existing approach described in section 2. These validation methodologies were expanded with more thorough description on the process of validation (sections: 3.2, 3.3, 3.4, 3.5 and 3.6), as well as providing correlation metrics for the actual validation that the measurement data is correlated to the simulation results output of a given simulation model within a simulation toolchain.

D4.5 expands into an area which in general is less explored compared to standard validation of a simulation model. It expands particularly in evaluation of objective and subjective metrics of toolchain functionality as elaborated in sections 3.7, 3.8, and 3.9. The methodologies developed offer a comprehensive framework that enhances traditional validation techniques and ensures a more thorough assessment of virtual simulation models.

By building on existing techniques and addressing less explored validation areas, this work enhances the reliability of virtual simulations by making them more robust and accurate.

D4.4 provided a Harmonised V&V Simulation Framework and different simulation toolchains that were derived from the mentioned framework in ways that depend mostly on SUNRISE use cases as well as contributing partner's experiences.

One of the main challenges encountered in execution of task T4.5, is to manage the development of validation methodologies that are able to cover the simulation toolchains that resulted from T4.4 while mostly running with T4.5 in parallel. The answer to this challenge is the universal approach that provided a blueprint for validation of these simulation toolchains, but it will cover any other simulation toolchain from external industry users as well.

To tailor the contents for SUNRISE project further, D4.5 also provides guidance to use-case specific toolchain validation and provides a universal blueprint that can be utilized for validation of toolchains resulting from the Harmonised V&V Simulation Framework (section 5).

It is important to mention that these guidelines for SUNRISE simulation toolchains are intended for T7.3 in its validation execution efforts. As stated previously, it is **not required for the guideline to be followed strictly**, but it provides a blueprint for validation.

The blueprint is serving the purpose of "Audit" section of the SAF (see

Figure 3), in which validation execution approach in T7.3 can be cross-checked with the recommendations in this deliverable if not possible to be fully implemented.

The methodologies explained in section 3 have been formulated to have primarily a **universal validation approach** which is **tool-independent**, but equally important to **provide blueprint guidance for toolchain validation in T7.3**. This universality is maximizing applicability of validation methodologies to a broad range of stakeholders.

Internal stakeholders, including project participants, will directly benefit from the structured methodologies. Additionally, **external stakeholders**, such as OEMs, Tier 1 suppliers, and other industry participants, can leverage these methodologies to enhance their own validation of simulation toolchains and their respective models. The universal nature of these validation techniques ensures their relevance and usability across different applications in the future as well.

Results from D4.5 (section 5) directly provide guidance for validation of simulation toolchains derived from the Harmonised V&V Simulation Framework from D4.4. It enables the utilization of outlined validation methodologies to assess their specific setups, ensuring consistency and reliability across the project's virtual validation efforts.

Key findings:

- **Expanded industry existing validation approaches** with the **universal** simulation toolchain approach (discussed in the deliverable as "**Validation setup**")
- **Explored objective and subjective metrics** that can be used for evaluation of relevant **toolchain functionalities**
- **Guidance** on **validating the use-case specific toolchains** that were described in D4.4 [1]
- Ensured methodologies **benefit** both **internal and external stakeholders**

By building upon existing techniques and addressing less explored validation areas, the work presented in this deliverable, enhances the reliability of virtual simulations and also provides a blueprint for validation methodologies.

7 REFERENCES

[1] SUNRISE, Deliverable D4.4 - Report on the Harmonised V&V Simulation Framework, 2024.

[2] SUNRISE, Deliverable D7.1 - CCAM Use cases validation requirements, 2024.

[3] SUNRISE, Deliverable D4.1 - Report on relevant subsystems to validate CCAM systems, 2024.

[4] SUNRISE, Deliverable D4.2 - Report on mapping of use case requirements to subsystems, 2024.

[5] T. e. a. Dueser, A Comprehensive Approach for the Validation of Virtual Testing Toolchains, IAMTS, 2021.

[6] I. 11010-1, “Passenger cars - Simulation model classification and taxonomy - Part 1: Vehicle dynamics”.

[7] R. K. L. E. J. K. J. S. a. A. Z. J. Bock, “Data Basis for Scenario-Based Validation of HAD on Highways,” in *27th Aachen Colloquium Automobile and Engine Technology*, Aachen, 2018.

[8] SUNRISE, Deliverable D4.3 - Report on CCAM simulation tool landscape, 2024.

[9] M. P. B. a. M. R. A. Ngo, “A Multi-Layered Approach for Measuring the Simulation-to-Reality Gap of Radar Perception for Autonomous Driving,” in *International Intelligent Transportation Systems Conference (ITCS)*, Indianapolis, 2021.

[10] R. D. a. B. Ciuffo, “Virtual Testing of Automated Driving Systems. A Survey on Validation Methods,” *IEE Access,* vol. 10, 2022.

[11] P. Rosenberger, Metrics for Specification, Validation, and Uncertainty Prediction for Credibility in Simulation of Active Perception Sensor Systems, Darmstadt: PhD thesis, TU, 2022.

[12] P. R. M. H. K. M. J. S. a. S. P. L. Elster, “Introducing the double validation metric for radar sensor models,” *Automotive and Engine Technology,* vol. 9, p. 6, 2024.

[13] W. L. O. a. L. G. S. Ferson, “Model validation and predictive capability for the thermal challenge problem,” *Computer Methods in Applied Mechanics,* vol. 197, 2008.

[14] A. &. A. A. Bowman, “Applied smoothing techniques for data analysis: the kernel approach with S-plus illustrations,” *Journal of the American Statistical Association,* 1999.

[15] J. Jovanovski, Crash Data Analysis and Model Validation Using Correlation Techniques, SAE Technical Paper 810471, 1981.

[16] C. Spearman, The Proof and Measurement of Association between Two Things, America Journal of Psychology, 1904.

[17] “https://www.unrealengine.com,” Unreal Engine. [Online]. [Accessed 10 February 2025].

[18] W. P. on Automated/Autonomous and C. Vehicles, “New Assessment/Test Method for Automated Driving (NATM) Guidelines for Validating Automated Driving System (ADS),” Economic Commission for Europe (ECE), 2023. [Online]. Available: https://unece.org/sites/default/files/2023-04/ECE-TRANS-WP.29-2023-44e.pdf. [Accessed 10 February 2025].

[19] I. A. for Mobility Testing Standardization, “IAMTS Best Practice for A Comprehensive Approach for the Validation of Virtual Testing Toolchains,” in *SAE Industry Technologies Consortia*, 2021.

[20] D. R. G. M. G. W. S. C. T. F. a. V. S. Ciuffo B, “Interpretation of eu regulation 2022/1426 on the type approval of automated driving systems,” no. no. KJ-NA-31-842-EN-N (online), 2024, issn: 1831-9424 (online). doi: 10.2760/86028(online).

[21] HESAI, “What You Need to Know About Lidar: The Strengths and Limitations of Camera, Radar, and Lidar,” HESAI, [Online]. Available: https://www.hesaitech.com/what-you-need-to-know-about-lidar-the-strengths-and-limitations-of-camera-radar-and-lidar/. [Accessed 10 February 2025].

[22] N. Parrott, “LIRDATA,” Medium, 2018. [Online]. Available: https://medium.com/@nathanparrott/improving-lidar-radar-and-other-photon-emissive-sensors-through-data-transmission-utilizing-d0adece334bb. [Accessed 10 February 2025].

[23] “How do ultrasonic sensors work?,” Manorshi, [Online]. Available: https://www.manorshi.com/how-do-ultrasonic-sensors-work.html. [Accessed 10 February 2025].

[24] E. v. Rees, “Toposens’ 3D ultrasonic sensor brings close-range sensing to robotics,” GEO WEEK NEWS, 2020. [Online]. Available: https://www.geoweeknews.com/news/toposens-3d-ultrasonic-sensor-brings-close-range-sensing-to-robotics. [Accessed 10 February 2025].

[25] L. B. P.-D. B. Velagic J., “A 3-level autonomous mobile robot navigation system designed by using reasoning/search approaches,” *Robotics and Autonomous Systems,* 2004.

[26] ISO, Road vehicles - Safety of the intended functionality, ISO 21448:2022.

[27] V. T. a. T. T. A. Widner, “Framework for Vehicle Dynamics Model Valida-tion,” *IEEE Access,* vol. 10, no. doi: 10.1109/ACCESS.2022.3157904, 2022.

[28] P. R. S. S. L. E. R. S. a. H. W. C. Linnhoff, “Towards Serious Perception Sensor Simulation for Safety Validation of Automated Driving - A Collaborative Method to Specify Sensor Models,” in *IEEE International Intelligent Transportation Systems Conference*, Indianapolis, 2021.

[29] P. e. a. Weissensteiner, “Operational design domain-driven coverage for the safety argumentation of automated vehicles,” in *IEEE Access 11*, 2023.

[30] C.-G. J. K. a. I.-J. I. Roh, Analysis of impact of rain conditions on ADAS, 2020.

[31] H. e. a. Li, “The effect of rainfall and illumination on automotive sensors detection performance,” *Sustainability,* no. 15.9, 2023.

[32] “Tesla Dashcam Footage Suggests Reasons for Autopilot Crashes,” Youtube, [Online]. Available: https://www.youtube.com/watch?v=V2u3dcH2VGM&t=32s.. [Accessed 10 February 2025].

[33] K. e. a. Yoneda, “Automated driving recognition technologies for adverse weather conditions,” *IATSS research,* no. 43.4, pp. 253-262, 2019.

[34] F. e. a. Reway, “Test method for measuring the simulation-to-reality gap of camera-based object detection algorithms for autonomous driving,” *IEEE Intelligent Vehicles Symposium (IV),* 2020.

[35] ASAM e.V., ASAM OSI® (Open Simulation Interface) - Official Documentation, ASAM e.V., 2023.

[36] Springer Book in Review with Working Title, “Chapter Validation of Perception Sensor Models,” in *Validation and Verification of Automated Systems*, Springer Book.

[37] AVL List GmbH, “Model.CONNECT,” AVL, [Online]. Available: https://www.avl.com/en/simulation-solutions/software-offering/simulation-tools-a-z/modelconnect. [Accessed 10 February 2025].

[38] AVL List GmbH, Openmcx - cosimulation middleware, Graz: AVL List GmbH.

[39] Modelica Association, Functional Mock-up Interface, Modelica Association, 2022.

[40] AVL List GmbH, ENABLE-S3 - European Initiative to Enable Validation for, AVL List GmbH, 2019.

[41] DLR, PEGASUS - Projekt zur Etablierung von generell akzeptierten Gütekriterien, Werkzeugen und Methoden, Deutsches Zentrum für Luft und Raumfahrt e.V. (DLR), 2019.

[42] DLR, SET Level - Simulationbased Engineering and Testing of Automated Driving, Deutsches Zentrum für Luft und Raumfahrt e.V. (DLR), 2022.

[43] Technische Universität Ilmenau, VIVID - Virtual Validation Tool Chain for Automated and Connected Driving, Technische Universität Ilmenau, 2024.

[44] I. O. a. S. G. F. Hoffmann, “Scenario reconstruction for supporting virtual adas/ad development and testing,” *IEEE Transactions on Intelligent,* 2024.

[45] EUROPEAN NEW CAR ASSESSMENT PROGRAMME (Euro NCAP), “TEST PROTOCOL – AEB Car-to-Car systems, Version 4.0,” [Online]. Available: https://cdn.euroncap.com/media/67887/euro-ncap-aeb-c2c-test-protocol-v40.pdf. [Accessed 10 February 2025].

[46] EUROPEAN NEW CAR ASSESSMENT PROGRAMME (Euro NCAP), “TEST PROTOCOL – AEB/LSS VRU systems, Version 4.3,” 2022. [Online]. Available: https://cdn.euroncap.com/media/75436/euro-ncap-aeb-lss-vru-test-protocol-v43.pdf. [Accessed 10 February 2025].

[47] UN Regulation No. 157, “Automated Lane Keeping Systems,” 2021. [Online]. Available: https://unece.org/sites/default/files/2023-12/R157e.pdf. [Accessed 10 February 2025].

[48] A. G. K. M. S. L. T. P. K. E. Greg Chance, “On Determinism of Game Engines Used for Simulation-Based Autonomous Vehicle Verification,” no. IEEE, 2022.

[49] J. Gustafson, “Reevaluating Amdahl's law,” *In Communications of the ACM,* vol. 31, pp. 532-533, 1988.

[50] G. Amdahl, “Validity of the single-processor approach to achieving large scale computing capabilities,” in *AFIP Conference Proceedings*, 1967.

[51] X. K. S. a. J. P. A. Zhang, “An ODD-based scalable assurance framework for automated driving systems,” in *WCX SAE World Congress Experience*, Detroit, 2023.

[52] ISO, Road Vehicles — Test scenarios for automated driving systems — Specification for operational design domain, ISO 34503:2023.

[53] SUNRISE, Deliverable D7.2 - Report on relevant subsystems to validate CCAM systems, 2024.

ANNEX 1: SAMPLE RESULTS OBTAINED FROM THE CAMERA MODEL VALIDATION APPROACH

Experiments comparing real and virtual camera performance were conducted in a controlled lab environment, replicated virtually[2] in CARLA simulator with matching dimensions and layout. A Luxonis OAK-D Pro W camera (12MP, 4056x3040, 95° FoV) was used for physical tests under controlled LED soft lighting. The camera and the lights were configured in the simulator to match the real setup. Utilizing industry-standard Macbeth Colour chart for colour accuracy and the ISO 12233:2023 e-SFR chart for sharpness evaluation, we captured camera images at varying distances (ranging from 1.0m to 2.5m in 0.5m intervals) from the charts. The comparative assessment, based on ΔE_{00} and MTF50 metrics (as shown in Figure 56) demonstrates that the CARLA camera sensor model replicates color accuracy and sharpness aspects of its real-world counterpart relatively consistently across evaluated distances. Note that lower values of ΔE_{00} indicate higher color fidelity, while higher MTF50 values indicate better sharpness.

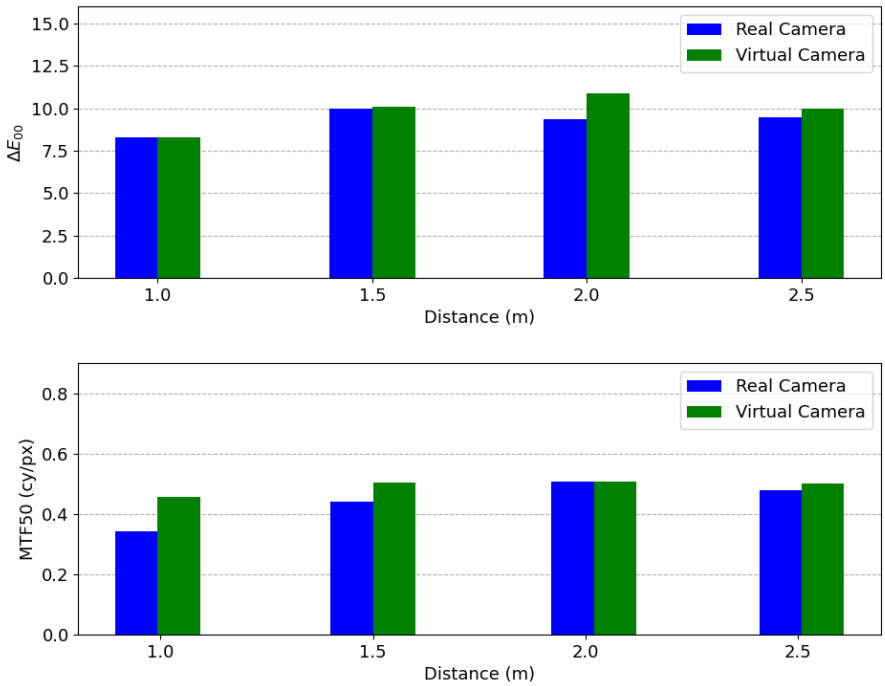


Figure 56. Comparative assessment of virtual camera model through colour accuracy and sharpness metrics

[2] Virtual lab environment is available publicly at https://github.com/RISE-Dependable-Transport-Systems/UE-Camera-Validation/tree/ue4_carla