PRISSMA Project
Plateforme de Recherche et d'Investissement pour la Sûreté
et la Sécurité de la Mobilité Autonome
04/2021 - 04/2024

[L2.3] WP2 STATE OF THE ART : FINAL REPORT

RAPPORT FINAL: ÉTAT DES LIEUX WP2

Main authors : Dominique Gruyer (UGE), Rémi Régnier (LNE), Sio-Song Ieng (UGE), Marc El Zeenny (IRT SystemX), Keilatt Andriantavison (VALEO), Elodie Gasch (AIRBUS Protect), Lydia Habib (AIRBUS Protect) Alessandro Renzaglia (INRIA), Cedric Gava (SPHEREA), Frédérique Bernardin (Cerema), Charlotte Segonne (Cerema), Gilles Mouchard (CEA)

Keywords: Simulation, evaluation, AI systems, validation plan and requirements, validation methods, verification process, evaluation protocol, scenario generation, metrics, KPI, risk assessment, performance indicators, simulation tools, sensor models, traffic simulation, graphical and physic engines, simulation environments

Abstract. This document describes the state of the art on the simulation of autonomous vehicles with regards to homologation. This state of the art involves validation process, state of the existing models and simulation platforms, including the metrics and criteria used to evaluate and validate algorithms, models, systems, and applications.

Résumé. Ce document décrit l'état de l'art sur la simulation des véhicules autonomes en ce qui concerne l'homologation. Cet état de l'art traite du processus de validation, de l'état des modèles et des plateformes de simulation existants, ainsi que des métriques et des critères d'évaluation et de validation des algorithmes, des modèles, des systèmes et des applications.

Contents

List of Figures

List of Tables

1 Introduction

Over the last decade, the rapid development of vehicle information systems, embedded sensor technologies, "smart" processing of data has highlighted a growing need for automotive manufacturers, as well as for research laboratories, to find the means to prototype, test, and evaluate complex embedded systems (ADAS: Advanced Driving Assistance Systems) which can be active (action on actuators) and cooperative (using of communication means: 802.11p, 5G). Since 2014, EuroNCAP has integrated several driver assistance systems during their evaluation process, which are classified under the title "Safety Assist" and weigh 20% of the final rating. Among these systems, AEB (Automatic Emergency Braking) can be mentioned as an example. However, the testing and validation process is either limited to a few use cases (three for AEB) or only applicable to the data provided by the car manufacturers. But it is precisely that road safety and the risk of a situation are directly related to the reliability and robustness of these embedded systems and also the information they retrieved. We have also seen that the sensors required to develop and deploy automated driving systems are increasingly ubiquitous, numerous and complex in recent years. Moreover, the processing stages are becoming more complex, and more and more new approaches and methods of system of systems and AI-based systems are used. In order to evaluate the performance and quality of such AI-based applications and algorithms for perception, decision-making, and control/command layers (included in the design of automation systems for partial, full, and/or shared driving), it is necessary to develop procedures, measurement tools, and ground truths. In view of the diversity of the situations to be tested (considering possible failure, degradation, adverse conditions, and attacks: climate, infrastructure, sensors, communication bus etc.), it is increasingly difficult to perform these tests only on real test tracks (real controlled environment). Consequently, alternative and above all, complementary solutions must be found to be able to consider a large number of representative situations and data. For this, the use of testing and simulation tools and platforms is becoming essential. In addition, it is necessary to interface all these testing and simulation tools to obtain exploitable and, most of all, valid results. Of course, taking the human aspect into account for these tools makes the problem more complex and requires real-time operations, which can be as close as possible to reality. Active and cooperative ADAS evaluation is a significant issue for automated driving applications using AI-based systems with a pre-certification in view.

The challenge is no longer just "technological" certification as such, but also certifying this technology "from the perspective of human use and interaction", which is inherently adaptable and changeable. Currently, there are no generic design processes, no standardized methods, and almost no integrated simulation tools to perform such tests for the design, development and evaluation of CAV using AI-based systems. This is also true for evaluating CAV interaction in a fleet of vehicles fully or partially equipped with ADAS and/or means of communication. One of the PRISSMA project topics is to study this issue and propose simulation platforms, evaluation metrics, and scenario managers to provide a first alternative and effective solution for evaluating and validating CAV using AI-based algorithms, applications, and systems.

The subgoals of such a solution will be to address the following requirements and constraints:

- Decreasing the number of kilometers required to travel in the real world to evaluate and to validate an AI-based system, or a "system of systems", in a large and rich set of critical scenarios, which can involve infrastructure (roadside) degradation, climatic conditions, sensors, and algorithms;
- Ensuring the repeatability and reproducibility of driving conditions and equipment;

- Ensuring the data quality and the physico-realistic modeling of the information sources;
- Measuring the performance of AI-based systems for CAV applications and services with a reliable and precise generation of “ground truths”;
- Integrating human and cognitive engineering aspects, including monitoring considerations, Human-Machine Cooperation, and simulation practices on how end drivers could use, interact, and accept IA-based systems for automated driving.

Figure 1 shows the complexity of the data to assess, manage, and potentially consider in the validation process.

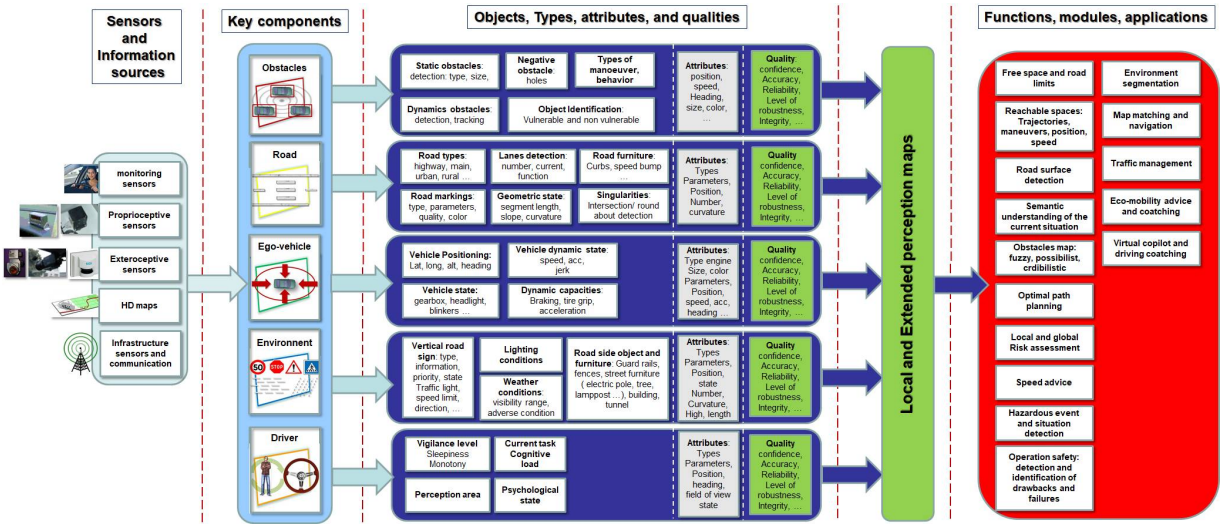

Figure 1: Perception layers, attributes, and objects to consider to feed functions, modules, and applications involved in the design of AD systems with both AI-based methods and system of systems applications (source: Univ Eiffel)

Integrating AI-based systems in sensor management, perception layers, decision-making, and control systems brings issues and V&V Challenges. From the previous study, some researches are highlighted the following challenges:”

- **Lack of an ”oracle” or a reference**: It is difficult or impossible to clearly define the correctness criteria for system outputs or the right outputs for each individual input.
- **Imperfection**: It is intrinsically impossible for an AI system to be 100% accurate because it depends of what it had used in the training stage.
- **Uncertain behaviour for untested data**: There is high uncertainty about how the system will react with untested input data, as evidenced by radical behaviour changes given slight input changes (e.g., adversarial examples). Some work tries to propose a solution to manage uncertainties and quantify the level of uncertainty of a specific output.
- **High dependency of behavior on training data**: System behavior is highly dependent on the training data and the quality/representativeness of the training dataset.

These remarks and comments are characteristic of AI itself and can be generalized as follows:

- Erosion of determinism
- Unpredictability and unexplainability of individual outputs
- Unanticipated, emergent behavior, and unintended consequences of algorithms
- Complex decision making of the algorithms
- Difficulty of maintaining consistency and weakness against slight changes in inputs (Goodfellow et al., 2015 [50])

Some types of data are costly to collect, or they are rare. For instance, collecting data representing the variety of real-world road events for an autonomous vehicle may be prohibitively expensive. Generating synthetic data that reflects the important statistical properties of the underlying real-world data can solve these problems. It is inexpensive compared to collecting large datasets and can support AI/deep learning model development or software testing without compromising customer privacy. It's estimated that by 2024, 60% of the data used to develop AI and analytics projects will be synthetically generated. If we compare efficiency of synthetic data comparatively to real data performance, a recent experiment based on Machine learning (one of the most common use cases for data today) and made by MIT scientists has highlighted an interesting result. MIT researchers wanted to measure if machine learning models from synthetic data could perform as well as models built from real data. In this study, they split data scientists into two groups: one using synthetic data and another using real data. 70% of the time group using synthetic data was able to produce results on par with the group using real data.

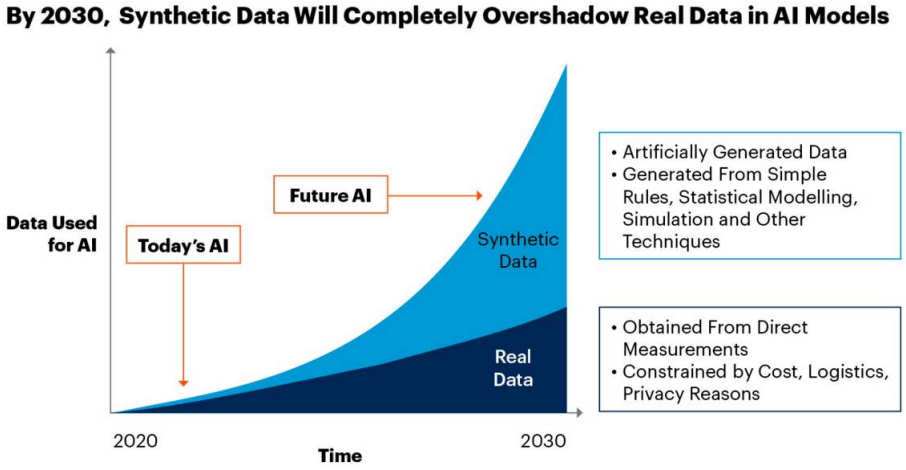

Figure 2: Synthetic data vs real data: An strong increase of Using of synthetic data for AI-based system, (Source: Gartner)

Benefits of synthetic data are mainly due to the capability to generate data that mimics the real thing may be seem like a limitless way to create critical and hazardous scenarios for testing and development. While there is much truth to this, it is important to remember that any synthetic models deriving from data can only replicate specific properties of the data, meaning that they'll ultimately only be able to simulate general trends. However, synthetic data has several benefits over real data:

- Overcoming real data usage restrictions: Real data may have usage constraints due to privacy rules or other regulations. Synthetic data can replicate all important statistical properties of real data without exposing real data, thereby eliminating the issue.

- Creating data to simulate not yet encountered conditions: Where real data does not exist, synthetic data is the only solution.
- Immunity to some common statistical problems: These can include item non-response, skip patterns, and other logical constraints.
- Focuses on relationships: Synthetic data aims to preserve the multivariate relationships between variables instead of specific statistics alone.

These benefits demonstrate that the creation and usage of synthetic data and dedicated tools and models will only stand to grow as the data becomes more complex and more closely guarded.

When determining the best method for creating synthetic data, it is important to first consider what type of synthetic data you aim to have. There are two broad categories to choose from, each with different benefits and drawbacks:

- Fully synthetic: This data does not contain any original data. This means that re-identification of any single unit is almost impossible and all variables are still fully available.
- Partially synthetic: Only data that is sensitive is replaced with synthetic data. This requires a heavy dependency on the imputation model. This leads to decreased model dependence, but does mean that some disclosure is possible owing to the true values that remain within the dataset.

Three general strategies for building synthetic data include:

- Drawing numbers from a distribution: This method works by observing real statistical distributions and reproducing fake data. This can also include the creation of generative models.
- Agent-based modeling: To achieve synthetic data in this method, a model is created that explains an observed behavior, and then reproduces random data using the same model. It emphasizes understanding the effects of interactions between agents on a system as a whole.
- Learning methods (NN, SVM, Machine Learning, Deep learning models): Variational auto-encoder and generative adversarial network (GAN) models are synthetic data generation techniques that improve data utility by feeding models with more data.

Some challenges have to be addressed in the generation of Synthetic Data for AI-based systems evaluation and validation. Though synthetic data has various benefits but also has limitations:

- Outliers may be missing: Synthetic data can only mimic the real-world data (from statistic and/or physical models), it is not an exact replica of it. Therefore, synthetic data may not cover entirely and accurately outliers that original data has. However, outliers in the data can be more important than regular data points.
- Quality of the model depends on the data source: The quality of synthetic data is highly correlated with the quality of the input data and the data generation model. Synthetic data may reflect the biases in source data.
- User acceptance is more challenging: Synthetic data is an emerging concept and it may not be accepted as valid by users who have not witnessed its benefits before.

- Synthetic data generation requires time and effort: Though easier to create than actual data, synthetic data is also not free.

- Output control is necessary: Especially in complex data-sets, the best way in order to guarantee the output accuracy is by comparing synthetic data with actual data or human-annotated data. this is because there could be inconsistencies in synthetic data when trying to replicate complexities within original data-sets.

1.1 Verification and Validation of AI Systems

Challenges considering AI requirements are extensive and due in part to the practice by some to treat the AI element as a "black box" (Gunning 2016 [51]). Formal specification has been attempted and has shown to be difficult for those hard-to-formalize tasks and requires decisions on the use of quantitative or Boolean specifications and the use of data and formal requirements. The challenge is to design effective methods to specify both desired and undesired properties of systems that use AI-based components. In Seshia 2020 [52], considers Verified AI from a formal methods perspective. He describes five challenges for achieving Verified AI, and five corresponding principles for addressing these challenges. A taxonomy of AI requirements engineering challenges is proposed by Belani 2019 [2] and shown in figure 3. Moreover, [53] addresses Explainable Artificial Intelligence (XAI) with the Concepts, the taxonomies, the opportunities and the challenges toward responsible AI. In this paper, the most commonly used nomenclature is given to clarify the distinction and similarities among terms often used in the ethical AI and XAI communities:

- **Understandability (equivalent to intelligibility)**: it denotes the characteristic of a model to make a human understand its function –how the model works –without any need for explaining its internal structure or the algorithmic means by which the model processes data internally

- **Comprehensibility** : When a system is designed and built for Machine Learning models, comprehensibility refers to the ability of a learning algorithm to represent its learned knowledge in a human understandable fashion. This notion of model comprehensibility stems from the postulates which stated that "the results of computer induction should be symbolic descriptions of given entities, semantically and structurally similar to those a human expert might produce observing the same entities. Components of these descriptions should be comprehensible as single 'chunks' of information, directly interpretable in natural language, and should relate quantitative and qualitative concepts in an integrated fashion ". Given its difficult quantification, comprehensibility is normally tied to the evaluation of the model complexity.

- **Interpretability** : It is defined as the ability to explain or to provide the meaning in understandable terms to a human.

- **Explainability** : Explainability is associated with the notion of explanation as an interface between humans and a decision-maker that is, at the same time, both an accurate proxy of the decision-maker and comprehensible to humans

- **Transparency** : A model is considered to be transparent if by itself it is understandable. Since a model can feature different degrees of understandability, transparent models are

divided into three categories: simulatable, decomposable and algorithmically transparent models.

Requirements engineering for AI	AI related entities		
Activities	**Data**	**Model**	**System**
Elicitation	Availability of large datasets Requirements analyst upgrade	Lack of domain knowledge Undeclared consumers	How to define problem /scope Regulation (e.g., ethics) not clear
Analysis	Imbalanced datasets, silos Role: data scientist needed	No trivial workflows Automation tools needed	No integration of end results Role: business analyst upgrade
Specification	Data labelling is costly, needed Role: data engineer needed	No end-to-end pipeline support Minimum viable model useful	Avoid design anti- patterns Cognitive / system architect needed
Validation	Training data critical analysis Data dependencies	Entanglement, CACE (Change Anything, Change Everything) problem High scalability issues for ML	Debugging, interpretability Hidden feedback loops
Management	Experiment management No GORE-like (Goal-Oriented Requirements Engineering) method polished	Difficult to log and reproduce DevOps role for AI needed	IT resource limitations, costs Measuring performance
Documentation	Data & model visualization Role: research scientist useful	Datasets and model versions Education and training of staff	Feedback from end-users Development method
All of the abobe	Data privacy and data safety		

Figure 3: Requirements engineering for AI taxonomy, mapping challenges to AI-related entities and requirements engineering activities([2])

1.1.1 Data in AI

Data is the life-blood of AI capabilities, given that it is used to train and evaluate AI models and produce their capabilities. Data quality attributes of importance to AI include accuracy, currency and timeliness, correctness, consistency, usability, security and privacy, accessibility, accountability, scalability, lack of bias and others. The correctness of unsupervised methods is embedded in the training data and the environment.

There is a question of coverage of the operational space by the training data. If the data does not adequately cover the operational space, the behaviour of the AI component is questionable. However, there are no firm guarantees on when a data set is 'large enough'. In addition, 'large' is not sufficient. The data must sufficiently cover the operational space. A logical solution is to use simulation tools to generate the most considerable possible relevant and diverse data.

Another challenge with data is that of adversarial inputs. Szegedy et al. (2013) [54] discovered that several Machine Learning models are vulnerable to adversarial examples. This has been shown many times on image classification software, however, adversarial attacks can be made against other AI tasks (e.g., natural language processing) and against techniques other than neural networks (typically used in image classification) such as reinforcement learning (e.g., reward hacking) models. In simulation platforms and tools, it is essential to have the capability to generate all these adverse and degraded data and conditions.

1.1.2 Model in AI

Numerous Verification and Validation challenges arise in the model space, some of which are provided below.

- **Modeling the environment**: Unknown variables, determining the correct fidelity to model, modeling human behavior. The challenge is providing a systematic method of

environment modeling that allows one to provide provable guarantees on the system's behavior even when there is considerable uncertainty about the environment. (Seshia 2020) [52]

- **Modeling learning systems**: In this case, the challenges are mainly dedicated to the very high dimensional input space, the very high dimensional parameter or state space, the online adaptation/evolution of the AI-based system, and the modeling context (Seshia 2020).
- **Design and verification of models and data**: data generation, quantitative verification, compositional reasoning, and compositional specification (Seshia 2020). The challenge is to develop techniques for compositional reasoning that do not rely on complete compositional specifications. Optimization strategy must balance between over- and under-specification. One approach, instead of using distance (between predicted and actual results) measures, uses the cost of an erroneous result (e.g., an incorrect classification) as a criterion (Faria, 2018) [55].
- **Online learning**: requires monitoring; need to ensure its exploration does not result in unsafe states.
- **Formal methods**: intractable state space explosion from the complexity of the software and the system's interaction with its environment, an issue with formal specifications. Bias in algorithms from underrepresented, incomplete training data, or reliance on flawed information that reflects historical inequities. A biased algorithm may lead to decisions with a disparate collective impact. Trade-off between fairness and accuracy in the mitigation of an algorithm's bias.
- **Test coverage**: effective metrics for test coverage of AI components is an active area of research with several candidate metrics, but currently no clear best practice.

1.1.3 Properties in AI

Assurance of several AI system properties is necessary to enable trust in the system, e.g., the system's trustworthiness. This is a separate though necessary aspect of system dependability for AI systems. Some important properties are listed below and though extensive, are not comprehensive.

- **Accountability**: refers to the need for an AI system to be answerable for its decisions, actions and performance to users and others with whom the AI system interacts.
- **Controllability**: refers to the ability of a human or other external agent to intervene in the AI system's functioning (A recent report is available on this topic: [56])
- **Explainability**: refers to the property of an AI system to express important factors influencing the AI system results or to provide details/reasons behind its functioning so that humans can understand. ([53])
- **Interpretability**: refers to the degree to which a human can understand the cause of a decision. (Miller 2017) [57]
- **Reliability**: refers to the property of consistent intended behavior and results.

- **Resilience**: refers to the ability of a system to recover operations quickly following an incident.
- **Robustness**: refers to the ability of a system to maintain its level of performance when errors occur during execution and to maintain that level of performance given erroneous inputs and parameters.
- **Safety**: refers to the freedom from unacceptable risk.
- **Transparency**: refers to the need to describe, inspect and reproduce the mechanisms through which AI systems make decisions, communicating this to relevant stakeholders.

1.1.4 Recent Verification and Validation approaches dedicated to AI-based systems

More and more, AI-based systems are used in all parts involved in Automated Driver system. Specifically, Machine Learning, CNN, and Deep Learning are strongly developed and used. In parallel to these developments, researches on Verification and Validation of AI-based training systems involved in the adaptation of available standards, such as the then-current IEEE Std 1012 (Software Verification and Validation) processes (Pullum et al. 2007), areas need to be augmented to enable Verification and Validation (Taylor 2006), and examples of V&V for high-assurance systems with AI-based methods (Schumann et al., 2010). While these books (mainly for Neural Networks) provide techniques and lessons learned, many of which remain relevant, additional challenges due to deep learning remain unsolved.

One of the challenges is yet clearly the data validation. It is essential and critical that the data upon which AI depends undergo Verification and Verification. Data quality attributes vital for AI systems include accuracy, currency and timeliness, correctness, consistency, usability, security and privacy, accessibility, accountability, scalability, lack of bias, and state-space coverage. Data validation steps can include file validation, import validation, domain validation, transformation validation, aggregation rule and business validation (Gao et al. 2011) [4]. This aspect is vital for the PRISSMA project because of the need to generate a massive scale of data to validate AI-based systems. In this condition, it is necessary to quantify the quality of the datasets and the generated data with the different stages of data processing, modification, propagation, transformation. Figure 5 presents the different data layers and factors which can impact data and Big Data Quality. The figure 6 shows a set of Big Data validation tools and major companies addressing these topics.

Several approaches to verify and validate AI components include formal methods (e.g., formal proofs, model checking, probabilistic verification), software testing, simulation-based testing, and experiments. Some specific approaches are:

- **Metamorphic testing**: Metamorphic testing consists of the test of Machine Learning and other AI-based algorithms addressing the oracle problem (Xie et al., 2011)[58]. (Breck et al., 2016) [59] addresses in his paper the issues of Machine Learning test score calculation. He proposed a set of test categories involving tests for features and data, model development and Machine Learning infrastructure, and monitoring tests for Machine Learning.
- **Checking for inconsistency**: this type of test addresses verifying inconsistency with desired behavior and systematically searching for worst-case outcomes while testing consistency with specifications.

- **Corroborative verification**: Several verification methods, working at different levels of abstraction and applied to the same AI component, are applied and may prove useful to the verification of AI components of systems. (Webster et al., 2020) [3] (see figure 4)
- **Testing against strong adversarial attacks**: Some recent works on AI-based approaches have shown that used models and methods may be robust to weak adversarial attacks but show little accuracy to strong attacks ([60]).
- **Use of formal verification** to prove that models are consistent with specifications, e.g., (Huang et al., 2017) [61].
- **Assurance cases** combining the results of Verification and Validation and other activities as evidence to support claims on the assurance of systems with AI components (Picardi et al. 2020) [62].

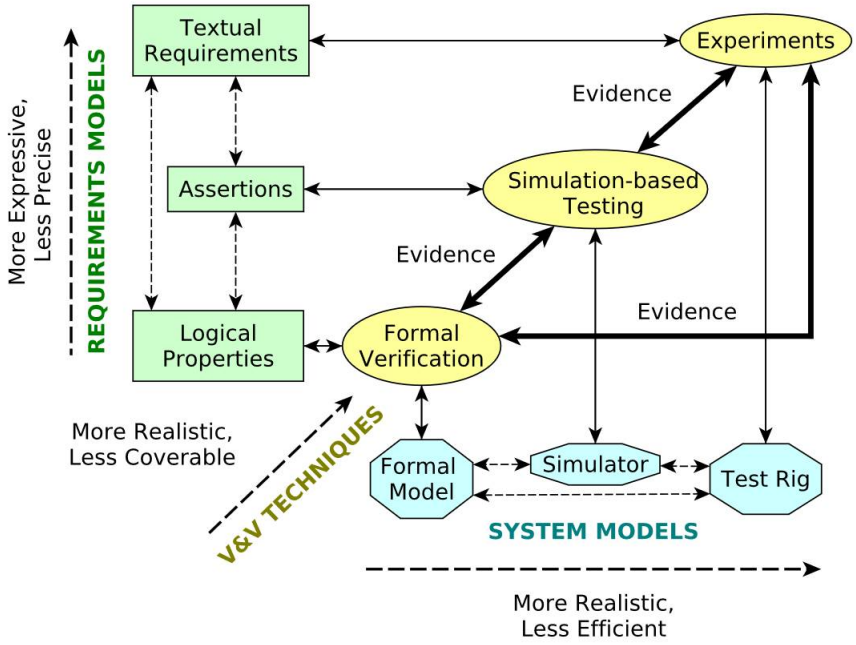

Figure 4: A framework for corroborative Verification and Validation of systems ([3])

This deliverable has for main objective to provide a first global overview and a state of the art on this topic concerning Evaluation, Verification, Validation, and Certification of AI-based systems for automated driving but with simulation methods, process, procedures, tools. This state of the art has to address this topics with all the concepts and challenges enumerated below.

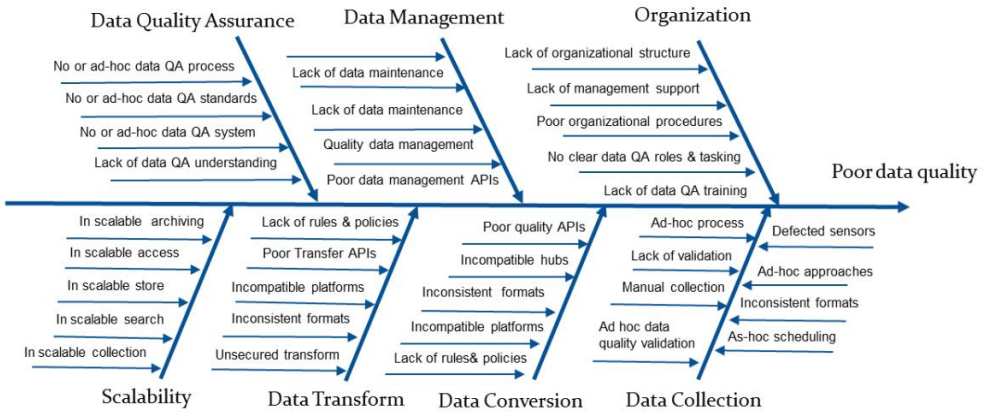

Figure 5: Factors Causing Data and Big Data Quality Problems ([4])

Tools / Features			Datameer (www.datameer.com)	Talend Open Studio (http://www.talend.com/)	Informatica (www.informatica.com)	IBM QuerySurge (www.querysurge.com	C3 Intergrity (www.c3integrity.com)	Microsoft Azure HDInsight (www.microsofty.com)	SAP HANA (www.sap.com)	Jumbune (www.jumbune.org)
Operation Environment	OS	Windows	●	●	●	●	●	●	●	○
		Linux	●	●	●	●	●	●	●	●
		MAC OSX	○	●	●	●	○	○	○	○
		Debian	●	○	●	○	○	○	○	○
		Centos	●	○	●	○	○	○	○	○
		Ubuntu	●	○	●	○	○	○	○	○
		Solaris	●	○	●	○	○	○	○	○
	Basic Features	VMware	○	○	●	●	○	○	○	○
		Java Environment	●	●	○	○	○	●	●	●
		Intuitive user interface	●	●	●	●	●	●	●	●
		Open source	○	○	○	○	○	○	○	●
		ETL(Extract-Transform-Load)	○	●	●	●	●	○	○	○
Supported Data Source	File Formats	CSV/TSV	●	●	●	●	●	●	●	●
		TXT Files	●	●	●	●	●	●	●	●
		Web Server Logs	●	●	●	●	●	●	●	●
		NoSQL	●	●	●	●	●	●	○	●
		Twitter Firehose	●	●	●	●	○	●	●	○
		Fackbook Graph API(Files)	●	●	●	●	○	●	●	●
		Fixed Width Text	●	●	●	●	●	●	●	●
		HTML	●	●	●	●	●	●	●	●
		JSON	●	●	●	○	○	●	●	○
		XML	●	●	●	●	●	●	○	●
		Log4jLog File	●	●	●	●	●	●	●	●
	Database	Hive	●	●	●	●	●	●	●	○
		Hbase	●	●	●	●	○	●	●	○
		MySQL	●	●	●	●	●	●	●	○
		DB2	●	●	●	●	●	●	●	○
		Oracle	●	●	●	●	●	●	●	○
		PostgreSQL	●	●	●	●	●	●	●	○
		Vertica	●	●	●	●	●	●	●	○
		Teradata	●	●	●	●	●	●	●	○
		Sybase	●	●	●	●	●	●	●	○
		Excel	●	●	●	●	●	●	●	○
		Azure Blob Storage	●	○	●	●	○	●	●	○
		Amazon Redshift	●	○	●	●	○	●	●	○
Basic Data Validation Functions	Null Data Value		●	●	●	●	●	●	●	●
	Regex(Regular Expressions)		●	●	●	●	●	●	●	●
	Data Type Check		●	●	●	●	●	●	●	●
	Data Range Validation		●	●	●	●	●	●	●	●
	Data Constraint Validation		●	●	●	●	●	●	●	●
Success Application	Education		○	●	●	○	○	●	○	○
	Finance&Insurance		●	●	●	●	●	●	●	○
	Healthcare		○	●	●	●	●	●	●	○
	Manufacturing & Retail		●	●	●	●	●	●	●	○
	Media & Entertainment		●	●	●	●	○	●	●	○
	Public Sector		●	●	●	●	○	●	●	○
	Services		●	●	○	●	○	●	●	○
	Telecommunications		●	●	●	●	○	●	○	○

Figure 6: A Comparison of the Big Data validation tools and main companies addressing this topic. black dot represents supported features and white dot represents not supported features ([4])

1.2 Cycle of design

Virtual testing, evaluation, validation, and certification enter a specific design plan adapted from the V-cycle (see figure 7). A proposal has been made in University Gustave Eiffel to design a virtual co-pilot in a set of previous projects (FP7 Have-it, H2020 eFUTURE, ANR ABV). This new design cycle is called W-cycle, where simulation tools are applied at the first stage. This W-Cycle is presented in the figure 8.

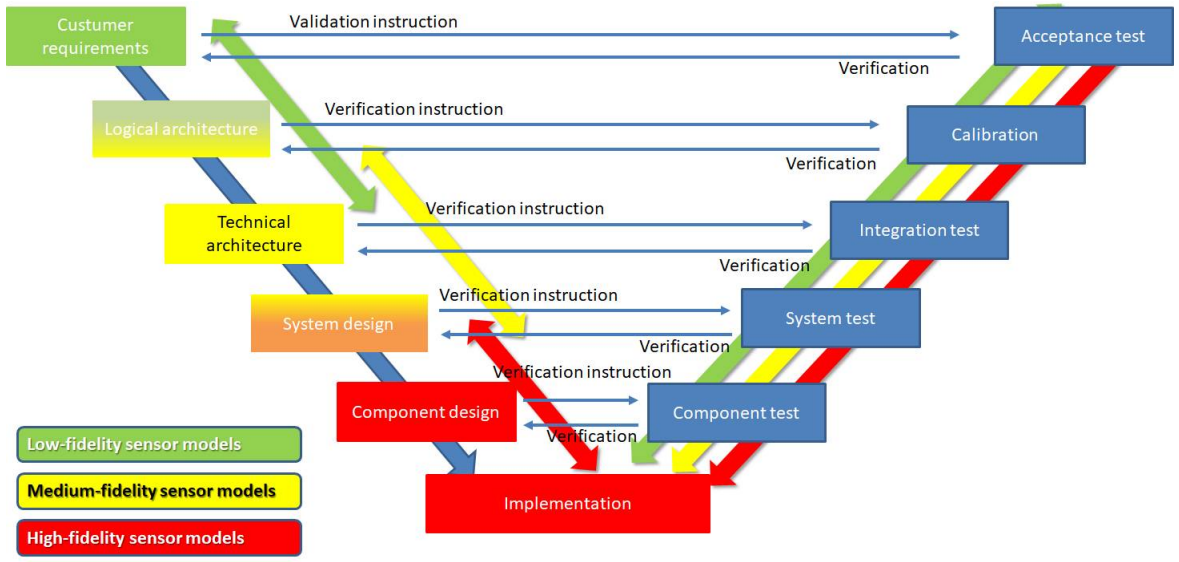

Figure 7: V-cycle for virtual prototyping, test, evaluation, and validation([5])

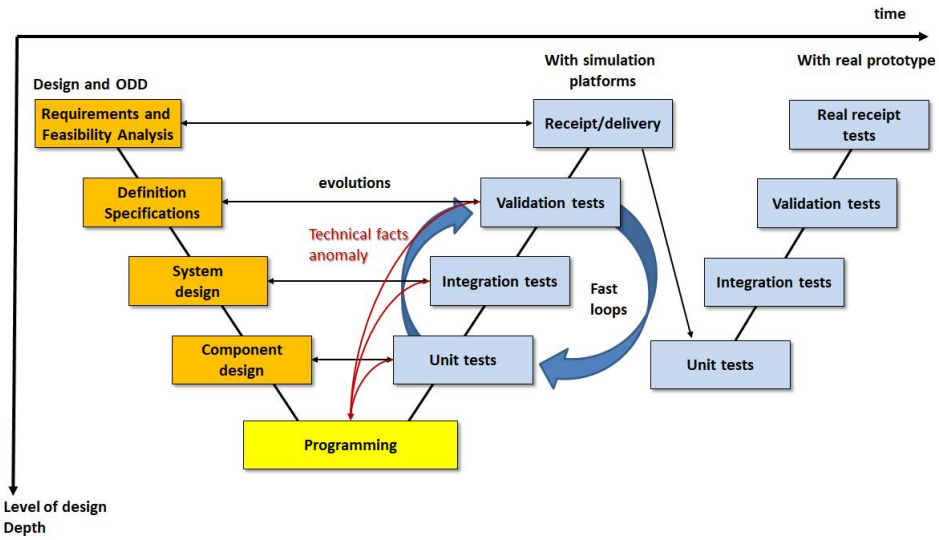

Figure 8: Virtual testing, evaluation, validation, and certification in W-cycle (source: Univ Eiffel)

1.3 Validation plans

Virtual Testing is introduced to reduce the burden of physical tests and effectively provide evidence on the AI performance across the operational domain. Each virtual testing tool will

have its strengths and weakness based on the speed and cost of execution and the level of fidelity achieved. Typically lower fidelity tools are used to cover a vast number of scenarios to obtain a general understanding of the system's performance. Then it is possible to increase the level of fidelity within a subset of scenarios to validate the performance of the AI in a statistically relevant number of realistic scenarios.

- **Model-In-the-Loop:** Model-In-the-Loop is a case where you are using a model of the control to work with a model of the car. The model of the control is probably in Simulink and is connected directly to a physical model of the system within the same Simulink diagram. Rapid development occurs at this stage as you can make small changes to the control model and immediately test the system.

- **Software-In-the-Loop:** This is a case where the control model is slightly more "real" in the sense that you are no longer executing the model but rather have probably coded the model into C or C++ and then inserted this coded model back into your overall plan simulation. This is essentially a test of your coding system (whether autocoded or human coded). Design iteration slows down slightly from MIL, but coding failures start to become evident.

- **Driver-In-the-Loop:** DIL virtual testing can be helpful to support the assessment of this category of functional requirement by analysing the interaction between the driver and the ADS in a safe and controlled environment.

- **Hardware-In-the-Loop:** HIL is mainly used to integrate a hardware component or system to be evaluated and validated in the validation environment. For instance, such an approach have been proposed in Univ. Eiffel to validate hardware box involving perception and fusion layers for ADS.

- **Vehicle-In-the-Loop:** VIL provides a validation environment for ready-to-drive vehicles in combination with a virtual environment simulation. It allows the execution of complex and safety-critical scenarios on the vehicle level.

  VIL on proving grounds focuses more on the interaction between the driver/passenger and the vehicle with virtual data (environment, sensors, perception, etc.). In this configuration, the vehicle's real acceleration (longitudinal and lateral) can be experienced by the driver/passenger (difference to Vehicle-in-the-Loop at testbeds). A judgment and rating by the real driver are possible. Some existing ViL platforms are presented in section 5.

  VIL on proving ground may consist of the following elements:

  - **Longitudinal dynamics**: The real longitudinal dynamics are available and interconnected with virtual environment.
  - **Lateral dynamics**: The real lateral dynamics are available and interconnected with virtual environment.
  - **Perception outputs**: replace and/or enhance perception outputs in order to feed the embedded decision-making and path planning layers.
  - **Sensor data**: replace real sensor outputs by simulated ones.
  - **Communication means**: replace and/or merge real messages with virtual messages

– **embedded virtual reality**: provide an immersive rendering for a real driver in a real vehicle.

Interface virtual environment simulation: Typically, the interface between the vehicle and the virtual environment is done via object list injection. Also, raw data injection is possible. Real sensors cannot be considered (with a few exceptions for very simple sensors like ultrasonic).

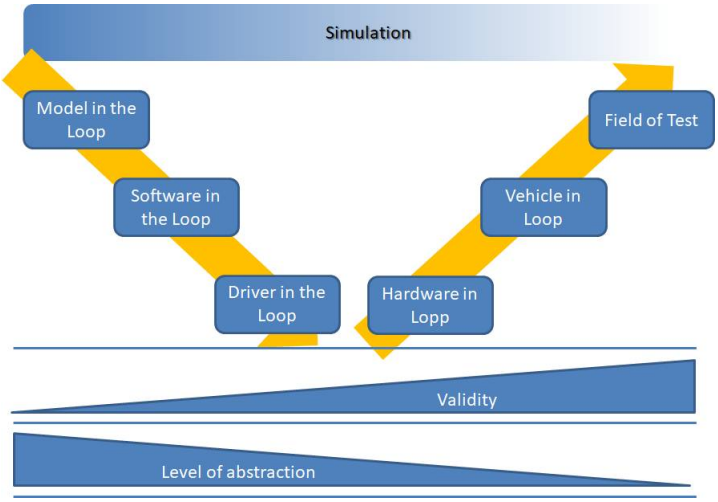

Figure 9: Validation methods in the V-model. MiL, Model-in the-Loop; SiL, Software-in-the-Loop; DiL, Driver-in-the-Loop; HiL, Hardware-in-the-Loop; ViL, Vehicle-in-the-Loop; FOT, Field Operational Test.

1.4 Validation/certification scenarios and procedures

The V-Model is the reference to present the design life cycle of a product such as an ADAS or an ADS as shown in Figure 1 ([63]). The validation stream is always related to the specification stream. It means that validation plans are designed concerning the specifications. However, specifying and validating complex systems of systems such as a CAV is a challenging process. To operate validation plans showing a suitable level of safety and reliability with an acceptable time and budget, virtual method tests from MIL to VIL (see paragraph 1.1) now supplement physical testing: closed site tests and open road tests. The validation phases go from the component tests to the functional test of the full system in its ODD. At the end of a CAV or an ADAS validation process, homologation and certification usually rely on physical tests. However, simulation results are cited in the list of elements that can contribute to the safety demonstration for the authorisation of a Highly Automated Vehicle to be operated on its ODD (French Decree n° 2021-873 du 29 June 2021, Art. R. 3152-6.-I ([64])).

As explained in PFA 2020 ([63]): “Driving scenarios are central in autonomous system validation approaches. They should enable an appropriate screening of events and failures, and their combination, that the vehicle might encounter in its driving environment during its driving policy in its ODD. Scenario management has a vital role in reducing the gap between the potentially infinite combination of driving conditions, events and responses (both from the ego vehicle and alter road users). It can also address a limited number of scenarios employing validation tools (either simulations or tests). Managing scenarios supports the identification of both the most likely and most critical scenarios. It helps reduce the probability of un-identified critical situations (“black-swans”).”

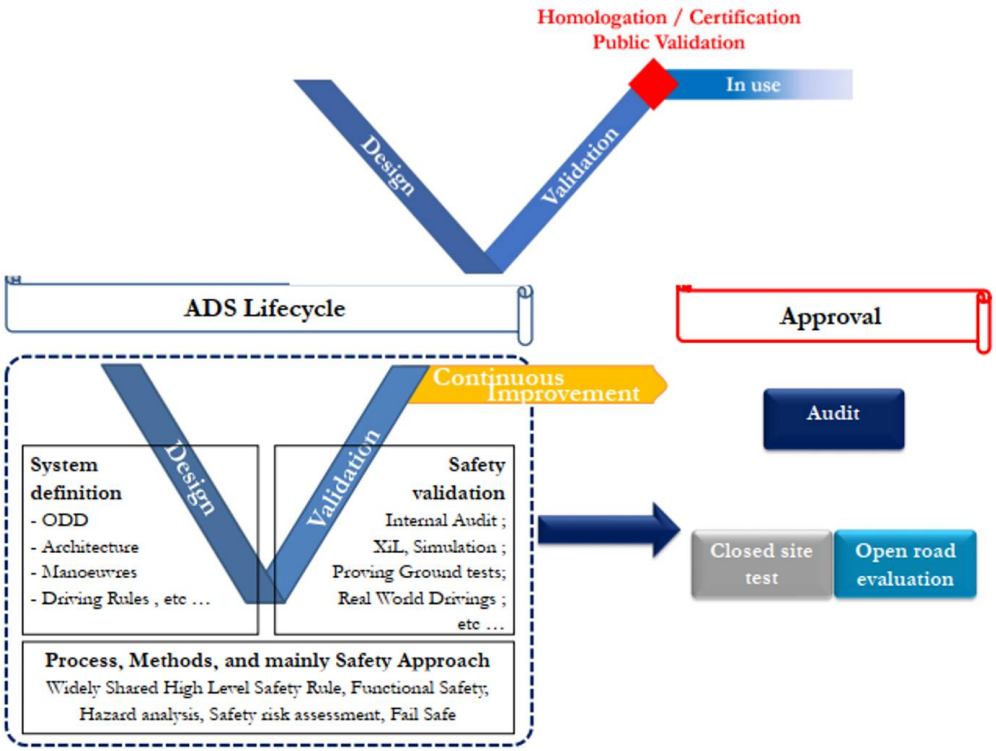

Figure 10: Overall validation architecture from PFA 2020 including design, validation, and approval

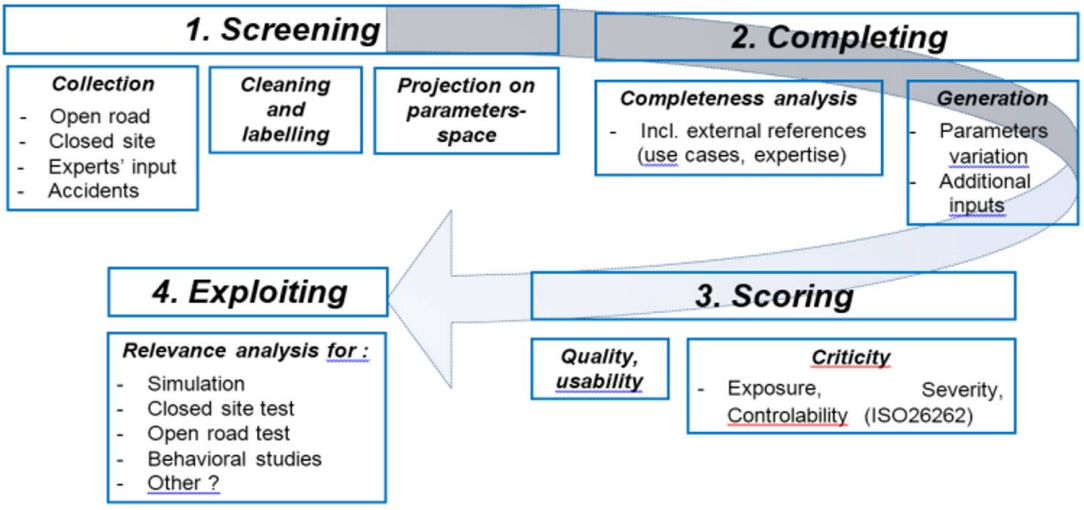

Figure 11: Main process steps for managing driving scenarios for validation from PFA 2020

Many programs are building databases for ADS validation such as PEGASUS (Germany), SAKURA (Japan) (See [65]), MOSAR from IRT System X (see next paragraph). The scenario approach is at the heart of the PRISSMA project as IAs require a large amount of data for their learning and validation. Large scenario databases built with relevant and rich situations might be the key for PRISSMA to show efficient validation methods for IAs in CAV up to the certification.

In [66], a taxonomy of validation strategies to ensure the safe operation of highly automated vehicles is given involving virtual aspects and the different types of validation architecture. In this study, the authors systematically review literature that proposes new methods in this specific area. The available methods were categorized into a proposal of taxonomy, dividing them into the strategies of combinatorial testing, robustness testing and search-based testing. They analyzed the literature regarding modeling capabilities, targeted automation subsystem, targeted driving task level and the metrics used for criticality evaluation and coverage of the scenario space. In this paper, the terminology is given, where the following essential domains are highlighted:

- **Scenario, situation, scene**: A commonly used classification for the terms scenario, situation and scene has been elaborated by Geyer et al. (2014) [67] and was improved by Ulbrich et al. (2015) [6]: A scene is therein defined as a snapshot of the spatial layout of the vehicle and its environment, without any temporal attributes. Furthermore, a situation is derived from a scene to include mission-specific goals and values. It is therefore always from the ego vehicle's point of view. Finally, the scenario includes the temporal aspect by combining several scenes to a progression. In [68], an ontology is proposed based Scene Creation for the Development of Automated Vehicles. This part is addressed with more detail in the next section.

- **Edge, boundary or corner case scenario**: When scenarios are used in the testing domain, further differentiation is used to separate different types of scenarios. Current literature uses various terms to describe test scenarios that can be safety-critical for an AV, such as edge, boundary or corner case scenarios.

- **Rare event**: Another term often used in research regarding scenario-based validation is the term rare event. A rare event is defined by a very low probability of occurrence, between 10^{-9} and 10^{-12}: The probability threshold to classify an event as rare varies in function of the application domain.

2 Modelling, management, and scenario generation

The transport sector is experiencing an unprecedented boom following technological developments which have brought new functions of delegation of driving with the prospect of putting on our roads, in a few years, vehicles more and more autonomous and more and more communicative. This development must not bring new risks and must therefore bring a high level of security and confidence among users.

Reflections on the future of automated driving safety homologation/certification have been very active recently, based on the general consensus that existing validation approaches have to be significantly modified. This implies that the validation approach of such systems should take into account the following main considerations:

- **Scenarios management** : the main challenge of automated driving validation and homologation/certification (compared to traditional automobile validation), is to manage not only system failures but to be able to manage driving hazards (previously handled by the driver) in risk analysis. Industry has undertaken this new item in standardization, under the safety of intended functionalities (SOTIF [69]) framework. Risk analysis, dependent of the ADS, is at the core of validation. Relevent scenarios to design & validate ADS should be managed (because of their huge number) to comply SOTIF and identify the residual risk. Scenario management should be the frame for validation architecture and the main window through which public authorities can scrutinize industry validation processes. SOTIF also requested an "acceptance criteria" which is a validation stop quantitative criteria, to ensure that the introduction of a highly automated vehicle on highways will not increase the risk level. It also ensures that the safety demonstration and its validation have a sufficient coverage.

- **In use system monitoring and validation improvement** : another key challenge for safety, validation and homologation/certification is to collect and take into account new rare scenarios, in order to improve systems' safety and validation capabilities.

In the following sections we introduce the scenario taxonomy and the related definitions shared between the various actors in the field of automotive industry. Then, we present the platform (Methods and Tools for the Safety Assessment and Robustness Analysis of Autonomous Vehicles) developed by IRT-SystemX in the context of former SVA[1] project with automotive actors. The third section of this chapter describes the need of using realistic scenarios and high definition scenery in order to achieve the validation process. Finally, we suggest a first vision on how to identify critical scenarios from field data and SOTIF methodology.

2.1 Simulation Standards

As mentioned in the other sections of this document, it's essential to apply some standards to enable interoperability with a variety of tools used in their overall autonomous vehicle development process. ASAM OpenX standards consist of **OpenDRIVE**, which defines a file format for the description of road networks (i.e. maps) and **OpenSCENARIO**, which defines a file format for the description of the dynamic content in simulation (i.e. driving maneuvers). These standards bring a number of benefits for developers, including but not limited to:

[1] https://www.irt-systemx.fr/en/simulation-for-the-safety-of-autonomous-vehicles-systemx-launches-the-sva-project/

- Create scenarios needed to validate specific safety requirements using any simulation platforms (in-house or third-party)
- Support compliance with SOTIF and ISO 26262 by discovering or creating scenarios that are going to identify hazards or excessive risks in events of component failures
- Scenarios can be shared across platforms to facilitate collaboration among teams inside a company, as well as between companies (e.g., OEMs and suppliers)
- Regulators understand how OEMs test their autonomous systems and can better assess commercial readiness

2.2 Scenario definition

2.2.1 Taxonomy and ontology of ODD, OEDR, OD for scene and scenario definition

2.2.2 Scene and Scenario

In a first approach, a test scenario, in the context of validation engineering or numerical simulation, is accepted as the highest level of abstraction of the test process. In fact, in the context of PRISSMA, it was first deemed relevant to speak of a scenario to describe a driving situation isolated from any system to be tested.

When a vehicle involved in this scenario is then named ”ego vehicle”, the autonomous driving system(s) (ADS) that the designer wishes to put into situation, can now be assigned to it. This is known as a ”use case”. Finally, the term ”test case” is appropriate only after instantiation: that is to say the assignment of determined values to all the variables concerned by this use case, and which can be processed for our needs.

Today, this definition has evolved and become more precise, notably by taking into account the terminology used by the peer PEGASUS project in Germany [6, 7]. Without addressing these revisions yet, it can nevertheless be said that a scenario is defined today as the composition of a series of scenes linked together by events or actions. These scenes constitute the nodes, that is to say, states frozen in time and succeed one another by the realization of events or actions. With the exception of the initial scene, the scenes are then the expected or controlled consequences one of the events or actions that precede them.

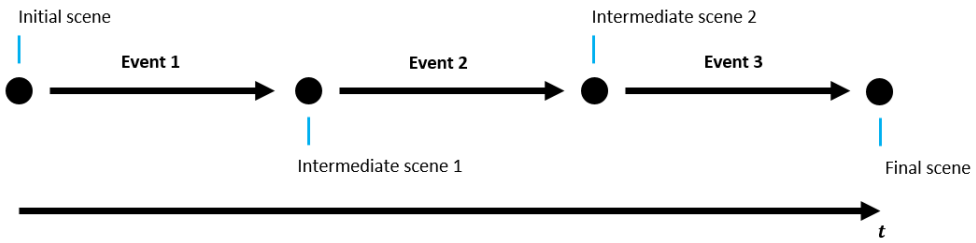

Figure 12: Temporal representation of a scenario

These two concepts of event and action will be specified in the sub-sections which are assigned to them. Below, the list of elements that a simulation scenario must include to be complete from the point of view of the writer:

- A title: The title indicates in a few words the interest of the scenario considered

- Dynamic elements - Actors of the scenario
- Static elements - Road infrastructure
- Environmental elements - Weather
- An initial scene - Initial conditions
- An event:
  - change of state of an object outside the ego
  - e.g.: a vehicle folds down just in front of the ego-vehicle
  - e.g.: pads are on the path
- And/Or an action :
  - A change in the state of the ego vehicle
  - e.g.: The ”under test” function is activated, the ego vehicle brakes
  - Note 1 : the ego vehicle does nothing as an action
  - Note 2 : Each scenario contains at least one action or event
- Intermediate scenes (optional)
- A final scene - end criteria
- Measurable variables - Specifying, invariant and variability parameters

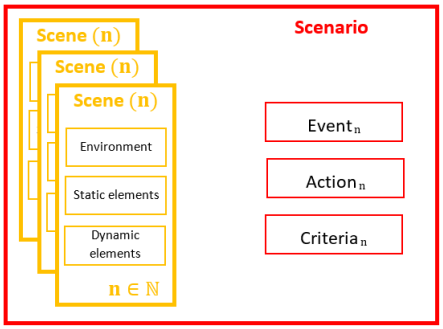

Figure 13: Scenario diagram [6]

2.2.3 Use cases

The use case is the expression of the situation of a driving assistance system, or autonomous driving, in a contextualized driving scenario. This scenario illustrates both the usefulness of this system (why we use it) as well as its use (how we use it), that is to say the appropriate use of this system. in a specific context, and the need it meets.

From the point of view of the designer of this system, this situation makes it possible to highlight the expected response of the latter, which is specified in a rigorous and measurable manner through the requirements of the system’s definition.

A use case is therefore a driving scenario that is exploited in order to verify the expected behavior of a system carried by a designated ego vehicle. This expected behavior is verified by compliance with the requirements associated with this system. Finally, each requirement is verified by compliance with the validation criteria associated with it. The use case also contributes to the restriction of the possibilities of infinite variation of the scenarios. In fact, a system to be tested is bounded by an operating domain with: limits, precision, margin of error, etc.

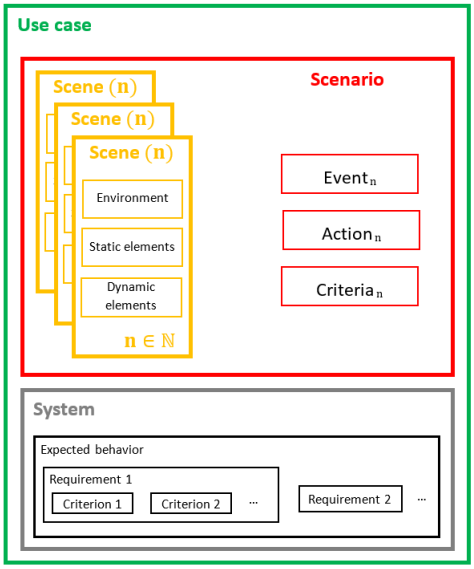

Figure 14: Use case diagram [6]

2.2.4 Test cases

When the specifying and invariant parameters are defined for a given scenario, the test designer or writer will now have to precisely define the variability he wishes to apply to the parameters of his scenario. This variability can result in a range of values and a cutting rule specific to each parameter. This variability must be chosen in a manner that is relevant to the use case under consideration, and judiciously so as to limit the infinity of combinations and possible variations to a reasonable number.

This selection can be controlled in multiple ways, for example: regular intervals, Gaussian distribution.

Once this draw is made, we obtain a finite number of possible combinations of the set of parameters for our scenario. When the test writer, or the automatic generation tool applies one of the parameter sets (from a draw), to the use case concerned, this is called a test case. In other words, variability consists in discretizing our scenarios across previously chosen ranges of values. The test case is then an instance of these possible use cases. This instance is defined by the entire set of parameters applicable to this use case.

2.2.5 Introduction to functional, logical and concrete scenarios

The definitions set out so far remain insufficient, as the distinction between these levels of description remains unclear, and can only be drawn from the use case.

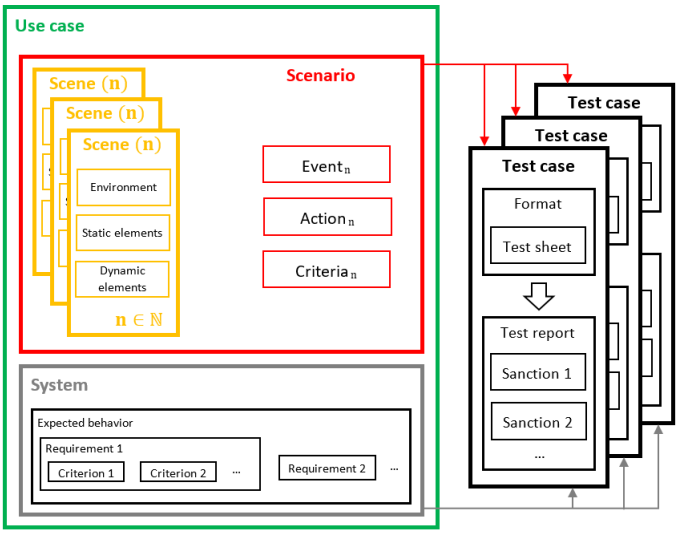

Figure 15: Test case diagram [6]

In fact, the use case constrains the scenario, then the test case in turn constrains the use case. However, the definition of the scenario alone does not propose to isolate levels of detail and depth of information during the parametric description of the scenario.

In addition, the work of writing use cases in a formalism based language has highlighted the dominant role of illustrations. While they should in principle only support the point without being specifying, they ultimately proved to be essential for understanding what the use case writer wanted to describe.

This finding poses a problem for the digital coding of the scenarios in question. Sometimes too rich in superfluous details, or on the contrary lacking information essential to understanding, the method which consists simply in enumerating parameters under the fields assigned for the scene, the action, and the event proves to be flawed and largely insufficient. The question that finally arose was the following: how to describe a comprehensible driving scenario without making illustrations essential?

The presentation boards of Hungar et al. [70] set out an interesting approach to answer this question, by proposing to distinguish three levels of description for a scenario: functional, logical, and concrete.

- The objective of the functional scenario is to specify ”high level”. This is a text description, possibly with a single optional illustration, allowing for concise writing and quick reading. It is nevertheless possible to assign values to parameters, only if they are specifying parameters for this scenario, that is to say parameters which would allow us to distinguish our functional scenario from another. On the other hand, no range of values is indicated, and the possible variants are not specified. The important thing here is to refer to the parent macro-scenario to which it belongs, and to specify verbatim the evolution of the relative positions of the actors in relation to each other.

- The logical scenario is the level of description ”medium level” which divides the parent functional scenario. It provides additional details to the functional scenario by scrolling down the exhaustive list of all its possible variations, evolutions or variants of the latter. For a logical scenario, the sequence of scenes, actions, and events is defined, in short, the logic of the scenario’s progression is fixed. The logical scenario goes further, and specifies

for all the parameters implemented, their ranges of values, as well as their occurrences and their probabilistic distributions. It is therefore central, because it is at this level that we allow ourselves to set scenario criteria, and that we must take into account the consistency of the scenario in the intervals of defined values, it that is, to guarantee the plausibility of the scenario in order to avoid impossible cases. Another major advantage of this method is the fact of imposing ”criticalities”. This notion forces the user to inform the variables on which he will pay particular attention with regard to the system to be tested. Also, the process of adding textual attributes (example: left, from behind), while respecting a certain logic, should favor, with ”criticalities”, a significant limitation of the quantity of variables to be bound, and thus alleviate the parametric specification of the scenario.

- This notion is in line with our definitions, because the concrete scenario is a possible instance of its parent logical scenario. Not to be confused with the test case, which is associated with the system. Finally: Concrete scenario + System = Test case. (Figure 16 shows a summary of the introduced notions)

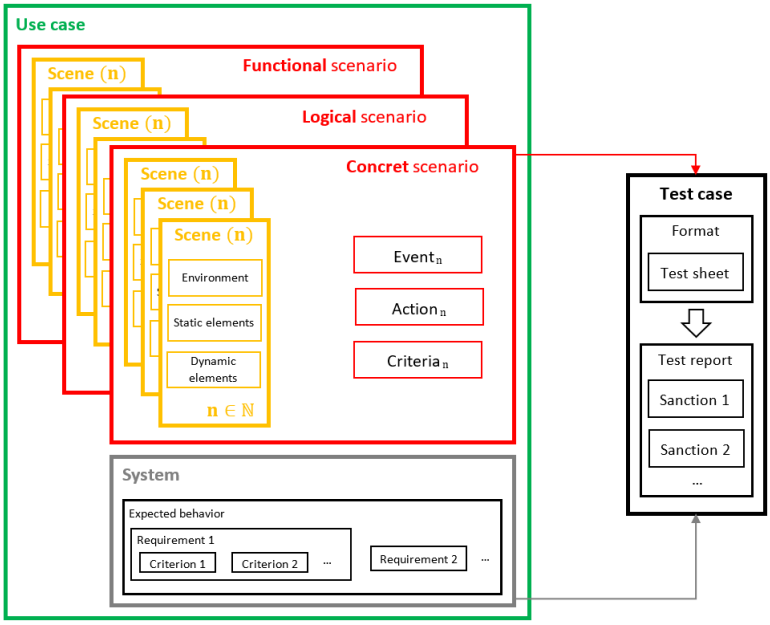

Figure 16: Redefined use case diagram - with PEGASUS definitions [7]

2.3 Automatic generation of test cases

The verification and validation of automated driving systems (ADAS) is a complex issue, as these systems can be confronted with a very large number of situations that can be considered endless. However, these situations do not have the same influence on the correct operation of the system and do not all occur within the same time-frame. Whatever the nature of the data used for validation, real or simulated, the Model-Based Testing approach can be used to automatically build a complete test base of limited size while ensuring coverage of most of the situations that have the greatest impact on the proper behaviour of the system under test.

2.3.1 Global strategy of model-based testing for test cases generation

Test case generation faces the question of inherent combinatorial explosion. As stated by the MaTeLo tool [2], the problem is to produce samples of large random vectors, whose components can be dependent and take a finite number of values with given probabilities. An important constraint is to generate almost all the situations as economically as possible. To achieve this, the COVADEC project opts for the probabilistic approach, which is based on the implementation of a Gibbs sample, briefly described below.

- The first step is to construct a Markov random field from the simulation graph generated by MaTeLo. When the parameters depend locally, this can be done using Bayes formulas.
- Next, test cases will be obtained using Gibbs samplers. In particular, we will seek to optimise the rate of convergence towards equilibrium, since we know theoretically that the speed of convergence is exponentially fast.

2.3.2 Gibbs samplers and random fields

In order to simulate systems with large state-space and given multi-dimensional distributions, such as those encountered in statistical physics to study equilibrium properties, powerful methods have been proposed as soon as in the 1950's. In particular, the Metropolis-Hastings's algorithms [71]. In the context of image processing, where digitised images can be viewed as the realisation of some random field, one must quote the seminal Gibbs sampler work [72].

- The first possible Gibbs sampling is a random scan:
  - We arbitrarily select an initial use case configuration, i.e. $X(0)$ and a vector of initial probabilities $(\alpha_1, \alpha_2, ..., \alpha_p)$.
  - On the j^{th} iteration, we choose a parameter s with probability αs.
  - The use case $Xs(t)$ is generated with probability $\pi(Xs|X_j(t-1), j \in Ns)$.
  - Iterate until converging to stationary probabilities.

We show that the Markov chain $X(t)$ obtained in this way is reversible and its invariant measure is the stationary distribution π of the vector X.

- The second Gibbs sampler is with periodic scanning that can be likened to a Markov random field:
  - In this case, the parameters are always traversed in an order fixed one and for all, which is a permutation of $(x_1, x_2, ..., x_p)$, i.e. $(s_1, s_2, ..., s_p)$
  - A Markov chain Z(t) is generated.
  - At each elementary stage, we select Xs_1 conditional of the current state, then Xs_2 as a function of Xs_1, then the same up to X_{sp}. This Markov chain has then made a transition, and by the iterative repetition of the process, we also obtain a Markov chain whose invariant measure is the stationary distribution π of the vector X.

[2]The MaTeLo tool was launched in 2002 by a consortium of industrialists and academics from the space industry, with the aim of deploying model-based testing methods, a project that came to fruition in 2004.

$$|X(0)M^n - \pi| \leq 1/2|X(0) - \pi|\delta(M)^n \quad (1)$$

Where M is the transition matrix of the Markov chain obtained from a Gibbs sampler and, between 0 and 1 is the ergodic coefficient of M, also known as the Dobrushin coefficient:

$$\delta(M) = 1 - inf_{i,j\in\Omega} \sum_{k\in\Omega} p_{ik}p_{kj} \quad (2)$$

A metric likely to provide a criterion for stopping test campaigns relating to the infinite universe of use cases can thus be constituted by this inequality.

2.3.3 Parameters required

In order to cover all the situations that ADAS systems may face, it is necessary to know the various situational variables used to initiate the driving environment and the context in which the system under test operates. These variables are made up of several values whose distribution follows a statistical distribution that must be known. The aim is to provide a meta-model of the test sequences, taking into account the influential parameters that express the variability of the situations with which the system may be confronted. The construction of such a model involves taking into account parameters of a heterogeneous nature, having very diverse impacts on the scene as perceived by the system. A model of the way the system operates must also be produced, in the form of Markov chains explaining the different operating states of the system and the conditions validating the transitions from one state to another, relating to the system's internal and external parameters. This model needs to collect information about the environment in which the ADAS operates, weather conditions and driving situations (i.e. ODD, OEDR and behavioural competencies of the system). The modelling of the environment must be as complete as possible. In fact, the model is supposed to represent all the situations that the vehicle may encounter. Raffaëlli and al. have proposed several categories of influential parameters [8], which are presented below.

- **Weather conditions:** have an impact on how the ADAS will perceive a scene. This includes not only the weather as such, but also disturbances induced by these conditions as well as the lighting conditions of the scene.
- **Structure of road and the environment:** this category includes the intrinsic characteristics of the road, that is to say the parameters to accurately describe its structure (curvature, topology, number of lanes, etc.), as well as its appearance and overall look (surface, marking, etc.).
- **Behaviour of the equipped vehicle:** this category is used to express the behaviour of the equipped vehicle in a test sequence, both in terms of speed or trajectory rate of change. In addition, this category includes the actions of the driver that may impact the function without implying a change in the trajectory or speed (e.g. wiper operation).
- **Behaviour of surrounding vehicles:** the presence of other vehicles can influence the perception of the scene by the ADAS either as a target vehicle or as a barrier masking what the ADAS should detect. The behaviour of other vehicles is described by a set of parameters identical to those defined for the behaviour of the equipped vehicle for which we have added parameters relating to their positioning in the scene as well as changes of trajectories they can make.

- **pedestrians:** this category of parameters can express how pedestrians will evolve in the scene (number, trajectory, crossing the road, etc.).
- **obstacles and disturbances:** this category includes all the obstacles and other disturbances known to have an impact on how the ADAS will perceive a driving situation. We grouped the barriers in several sub-categories, namely:
  - Fixed Targets set on the way: this includes work pads, a stationary vehicle, a lost loading or any other object that may be on the way.
  - Barriers at the trajectory limit: this includes road signs, guard rails, or a stationary vehicle.
  - Pedestrians in particular situations.
- **Equivalence classes:** the range of possible values for each parameter is divided into several equivalence classes, for two main reasons:
  - To select sets of values having a real impact on the ADAS function. This corresponds to the notion of "range", all situations are assumed equal within the range (e.g. 130 km/h and 131 km/h are considered equivalent in terms of ADAS, but 20 km/h belongs to a different equivalence class).
  - To mange the dependencies between parameters. Indeed, some values of an influential parameter X may not be possible or have a different probability if the parameter has a value Y (Y of X correlation - examples: "night" and "sunny" are incompatible; "speed $>$ 130 km/h" and "urban environment" is an unlikely event).

When building test campaigns, that is to say, sets of test cases which will be run for the ADAS function, if one test case has all its values in exactly the same equivalence classes another test case, it will be considered duplicate and eliminated from the campaign.

It should be noted that the list of these parameters is not exhaustive and could be found and completed by the list of ODD [3] and OEDR [4] parameters used by the other PRISSMA project tasks.

2.3.4 Results obtained

The method of automatic generation of test cases based on Gibbs samplers produces test cases consisting of a scenario which described the evolution of the state of the system and the parameters from an initial state, as a function of the characteristics of the system and the distribution of the parameters. A summary of the test cases generation as perceived by MaTeLo tool is given in 17.

[3] Operational Design Domain

[4] Objects, Events Detection and Response

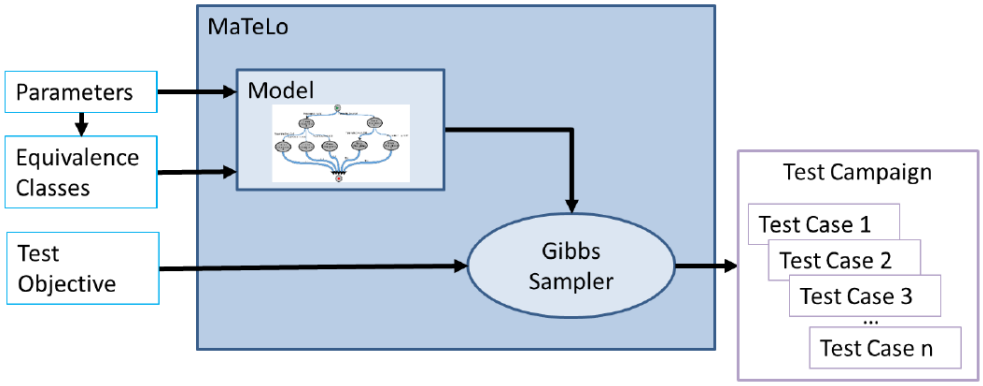

Figure 17: Summary of the automatic test cases generation (from [8])

2.4 Scenario management and generation platforms

Car manufacturers and suppliers are faced with a major scientific and technological challenge: to demonstrate the safety of the autonomous vehicle in a complex environment, generating a large number of driving situations in which it must react in a safe manner. The MOSAR platform (Methods and Tools for the Safety Assessment and Robustness Analysis of Autonomous Vehicles), is the result of IRT SystemX's collaboration with leading French automotive players, proposes a methodology and a tool-chain to design and validate the safety of the autonomous vehicle using a scenario database.

With the objective to define for French automotive industry a shared scenario catalogue for Automated Driving Systems (ADS) of SAE autonomy level 3 and more, SystemX decided to agree on the following term definition. We based this work on the following articles [67, 70, 6] that reflect idea close to the German PEGASUS project views and on the norms and standards from the software and system engineering community. Some of these definitions are already in ISO 21448 SOTIF standard [69], other are proposed for ISO/SAE PAS 22736 [73], ISO 22737:2021 [74] standard concerning ODD and Scenarios standardization for ADS.

Scenario Data Model for the database was created from this basis and from SystemX participation to OPENSCENARIO and OPENDRIVE discussions at the standardization organization ASAM). Scenario description parameters were chosen to be compliant with the maximal number of parameters used in the different input databases. in the following, a scenario is a specification of different classes of parameters.

2.4.1 Scenario languages and formats

The first brick is a way to import or set up a scenario described by the user. This require the tool to be able to either import a standard scenario description, or/and be able to access and adapt the various scenario and simulation parameter inside the tool. In [9], a number of domain-specific languages and systems are presented. These scenario description and specific languages have been developed to support scenario-based testing for Automated Vehicles. this paper makes a review and a comparison of the main features and characteristics of the major scenario description languages and systems (SDLS). The main scenario description languages are presented in the figure 18 where the authors of the survey provide the data format of each one, and give additional information about the road-network description format and the simulation platform dedicated AV which use them.

SDL	Data Format	Open Source	Road Description	Simulation Platform
stiEF	XML	Yes	OpenDRIVE	*ViresVTD*[2]
Hesperia	Programming language	Unknown	3D	CxxTest Prescan
OAS	HTML	Yes	2D	OAS
SceML	XML	Yes	OpenDRIVE	CARLA
OpenSCENARIO	XML	Yes	OpenDRIVE	Prescan CARLA *ViresVTD*[2]
GeoScenario	XML	Yes	Lanlets	Unreal Engine
CommonRoad	XML	Yes	Lanlets	SUMO
SCENIC	Probabilistic programming language	Yes	OpenDRIVE OSM	VERIFAI CARLA Webots LGSVL
M-DSL	Python-like Syntax	Yes	OpenDRIVE	CARLA SUMO

Figure 18: Comparison of main scenario description language used for AV evaluation and validation [9]

2.4.2 Scenario Manager

The Scenario Manager module provides access to a Reference Scenario Library, allowing its management, to generate test cases for virtual testing, bench tests, and proving ground or open roads. The purpose of these developments, carried out in collaboration with industrial partners, is to pool together tools and scenarios that are essential for the design, validation and approval of autonomous vehicles.

2.4.3 Test Case Generator

The Test Case Generator module should oversee the generation of test cases by exploiting the variability of the scenario parameters to represent the diversity of situations encountered by the vehicles.

Test case generation has been intensively studied in the last years. An extensive survey of test case and concrete scenario generation can be found in [12]. Most of the current methods start from manually created logical scenarios. Their configurations are based on frequent real accidents like unlawful crossing of pedestrians (sometimes with poor visibility) or cut-in manoeuvres on highway. The parametrization of these scenarios and the range values are then arbitrary chosen. Parameters like speed usually conform with the traffic laws while the range of parameters like the initial position of the actors depends on the use case of the scenario or the desired complexity.

However, not all the possible parameters combinations are critical and relevant, and only the challenging ones are interesting to run in order to assess the safety of the ADS. The concept of criticality is so introduced and the challenge is to design algorithms able to identify critical scenarios.

A common way to define the criticality is through a function of a physical characteristic

of the concrete scenario, usually the speed of collision or the distance to the obstacle after avoidance [75]. A more complex definition is instead proposed by [76], where the criticality is function of the solution space of the ADS.

Several methods exist to find relevant values of these parameters, mainly based on optimisation algorithms, but also exploiting machine learning techniques. The level of parameterization of the logical scenario strongly depends on the adopted approach. Several solutions have been proposed to model the scenarios: for instance, [77] uses probabilistic distributions to model the concrete scenarios while [78] have created an ontology to model the logical scenario.

In [79], the authors exploit the Matlab toolbox S-TaLiRo [80] to generate test cases of lane changing. As an experiment for their method, they manually define a logical scenario where an obstacle vehicle overtake the ego vehicle on a straight road, with another obstacle vehicle behind the ego vehicle. They then use stochastic optimization algorithms implemented in S-TaLiRo (simulated annealing, ant colony optimization, genetic algorithms, etc.) to find parameters of the logical scenario causing a collision between the overtaking and ego vehicles. Feedback from the simulator is used as input of the optimization to iteratively find critical test cases. A batch of test cases is run in the simulator and, if no relevant test case is found, the results of the runs are used by the optimizer for the generation of a new batch.

Bayesian optimization is used by [75] to generate concrete scenarios to test a low speed ADS. They use Systems-Theoretic Process Analysis (STPA) to create a logical scenario where the ADS must stop before colliding a pedestrian unlawfully crossing the road. STPA (explained with an example in [81]) is an hazard analysis technique inspired by control theory and system theory used to identify the possible failures and design flaws of a complex system. However, the use of STPA to create a logical scenario does not seem particularly relevant in the experiment of the article as the low complexity of the scenario does not fully justify its use. The objective of the proposed method is to find test cases where the ego vehicle hits the pedestrian, which correspond to a maximum of the objective function. The value decreases with the ego vehicle stopping further away from the pedestrian, like a safety area around the pedestrian. To better converge to relevant concrete scenarios, the optimizer exploits the feedback from the simulated execution of the previously generated concrete scenarios.

In [82], the authors consider an AEB (Automatic Emergency Braking) system and define a logical scenario with a vehicle cuts-in into the lane of the ego vehicle and then brakes in front of it. They use the constrained randomization tool Vitaq [83] to generate three thousand test cases and the simulator 3xD [84] to run the tests. The speed of the ego vehicle is not stored in the test cases, it is controlled in real time by the tool Vitaq which is connected to the simulator. When running a test case in the simulator, the tool can analyze how this test case and the precedent cover the logical scenario with respect to its parameters and their ranges. The speed of the ego vehicle in the the next run of a test case will be computed using the coverage analysis and the result (collision or no collision) of the previous test case execution to increase the coverage and find critical test cases.

[76] and [85] propose two similar methods to maximize the criticality of an existing concrete scenario. They both model the criticality of a test case as a function of the solution space (defined as the drivable area that does not lead to a collision) of the ADS tested: reducing the size of the solution space, and so limiting the space of possible safe maneuvers, results in an increase of criticality. The solution space is computed by combining a reachability analysis with the prediction of the trajectories of the obstacle vehicles. The two approaches then differ in the method adopted to optimize the parameters of the concrete scenario (initial positions, speeds, etc.) to minimize the solution space. [76] spatially discretizes the problem and models

it as a quadratic problem. Once this problem is solved, a binary search algorithm can be used to find a better solution around the previous discrete result. The work presented in [85] is an improvement of the precedent method. The authors propose a better modeling of the test case and use a pruning method to ease the work of an evolutionary algorithm which finally minimize the solution space. This model allows non-linear speeds for the vehicles, more complex road topology, and more actors in the scenario.

[78] proposes a method to test an AEB system by defining the associated ontology. The ontology is based on a functional scenario: a pedestrian crosses a street right in front of the ego vehicle that must stop before crashing into the pedestrian. The ontology contains elements and properties similar to a logical scenario: road topology, vehicles, weather, initial positions and speeds, etc. The test cases are generated by an n-way testing method, that takes as input a combinatorial testing model obtained from the ontology by a translation algorithm developed by the authors. Then, the obtained test cases can be executed in a simulator for evaluation (collision or no collision between the ego vehicle and the pedestrian). The authors propose two methods for converting the ontology and they show that one of these methods can effectively process a complex ontology representing the testing logical scenario of an AEB system.

In other works, machine learning approaches have also been used to create realistic and critical scenarios. In [77], a neural network generates safety-critical AV scenarios from an abstract scenario of an urban intersection. The obtained scenarios are modelled as series of probability distributions, from which a final one is sampled and run on CARLA. [86] uses a generative adversarial network, trained on a data-set of vehicle trajectories on a highway, to generate realistic lane-change trajectories.

It is worth noticing that, as pointed out in [12], current methods never test the complete ODD (Operational Design Domain) of the considered ADS. The scenarios (logical and functional) are instead restricted to specific use cases, such as highway overtaking or an urban intersection. As a result, the concrete scenarios and the generated test cases are not sufficient for the complete validation and verification of the AV and new solutions need to be developed.

In [10], the basic layer of a methodology for scenario generation is provides. This proposal includes the static concepts and the mobile ones. The static concepts are those defined for the highway infrastructure and the weather while the mobile concepts are those defined for the autonomous vehicle and the other traffics. Some of the static concepts, such as the lights, can change state but not their position. They are the dynamic concepts.

In [11], the authors define the scenario generation and execution stages for Scenario-Based Testing for Automated Driving Systems in High-Fidelity Simulation. This paper focus a part of the paper on the different functions and data links involved in the scenario generation, execution. They have defined the scenario generation process as following (see figure 20):

- First, a system S under test has to be chosen and defined. This system or component will be tested in a specific environment E. For instance, the system could be OPENPILOT system and the simulation environment could be CARLA.

- Next, the functionality of the system to be tested and its ODD have to be specified. In the paper, the author proposed OPENPILOT's ACC functionality on a highway.

- A corresponding logical scenario is developed next, which consists of two parts: a static configuration set SC and a searchable space SS. The static configuration SC consists of fixed configurations and parameters, e.g., the map information and the trajectory of some background vehicles with fixed behaviours. The search space SS consists of searchable

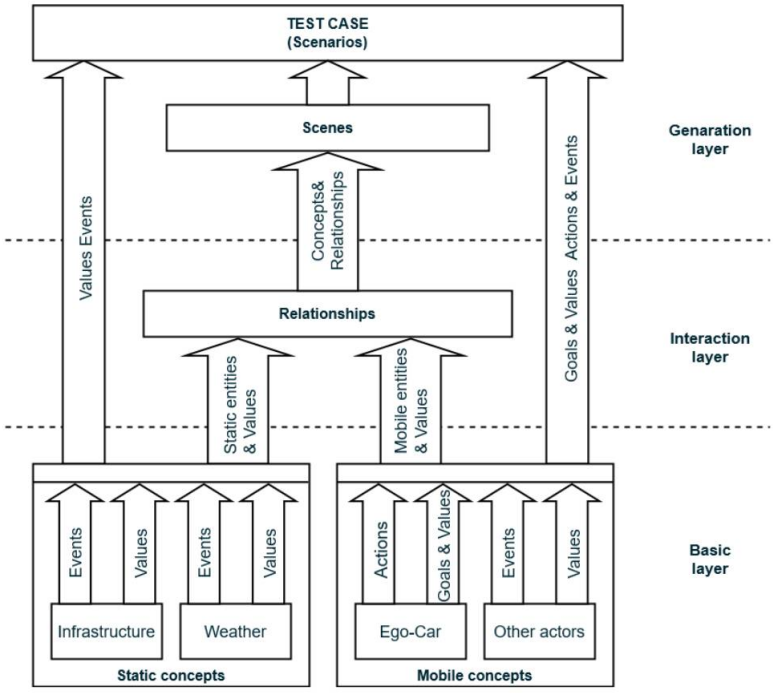

Figure 19: Test cases generation methodology ([10])

parameters, e.g., an NPC (Non-Player Character) vehicle's speed or the parameters of a Reinforcement Learning (RL) agent which controls an NPC vehicle's behaviour.

- Next, a testing objective is formulated. An example is finding more scenarios causing ego car's collision given a fixed time budget. A selection algorithm that can find the parameters from the searchable space SS to achieve the testing objective is then developed. Based on the search space and the testing objective, the algorithm can be either not adaptive, adaptive at simulation level, or adaptive at simulation step level. In other words, the algorithm can potentially update its search based on feedback at different frequencies. For any of the three categories, the selection algorithm samples a set of test parameters (scenario vectors) $p_1, ..., p_n$ and uses them to parametrize test functions $f_1, ..., f_n$. At each time step, a test function f_i takes in an environment state $S_i^{(t)}$ (e.g. the locations and speed of the ego car and an NPC vehicle) and outputs a test action $ta_i^{(t)}$ (e.g. the acceleration for an NPC vehicle).

- Finally, the test functions $f_1, ..., f_n$ along with static configuration SC are passed into the scenario execution module.

Then the scenario execution is applied. The scenario execution module consists of three parts: The first part is the current test function f_i, the chosen environment E, and the system under test S. All three parts have been provided or specified from the scenario generation module define previously. The test function f_i first outputs the initialization test action $ta_i^{(0)}$ (e.g. the initial speed of an NPC vehicle) to initialize the environment E. At every time step $t > 0$, f_i takes in the current environment state $S_i^{(t)}$ and outputs test action $ta_i^{(t)}$. Note that depending on the search space SS and the algorithm, the test action at $t > 0$ can either be empty or influence the environment e.g. controlling an NPC vehicle's acceleration. The environment E is initialized

using the static configuration SC as well as the initial test action ta provided by the test t_i at the beginning (i.e., $t_i^{(0)}$). At each time step t, the environment E takes in the test action $ta_i^{(t)}$ from the test function t_i to potentially modify the behaviours of NPC agents, and the system action $sa_i^{(t)}$ from the system M to control the behaviour of the ego vehicle. The environment E outputs the current state $S_i^{(t)}$ which usually consists of the positioning and states of all objects and the observation O_i^t which consist of the sensor information for the system S (e.g., the front camera, LiDAR, RADAR, IR camera, ... or other automotive sensors used by the system under test). It is also important to add a specific function allowing to generate the ground truth (see figure 20. At each time step $t > 0$, the system S takes in the observation O_i^t consisting of the sensor information and outputs the system action $sa_i^{(t)}$ containing the control commands on the ego vehicle in the environment. After all the test functions finish their corresponding executions, the execution results $r_1, ..., r_n$ will be returned to the algorithm that leverages this information to improve its scenario search potentially (optimisation process and complexity reduction). After all the scenarios execution finishes, the final results $r_1, ..., r_N$ will be passed to the evaluation module.

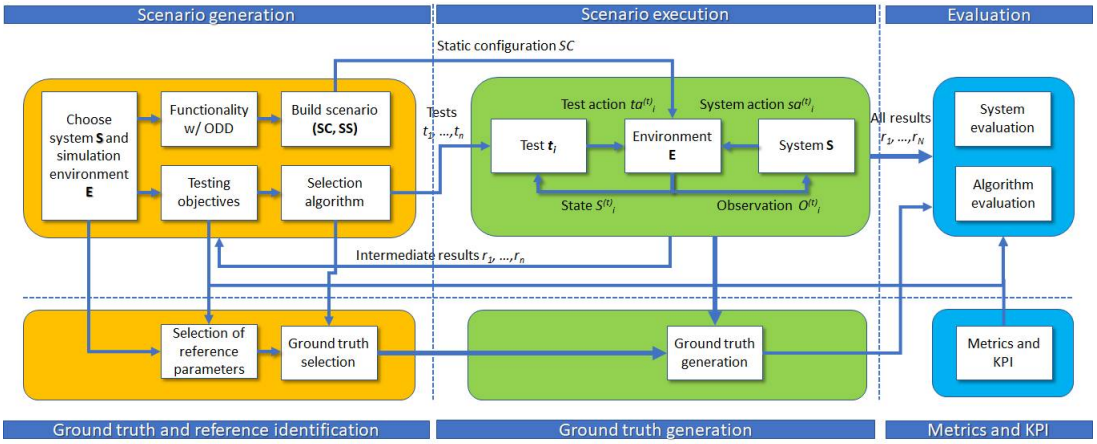

Figure 20: Adaptation of the general workflow of scenario-based ADS testing in high-fidelity simulators (from [11]) with ground truth generation (source UGE)

2.4.4 Virtual testing

The Virtual Testing module oversees the execution of virtual tests carried out with simulators and their analysis using the tools developed in the framework.

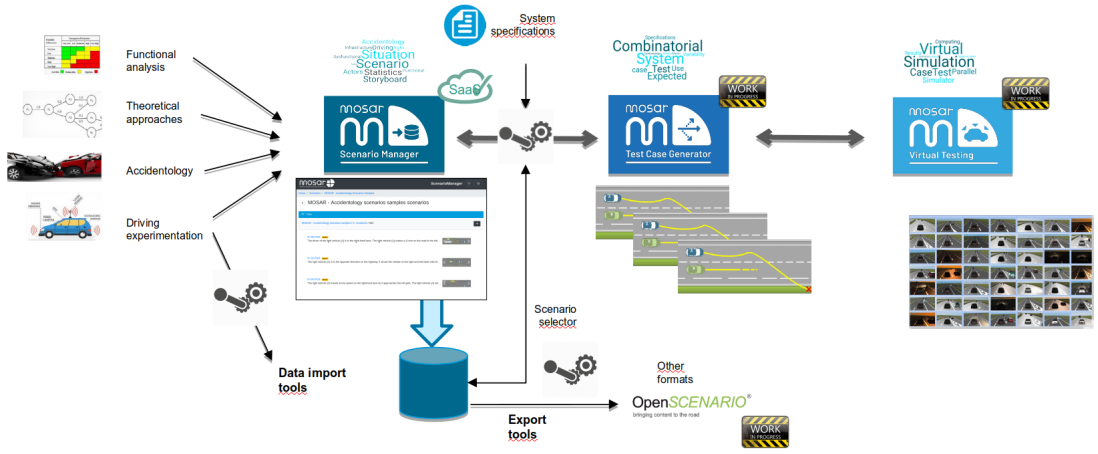

Figure 21: MOSAR Platform Architecture : a SaaS (Software as a Service) tool suite

2.5 Scene survey (digital twins)

The more intelligent the system tested is, the more realistic the scene needs to be. A scene is composed of two parts. A logical one that will define the infrastructure (roads, signs..) and then the 3D environment. While the first part can be obtained from various services like Here or OpenStreetMap it is more complicated to generated a photo realistic environment.

There are different approaches:

1. Manual methods: modeling by a graphics designer. Realism can be good based on photos, but small details and high-quality textures take a very long time to model. It can be expected that about 1 km of road can be produced per man month.

2. Automatic or semi-automatic approaches:

    - Modeling from data extracted from images. These approaches make it difficult to obtain a photo-realistic rendering of a scene because of the average quality of the reconstructed geometry, especially for large scenes. The company Virtuel City, for example, is based on 3D modeling constructed using aerial photogrammetry, incorporates high-definition texturing of buildings through a campaign to acquire ground photographs of buildings along the paths via a SnapCar car.
    - Modeling from data from time-of-flight sensors. LiDAR (Light Detection And Ranging) sensors make it possible to measure very precisely the distance between the sensor and objects and to recreate their general shape in the form of a cloud of points.

3. Procedural automation software : There are not many software that offer possibilities for three-dimensional urban modeling. For example, CityEngine, LandSim3D and Infrastructure Modeler software are based on procedural techniques which do not reflect reality and which cannot push the level of detail far.

Laser reading or LIDAR (Light Detection And Ranging) is an optical measurement technology based on the analysis of laser light returned by the environment. The mobile laser surveys

with the help of taking photos make it possible to digitize with great precision and high productivity (factor 10 with conventional survey techniques) both the geometry and the colorimetry of an outdoor scene.

Figure 22: 3D point cloud of Lille from MINES ParisTech

The digital twin aims to reproduce the geometric configuration integrating the materials, textures, and objects constituting the scene (road, barriers, vertical signs, bridges, tunnels, poles, etc.). For materials (specular, diffuse, ambient, shininess ...components), it also includes meta information characterizing the physical properties of materials (with SER, BRDF, level of conductivity, level of granularity, reaction to heat, HDR data for light intensity modeling.. .). About texture, it is necessary to be able to take into account a large panel of different types of textures (HDR, seemless, procedural, animated, bump-mapping, normal map, sphere environment map, modulate, ...) Therefore, declining the possible observations of a virtual sensor in an environment at a time t different from the time of data collection, it is simply necessary to vary the environmental conditions (light and on HDR, atmospheric and meteorological conditions) and the simulation will behave in a physico-realistic manner.

In Pro-SiVIC platform, a digital twin of the Satory test track has be developed.

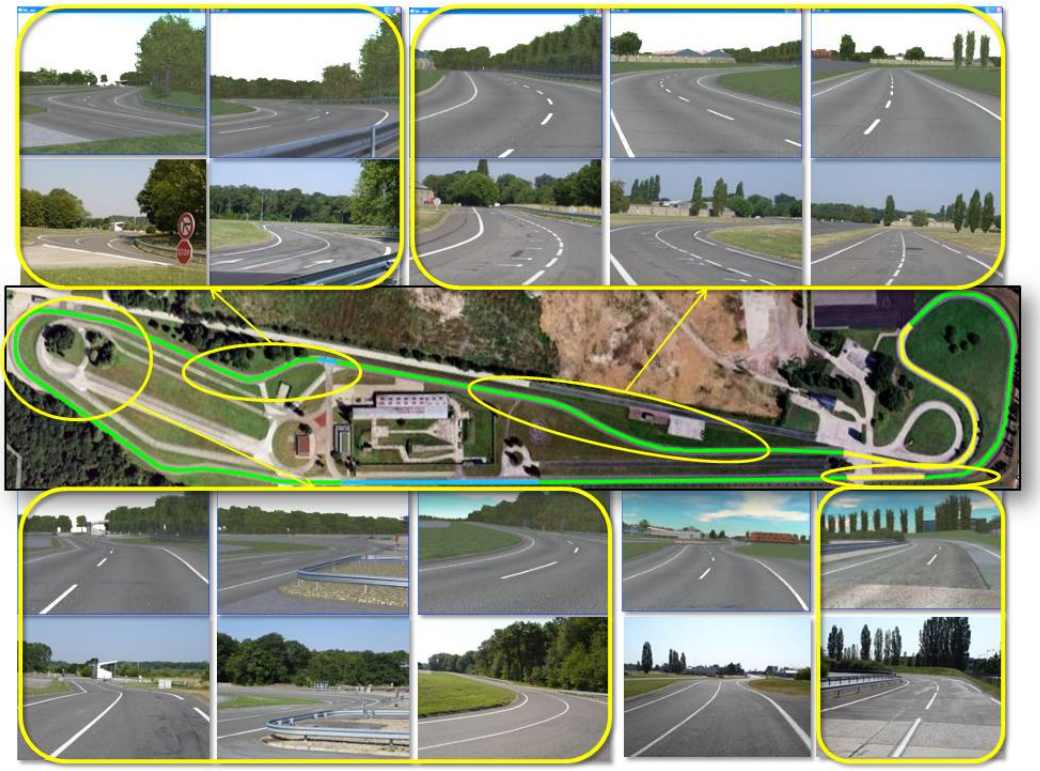

Figure 23: Digital twin of the Satory test track in Versailles. Implementation in Pro-SiVIC^{TM} (source: Univ Eiffel)

2.6 Identification of critical use cases

2.6.1 Critical and relevant scenarios

The Automotive Platform (PFA), which brings together the automotive industry in France, proposes as a definition for the relevant scenarios any scenario highlighting a safety risk or intervening in the definition or sizing of a safety barrier [5]. These scenarios can be used from the design phase to the validation and approval phases.

According to German work published in the paper [87], relevant scenarios refer to all scenarios that contribute to the validation of automated vehicles. The relevant scenarios can also be very simple, for example the start of a speed limit. This is relevant for certification homologation tests because an automated vehicle must comply with current traffic rules.

The scenarios that we seek to collect are those having an interest for the design and validation of the safety of autonomous vehicles, because:

1. It is a functional, logical or concrete scenario that can be used during the design or validation of a system: if the scenario presents a situation that can occur during the use of the system and a improper behavior can result in a dangerous situation. A - “List of potential dangers” should be provided as the list of the dangers to be taken into account in the case of autonomous vehicles. It may in particular be the description of a situation observed and in which the system or its environment have been endangered (experience feedback).

2. It is a logical scenario that can be used during the validation or for the evaluation of the ef-

[5] https://wiki.unece.org/download/attachments/92014309/VMAD-05-12%20AD%20safety%20validation%20-%20french%20views%20-%20Vdef.pdf?api=v2

ficiency of the safety of a system, because the scenario, in the expression of its variability, makes it possible to cover a set of known situations that the system may encounter.

3. It is a concrete scenario that is part of a set of scenarios that can be used for the purpose of extracting statistical information on the probabilities of occurrence of a situation or on the distribution of parameters. a situation in a given context. In each of the cases mentioned above, the notion of the relevance of a scenario depends on the system to which it is applied.

There is also some other scenarios coming from different sources (scenarios of real world driving collected data, scenarios of incident and accident studies, and scenarios coming from the safety expert of car-manufacturers and suppliers) that could be considered as relevant. We cite here the example of the present scenarios in the MOSAR platform as shown in figure24.

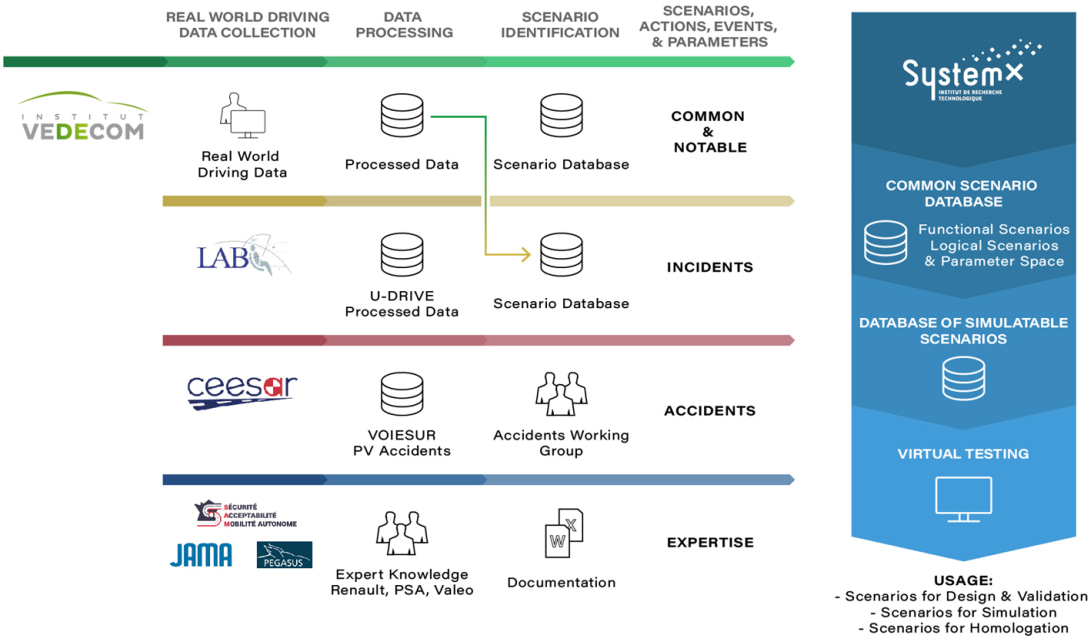

Figure 24: Relevant scenarios considered by MOSAR platform

Some other methods and process are proposed by Working Group (WG STPA) and companies in order to identify critical scenarios and the level of risk of a specific road segment. For instance All4Tech propose a platform called MATELO which allows to generate, from an identified set of relevant parameters and variables, the full scenario and situation combination. Then a filtering with some constraint allows to extract the sub set of configuration which respond to expected situation (i.e. critical road situation). WG STPA worked from 2019 to 2020 on the sharing of the road context in a set of segments. For each segment, a level of risk has been assessed.

2.6.2 SOTIF approach for critical scenarios identification [1]

A new version of SOTIF standard has been released in April 2021: ISO/DIS 21448. It is linked with a new version of taxonomy SAE J3016. This standard gives a methodology to identify and test scenarios which can lead to accident without failure.

The standard introduces the notion of Triggering Condition. It consists in an initiator element of the environment which transform a well-tested scenario into a hazardous one. To identify Triggering conditions the SOTIF propose to consider functional insufficiencies of the system: insufficiencies of specification or limitation of performance. These two elements come from designer knowledge and experiment and field feedback from other systems with the same technology.

Once triggering conditions are listed, in a second step, they are combined with operational and test scenarios in order to identify the consequences. Mixing triggering conditions and scenarios allow to explore a larger pool of relevant situation and identify critical ones.

In a third step, SOTIF Process propose to explore unknown scenarios to ensure robustness rather than completeness, by introducing variability from known situations.

In each step, criticality is reused from ISO 26262 functional safety analyses which are previously defined.

SOTIF assumed that the system works in nominal mode with appropriated level of performance before considering impact of triggering condition.

2.6.3 Scenario space and complexity reduction

To address the variety of real-world traffic situations, relevant test scenarios have to be selected from the scenario space through combinatorial management for the AV safety assessment. In their survey on scenario-based approaches [12], Riedmaier and al. describe 2 main types of scenario selection methods:

- Testing-based / coverage-based selection
- Falsification-based selection

Most testing-based selection methods rely on sampling strategies within the whole scenario space to ensure a good coverage.

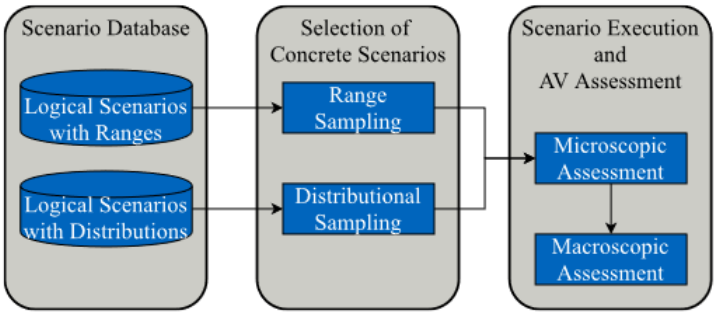

Figure 25: Testing-based scenario selection techniques from [12]

Some strategies consider all possible combinations of parameters, [88] [89] [90], while others do sampling from parameters distributions to select scenarios that will most likely be encountered in real situations. Among them, [91] [92] introduce the Extreme Value Theory approach with a criticality metric to sample scenarios. In [93], they use an Importance Sampling Theory to provoke critical situations. Other approaches, [94] [95], rely on Markov Chain Monte Carlo sampling.

These selection methods have the advantage of ensuring a good coverage, however, they require an important amount of tests, which does not solve the scenario space expansion issue.

On the other hand, falsification-based selection techniques aim to find challenging scenarios where the safety requirements are not met, which enable to handle the combinatorial explosion.

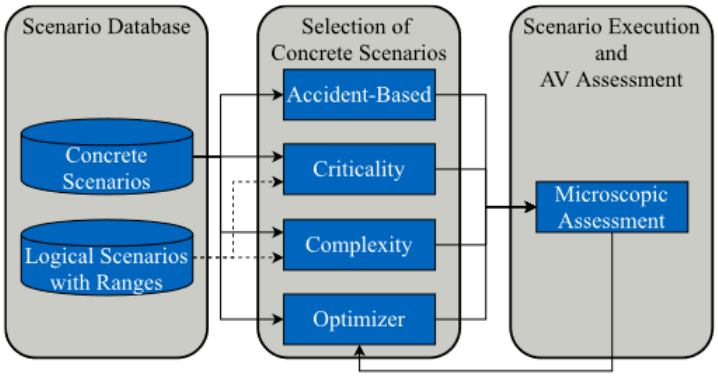

Figure 26: Falsification-based scenario selection techniques from [12]

Among these methods, some select concrete scenarios from accident database, such as in [96] [97] [98], but this is not sufficient for a global safety assessment.

Others consist in increasing the criticality or the complexity to generate challenging scenarios. The criticality is increased by minimizing the safe aera that be can used by the AV in [85] [76]. In [99] [100] [101], they increase the complexity of scenarios thanks to a complexity index which assigns weights to the scenario parameters, and combine that approach with combinatorial testing.

Other interesting approaches are simulation-based techniques. These methods rely on a feedback loop and a simulator to execute test scenarios selected at each iteration thanks to optimization algorithms.

Some work, [102] [103], propose an approach called Adaptive Stress Testing. It relies on Deep Reinforcement Learning and Monte Carlo Tree Search to find failure scenarios using rewards. They illustrate their method by generating sensor noise and pedestrian trajectories in order to select challenging scenarios.

Another approach is to use a surrogate model to approximate the simulation behavior and explore a large part of the scenario space while reducing the cost of simulation. In [104], they use Kriging as the surrogate model and Differential Evolution Genetic Optimization and Particle Swarm Optimization for the optimizer. This approach was introduced in [105] but they used a neural network as the surrogate model instead of Kriging.

A Range Adversarial Planning Tool framework is developed in [106]. They use adaptive search algorithms to find test scenarios close to the system's performance boundaries. These boundaries are identified using unsupervised clustering models. Related work use Bayesian optimization [107], random forest models [108] or simulated annealing [109] as search techniques to identify parameters that lead to failure cases.

Each method alone has its shortcomings but a combination of several of them, as suggested in [12], seems interesting and necessary. For instance, a combination of covering arrays for combinatorial testing and simulated annealing for falsification in a simulated-based framework

is proposed in [110]. The first selection with the covering arrays creates better initial conditions to allow a faster convergence of the optimizer during the falsification process.

Moreover, as stated in [12], all of the solutions cited above are promising but have only been tested in a limited scope, insufficient for a complete safety assessment for the moment.

3 Metrics and performance indicators

In the realm of autonomous vehicles, the need for metrics and KPIs to evaluate the performance of AI embedded within them is paramount. These metrics serve as vital yardsticks for assessing the efficacy, safety, and reliability of the AI systems governing these vehicles.

One crucial kind of metric is about safety. Given the potential life-or-death consequences of autonomous vehicle actions, safety metrics are of utmost importance. These metrics may include collision rates, near-miss occurrences, and adherence to traffic rules. By measuring these parameters, developers can gauge how effectively the AI navigates complex real-world scenarios and ensures passenger safety.

Reliability is also a key metric. It assesses the consistency and predictability of AI behavior across various conditions and scenarios. Reliability-type metrics may involve measuring failure rates, system downtime, or performance degradation over time. A reliable AI system inspires trust among users and stakeholders, essential for widespread adoption of autonomous vehicles.

Moreover, performance metrics are essential for continual improvement. These metrics may include computational speed, decision-making accuracy, and adaptability to dynamic environments. Analyzing performance metrics enables developers to identify bottlenecks, refine algorithms, and enhance the overall capability of AI systems.

Furthermore, ethical considerations play a role in metric selection. Metrics should align with ethical principles such as fairness, transparency, and accountability. For instance, evaluating AI decision-making against ethical standards ensures that autonomous vehicles prioritize human safety and well-being. We can talk also about efficiency. This encompasses factors such as energy consumption, time taken to reach destinations, and overall system responsiveness. Evaluating efficiency allows developers to optimize AI algorithms for smoother, more economical operation, ultimately enhancing user experience and resource utilization.

In summary, metrics are indispensable tools for evaluating AI in autonomous vehicles. They provide quantitative insights into safety, efficiency, reliability, performance, and ethical compliance, guiding the development and deployment of AI systems that meet the highest standards of quality and effectiveness in real-world settings. So in the evaluation and validation of system of systems, several aspects have to be addressed.

- Firstly, we have to define a set of KPIs which allows evaluating and validating the **quality of the system** in its wholeness. Is this system operating in its operating domain?
- Secondly, about the application, and in case of automated driving with or without the human aspect (Driver), we have to **quantify the level of risk** either perceived and/or generated by the application and by extension the CAV. The goal is to guarantee a minimum risk level.
- Thirdly, we have to quantify **the quality and the performance of each component** of the application (each system of the system of systems). For instance, it is the case for the different perception modules, or the decision-making module, or the path planning system.
- Fourthly, in the simulation context, we have to use a set of models for the environment, vehicles, sensors, pedestrians, road infrastructures, etc. In this case, we have to prove the **validity of the models** comparatively to the actual systems and components.

In the rest of this section we'll go into detail about the various KPIs and metrics used to validate VAs' embedded AI in simulation.

3.1 Definition of Key Performance Indicators for service and system of systems

In the literature, several projects have attempted to address the problem of defining a set of KPIs for validating systems of systems. One of the most comprehensive is the MAVEN project. The MAVEN H2020 Project [6] has used specific KPIs to its program:

KPI ID	KPI description with units	Expected impact
KPI 1	Number of stops at traffic lights (-)	Reduction
KPI 2	Control delay time (s)	Reduction
KPI 3	Produced emissions (g)	Decrease
KPI 4	Fuel consumption (l)	Reduction
KPI 5	Throughput (veh)	Increase
KPI 6	Travel times (s)	Reduction
KPI 7	Minimum time to collision (s)	Increase
KPI 8	Number of human interventions for safety (-)	Decrease

Figure 27: MAVEN specific KPIs

This same project has also used specific KPIs from TRANSAID [7]:

No.	KPI Name	KPI Description
1	Mean time headway (THW)	The mean value of the time gap to an object (e.g. a lead vehicle (bumper to bumper) or pedestrian, which is travelling in the vehicle's path of travel).
2	Standard deviation of time headway	Defined as the standard deviation of the THW.
3	Average delay time (per distance)	Extra travel time due to negative deviation from the intended speed profile.
4	Average travel time (per distance)	Time required to travel from origin to destination for a vehicle.
5	Average stop time (per distance)	Average time at standstill per vehicle per kilometre.
6	Throughput	Total number of vehicles per hour through a particular road section or intersection approach, normalised to number of lanes and proportion of green time (where relevant).
7	Average network speed	Average space mean speed of the vehicular fleet on a specific road network.
8	Average density	Average number of vehicles per kilometre for a specified road segment.
9	Average flow rate	Average number of vehicles per hour that have passed through a specific location of the road network during the simulation period.
10	Number of stops	Average number of stops per vehicle per kilometre.
11	Number of lane changes	Total number of lane changes per kilometre.
12	Average queue length	Average queue in a specific road segment during the simulation period. It is measured in vehicles.
13	Maximum queue length	Maximum length of the queue in a specific road segment, expressed as number of vehicles per lane.
14	Total travelled distance	Total number of kilometres travelled by all the vehicles that have crossed a specific road segment.

Figure 28: TransAID specific KPIs: Network efficiency

[6] http://adas.cvc.uab.es/maven/wp-content/uploads/sites/16/2019/09/MAVEN_Schindler_KPI.pdf
[7] transaid.uic.fr

No.	KPI Name	KPI Description
1	Mean speed	Mean vehicle speed
2	SD speed	Standard deviation of vehicle speed
3	Maximum longitudinal acceleration	Peak level of longitudinal acceleration achieved during a scenario.
4	Maximum lateral acceleration	Peak level of lateral acceleration achieved during a scenario.
5	Frequency of left lane changes	Time frequency of performed left lane changes (either time or distance based).
6	Frequency of right lane changes	Time frequency of performed right lane changes (either time or distance based).
7	Deviation from desired lane	Number of lanes from the current lane to the desired lane (0 if driving in the desired lane).
8	Frequency of active overtaking	Time frequency of active overtaking (i.e. overtaking conducted by the subject vehicle), either time or distance based.
9	Frequency of passive overtaking	Time frequency of passive overtaking (i.e. overtaking in which the subject vehicle is overtaken), either time or distance based.
10	Frequency of lane exceedances	The number of times per a certain distance or time the vehicle leaves the own lane boundaries.
11	Minimum accepted gap in lane changes	Minimum space or time gap accepted by a driver or vehicle to perform a lane change.

Figure 29: TransAID specific KPIs: Vehicle Operations

No.	KPI Name	KPI Description
1	Average fuel consumption (l/km)	Fuel consumed per road-km for a vehicle's trip from origin to destination.
2	Total fuel consumption (l)	Total fuel consumed by all vehicles on the road network during the analysis time-frame.
3	Average CO_2 emissions (g/km)	CO_2 emitted per road-km for a vehicle's trip from origin to destination.
4	Total CO_2 emissions (g)	Total CO_2 emitted by all vehicles on the road network during the analysis time-frame.

Figure 30: TransAID specific KPIs: Energy and Environment

No.	KPI Name	KPI Description
1	Mean of time-to-collision (TTC)	The mean time required for two vehicles (or a vehicle and a object) to collide if they continue at their present speed and on the same path. Measures a longitudinal margin to lead vehicles or objects.
2	Post Encroachment Time (PET)	Time lapse between end of encroachment of turning vehicle and the time that the through vehicle actually arrives at the potential point of collision.
3	Deceleration Rate to Avoid Collision (DRAC)	The rate at which a vehicle must decelerate to avoid a probable collision.
4	Time exposed Time to Collision (TET)	Summation of all time intervals that a vehicles experiences TTC values that are lower that a specific TTC threshold value.
5	Time integrated Time to Collision (TIT)	The difference between observed TTC and threshold TTC value for a given time interval cumulative to the time the vehicle traverses the study area.
6	Number of instances with hard braking	Number of instances that decelartion rate exceeds a minimum pre-determined threshold.

Figure 31: TransAID specific KPIs: Traffic Safety

No.	KPI Name	KPI Description
1	Mean duration of the transfer of control	Mean duration of the transfer of control between operator/driver and vehicle (when requested by the vehicle).
2	Maximum duration of the transfer of control	Maximum duration of the transfer of control between operator/driver and vehicle (when requested by the vehicle).
3	Total Number of ToCs	Number of ToCs performed in the whole network.
4	Number of ToCs (per distance)	Number of ToCs performed per kilometre.
5	Total Number of Lane Changes	Number of Lane Changes performed in the whole network.
6	Number of Lane Changes (per distance)	Number of Lane Changes performed per kilometre.
7	Total Number of MRMs	Number of MRMs performed in the whole network.
8	Number of MRMs (per distance)	Number of MRMs performed per kilometre.

Figure 32: TransAID specific KPIs: Transition Area specific

No.	KPI Name	KPI Description
1	Neighbourhood Awareness Ratio	The proportion of vehicles in a specific range from which a message was received in a defined time interval.
2	Neighbourhood Interference Ratio	The ratio between the number of vehicles outside the specified range from which the given vehicle received a message, and the total number of vehicles from which the given vehicle has received a message.
3	Latency	The time difference between the transmission and reception time of a packet.
4	Date age	The time interval between the instant when the data is generated in the source vehicle and the actual time.
5	Packet Delivery Ratio	The ratio of packet successfully received over the total number of packets transmitted.
6	Footprint	The total channel resources consumed by the radio of a single vehicle in time and space.
7	Channel Busy Ratio	The percentage of time that the channel is perceived as busy for a given time interval.
8	Messages received per vehicle	The number of messages of a specific type received by a vehicle in a determined time interval.
9	Inter Package Reception Time	The interval of time elapsed between two successful receptions of packets of the same type.

Figure 33: TransAID specific KPIs: Communication related

No.	KPI Name	KPI Description
1	Mean duration of the transfer of control	Mean duration of the transfer of control between operator/driver and vehicle (when requested by the vehicle).
2	Maximum duration of the transfer of control	Maximum duration of the transfer of control between operator/driver and vehicle (when requested by the vehicle).
3	Total Number of ToCs	Number of ToCs performed.
4	Total Number of Lane Changes	Number of Lane Changes performed.
5	Total Number of MRMs	Number of MRM performed.
7	Mean speed	Mean vehicle speed
8	SD speed	Standard deviation of vehicle speed
9	Maximum longitudinal acceleration	Peak level of longitudinal acceleration achieved during a scenario.
10	Maximum lateral acceleration	Peak level of lateral acceleration achieved during a scenario.

Figure 34: TransAID specific KPIs: Real world feasibility

Once you've decided on the set of KPIs you want to achieve, you need to associate related metrics with them to measure performance against the KPIs. The first set of metrics to be addressed in this deliverable are those related to VA security.

Classically, the metrics for the security of autonomous vehicles can have two levels of time granularity:

- **Microscopic** : metrics that focus interest only at the situation in a single, very specific moment;
- **Macroscopic** : metrics that evaluate a sliding window of time or a complete scenario.

Microscopic metrics must verify at least four constraints:

- Translate the danger likelihood;
- Detect an accident;
- Avoid false negatives;
- Increase when facing more danger.

3.2 Quantification of the risk level for automated driving systems

One of the examples of safety microscopic metrics that is beginning to emerge in standardisation is the Responsibility-Sensitive Safety (RSS) model was developed and implemented by Intel/Mobileye [111] but classically, "*Time to X*" type measures stay the safety metrics standard. RSS determine safe following distance to avoid collisions in the worst possible conditions (hard braking of the following vehicle during an acceleration phase of the EGO vehicle).

Another metric has been recently proposed by NVIDIA ([112]). This risk metric is called *Safety Force Field*. This metric considers uncertainty to provide confidence intervals for all the metrics needed to calculate the Safety Force Field constraints, such as shape, position (including distance), velocity, and acceleration.

One example of "*Time to X*" metric is the **Extended Time to Collision** as proposed by [13]. This metric is integrated in the Mobileye systems ([113]). It proposes to calculate the time to collision in the bi-dimensional plan 2D near the reality between the two nearest points of each vehicle, as shown below:

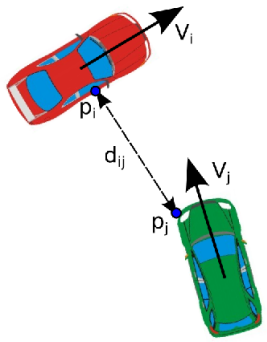

Figure 35: Definition of vehicle state parameters according to [13]. The position of each vehicle is the point on each vehicle closest to the other

Other "*Time to X metrics*" exist: "*Time to Break*" [114], *Time to Steer*, *Time to React* (maximum between TTB and TTS), etc. The "*Time to X*" measurements are then compared to user-selected thresholds (e.g. the 2 second safety distance) to give an indication of safety threshold violation.

In [14], an improved method to calculate the Time-to-Collision value between two Vehicles has been proposed. This approach extend the existing TTC to lateral collision. This study was made in order to improve integrated safety models of road vehicles. An overview of the different risk levels is given in the figure 36. In this modelling, a phase of interaction between primary and secondary safety systems has been defined.

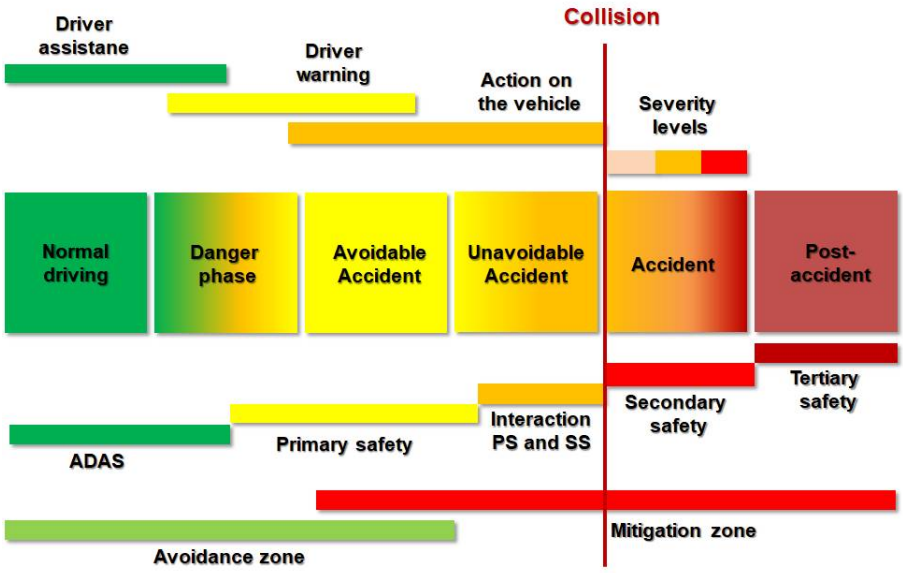

Figure 36: The different risk levels and situation intervals ([14])

In [115] the authors compare three different model-based risk measures by evaluating their strengths and weaknesses qualitatively and testing them quantitatively on a set of real longitudinal and intersection scenarios. They started with the well-known heuristic Time-To-Collision (TTC) and they extended it to a more generic 2D non-crash case to retrieve the Time-To-Closest-Encounter (TTCE). The second risk measure models position uncertainty with a Gaussian distribution and uses spatial occupancy probabilities for collision risks ([116]). From these risk indicators, the authors have derived a novel risk measure based on the statistics of sparse critical events and so-called *survival* conditions. The resulting survival analysis shows to have an earlier detection time of crashes and less false positive detections in near-crash and non-crash cases. It can be seen as a generalization of TTCE and the Gaussian method which is suitable for the validation of ADAS and AD. This work was awarded in the FAST-Zero symposium.

In [117] a summary of various surrogate estimators is proposed: TTC, deceleration rate to avoid the crash (DRAC), and Integrated conflict risk index (ICRI). This last indicator is used in a prediction modeling in lane changing maneuvers with intersecting trajectories. Another category of risk estimators is the probabilistic collision estimation methods. They assess the general risk of a situation and provide a probability of collision (occurrence of an event) and the severity of the risk, based on statistical analysis. There computation is mostly derived from other distance-based estimators for the probability thresholds and tuning.
In [66], the authors enumerate the more used metrics in the assessing of the criticality of scenarios and can easily be automated in a simulation setup:

- Time-to-collision (TTC)
- Time-to-brake (TTB)
- Time-headway (THW) or Gap time (GT)
- Post-encroachment-time (PET)
- Proportion of Stopping Distance (PSD)

- Initially-Attempt-Post-Encroachment-Time (IAPET)

Metrics from accident research, which take into account multiple factors have also been proposed to rate scenarios. Such metrics include the crash potential index (CPI) which takes into account the deceleration rate to avoid a crash and the maximum deceleration rate, and the road safety index (RSI) which is a function of injury severity score (ISS), traffic volume and number of crashes. Using the ISS on its own as suggested by Alvarez et al.(2017) [118] is not regarded as expedient, as it can only be applied once a crash with injury has happened. Hallerbach et al. (2018) [119] suggests to complement TTC and TTB with a combination of multiple traffic quality metrics, which take microscopic and macroscopic traffic data like the change of traffic density and velocity fluctuations into account. Combinations of multiple criticality metrics reduce the amount of false-positive and false-negative evaluations and should therefore be favored. A criticality assessment metric with a clear physical interpretation is introduced by Stellet et al. (2016) [120]. They use a collision energy reduction metric, which evaluates scenarios based on how the collision energy is reduced. This metric is however only applicable if a crash is unavoidable and cannot compare the criticality of non-crash scenarios to each other.

In the thesis [121], Althoff studied a collision probabilistic approach with an estimation method of the collision probability for a single time point using a multidimensional space cell decomposition. Their approach has the advantage to build a road map and search for pathway with smallest expected collision probability for motion planning. They based safety assessment on collision Probability of Collision State (PCS) for a vehicle which is found in an Inevitable Collision State (ICS) and they computed the Overall Collision Probability (OPC). However this method has a high computational cost. In [116] Lambert et al. have proposed a probabilistic approach for collision assessment related to distance of a subject to an object thanks to a provided Gaussian normalized function on multiple-dimensional space (x,y,V). Besides, a definition of the collision risk is presented as the product of the collision probability function and the cost of collision, approximated as the $EES : costcoll(V) = EES(V)$. The collision probability $Pcoll$ used is a multivariate distribution on 2D real space. Hence, the risk of collision Riskcoll(v) is computed with: $Riskcoll = Pcoll \cdot costcoll(v)$ In more recent works, [122] proposed an advanced collision risk modeling for autonomous vehicles, extended from an interaction-aware motion modeling based on Dynamic Bayesian Networks (DBN). They took in consideration a more global view of the current situation (network and traffic level) in order to estimate a "network level collision prediction". This prediction capacity allows to have a risk anticipation which clearly improves the risk assessment in complex and large traffic scenarios. Later, modeling relying on octagonal representation of the surrounding space is proposed by [123] as an analytic approach to assess the collision risk with obstacles. They used two probabilistic methods to calculate the risk: the Collision State Probability (CSP) in real-time and the Collision Event Probability (CEP) density. They developed the octagon concept which models the trace of the obstacle centroids when it moves around the ego-vehicle (represented by a rectangular box). This concept provides a spatial multidimensional safety indicator.

An alternative approach is provided by grid-based solutions, where the estimation of the collision risk can be computed for each cell regardless of the object it belongs to. This grid-based approach is for instance used in the Conditional Monte Carlo Dense Occupancy Tracker (CMCDOT) [124]. Directly estimating the velocity of each cell in the grid, this algorithm predicts the position in the near future of every cell as well as the trajectory of the ego-vehicle. These estimations are computed over short periods, and a probability of potential collision is provided. TTC probabilities are then associated to the cell from which the colliding element came from. In

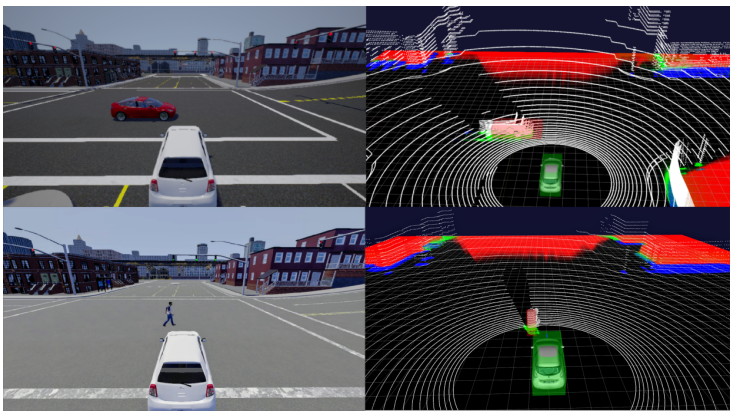

Figure 37: Left: Simulated collision scenarios reproduced in CARLA with the ego-vehicle (white) colliding with another vehicle (red) and with a pedestrian; Right: Simulated LiDAR sensor and corresponding occupancy grid output from the CMCDOT [15].

CMCDOT, the TTC probabilistic estimations are discretized corresponding to collisions within 1, 2 and 3 seconds. This strategy, originally presented in [125], enjoys the advantage of being model-free, avoids solving the complex problem of multi-object detection and tracking, while integrating the totality of the available information and providing a probabilistic estimation of the risk associated to each part of the scene. The result of this estimation can then be used as the starting point for any control and decision-making system. Though in [125] the authors have presented an evaluation of their approach, a proper validation is absent and poses a big challenge.

To try solving this problem, [15] proposes a methodology based on Statistical Model Checking (SMC) [126]. The SMC approach makes use of specifically defined KPIs, expressed as temporal properties depending on a set of identified metrics. By analyzing the behavior of these metrics through both real experiments and numerous simulations in realistic environments generate in CARLA [127] (see Fig. 37), a probability for the system to finally respect the KPIs is provided by the model checker. The validation involves both positive cases (the collision is real unless actions are taken) and negative cases (the collision is not going to happen). The informal description of the KPI is that the high probability of collision predicted by a perception system should be followed by a real collision, while low probability of collision should guarantee its absence. Formally, the KPIs are expressed as Bounded Linear Temporal Logic (BLTL) formulas as follows:

- $G_{\leq L}\ ((F_{\leq t}\ collided) \Rightarrow (cmcdot\_risk_i > \tau_h))$. This property states that, at any time, a collision happening within t seconds must be perceived by CMCDOT with a high collision risk. Any violation of this property for $t \leq i$ falls into false negative case: a real collision is not perceived in time.

- $G_{\leq L}\ ((G_{\leq t}\ \neg\, collided) \Rightarrow (cmcdot\_risk_i < \tau_l))$. This property states that the collision risk estimation must be low in all states such that no collision will happen in t seconds. A violation of this property for $t \geq i$ falls into false positive case: a non-existent collision is predicted.

where: i represents the seconds before the potential collision at the time we evaluate the collision risk; $cmcdot\_risk_i$ is the maximum probability of collision in i seconds output by CMCDOT among all cells belonging to the approaching vehicle; the thresholds τ_h and τ_l are the

boundaries when the collision risk is considered high and low, respectively; L is the number of states in the longest trace; the Boolean variable $collided$ is $True$ in all states after a collision has occurred and $False$ otherwise. The temporal operators G and F are: *always* ($G_{\leq t}\ \phi$), requiring ϕ to hold in all states for the next t units of time, and *eventually* ($F_{\leq t}\ \phi$), requiring ϕ to become true within t units of time. In [128], a different approach based on formal verification was adopted to analyse the same probabilistic collision risk estimation generated in simulated scenarios by the CMCDOT. In this case, a number of typical temporal properties (invariants, safety, liveness) were verified on each trace obtained in the CARLA simulator by using the XTL [129] model checker of the CADP toolbox [130], producing quantitative verdicts (sets of events violating the properties, with their diagnostic information). However, the scenarios considered to conduct this study are not fully representative of the entire configuration space and an automatic scenarios generation approach with a rigorous coverage analysis would make these studies more solid.

Going beyond the classic concept of collision risk, the authors of [131] proposed a new approach to quantify the risk of injury issued by the accidentology of each class of object present in the scene. Each class of object presents an injury probability with respect to the impact speed and ethical and/or economical factors. This method generates a cost map containing an estimation of the collision probability along with the risk of injury.

In [132], some new notions of cooperative risk and global risk have been proposed in autonomous driving context and with CAV. In this case, the risk assessment algorithm has the capability to access to the remote information coming from the other vehicles and road components.This also addressed impact on both the car and the driver. Recently, Leroy, Gruyer, and Orfila [17] have proposed five key components-based risk indicators ontology for the modelling and identification of critical interaction between human driven and automated vehicles (see figure 40). This work investigated a new ontology of risk for interaction between autonomous vehicles and human driven vehicles (mixed traffic configuration) based on a five key components decomposition. The improvements of this ontology compared to previous ones are a more detailed classification of risks according to the different components attributes. But the most important is the more consistent decomposition that improves the classical DVE (Driver Vehicle Environment) one by adding the road and the obstacles components. These improvements enable other AV functions to operate in better conditions (trajectory planning) but also a better estimation of global risks indicators. Moreover, this ontology is also a new way of describing risk estimation structures that can be used as a pedagogic tool. Extended from this work, the same authors have proposed a new adapted risk indicator for autonomous driving system with uncertainties and multi-dimensional configurations modeling called RIMU [133]. This risk estimator, on which relies AV decision-making, is based on an extended version of the distance of Gruyer (DG). This estimator provides an answer and a solution to the risk assessment needed as a part of a generic and extended architecture dedicated to the building of a generic driving meta-model usable for multi-modal driving behavior simulation (personal vehicle, connected vehicle, connected and automated vehicles and autonomous vehicle). The proposed estimators have been tested, evaluated, and analyzed on a set of representative highway scenarios with three key performance indicators. Results show that the proposed risk estimator (RIMUM) is more realistic, extended DG more reversible.
In [134], an overview of the main surrogate and safety indicators are presented:

- Temporal proximal indicators
  - Time to collision (TTC)

  - Extended Time to Collision (TET,TIT)
  - Time Exposed Time-to-Collision (TET) indicator
  - Time Integrated Time-to-Collision (TIT) indicator
  - Modified TTC (MTTC)
  - Crash Index (CI)
  - Time-to-Accident (TA)
  - Time Headway(TH)
  - Post-Encroachment Time(PET)

- Distance based proximal indicators
  - Potential Index for Collision with Urgent Deceleration (PICUD)
  - Proportion of Stopping Distance (PSD)
  - Margin to Collision (MTC)
  - Difference of Space distance and Stopping distance (DSS)
  - Time Integrated DSS (TIDSS)
  - Unsafe Density(UD)
  - Deceleration based indicators
  - Deceleration Rate to Avoid a Crash(DRAC)
  - Crash Potential Index (CPI)
  - Criticality Index Function (CIF)

- Other indicators
  - Jerks: The term Jerks is a composite of g-force and speed or a derivative of acceleration. This indicator was used in order to assess the tendency for left/right g-force and mean speed to predict bus accidents.
  - J-value: J-value is an accumulative safety indicator related to the accumulation of risk of vehicles inside a platoon.
  - Standard Deviation of Lateral Position (SDLP): This measure is similar in nature to the degree of vehicular control a driver exerts in any particular driving situation and is allegedly related to the probability of running-off the road. This indicator is mainly suitable for driving simulator or instrumented vehicle studies i.e. naturalistic driving approach.
  - other: Finally, there are several other safety indicators that are less widely used, such as Critical Gap, Gap Time (GT),Time to Intersection(TTI), Time to Stop Line (TSL), Time to Line Crossing (TLC),Predicted Minimum Distance (PMD); ODCA & PDCA, Potential Energy (PE) ...

In [135], authors have used a part of these safety indicators in order to evaluate the usability of surrogate measures at different road geometries(intersections, round-about ...). In [16], authors propose a probabilistic driving risk assessment framework based on intention identification and

risk assessment of surrounding vehicles. Firstly, we set up an intention identification model (IIM) via long short-term memory (LSTM) networks to identify the intention possibility of the surrounding vehicles. Then a risk assessment model (RAM) based on the driving safety field is employed to output the potential risk. The last stage consist to integrate risk evaluation model combining both IIM and RAM in order to build a dynamic potential risk map considering multi-vehicle interaction. In a typical but challenging lane-changing scenario, an automated vehicle with this type of safety indicator can assess its driving status by calculating a risk map in real time that represents the risk generated by the estimated intentions of surrounding vehicles (see figure 38).

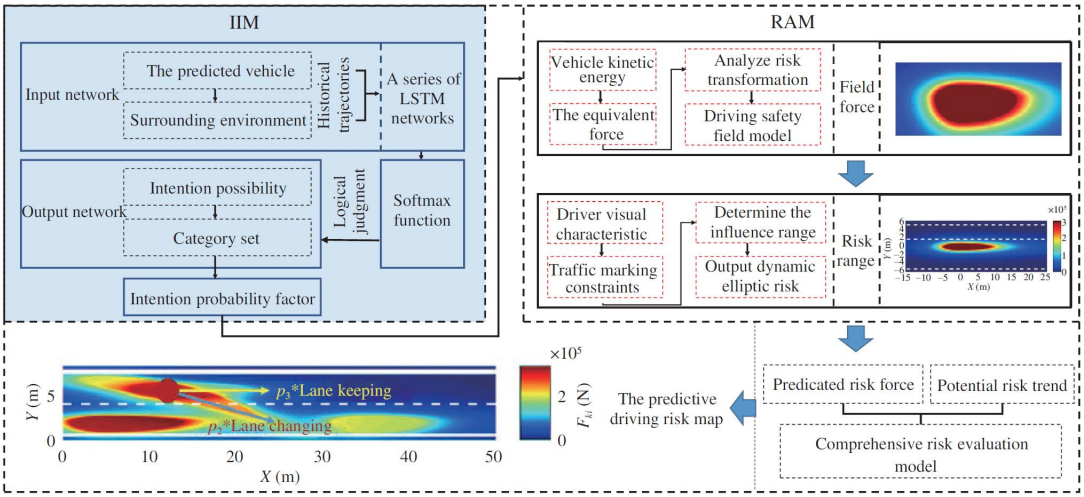

Figure 38: Framework of the comprehensive risk evaluation model. Combination of both Intention Identification Model (IIM) and Risk Assessment Model (RAM) to build a predictive risk map quantifying the potential risk.([16])

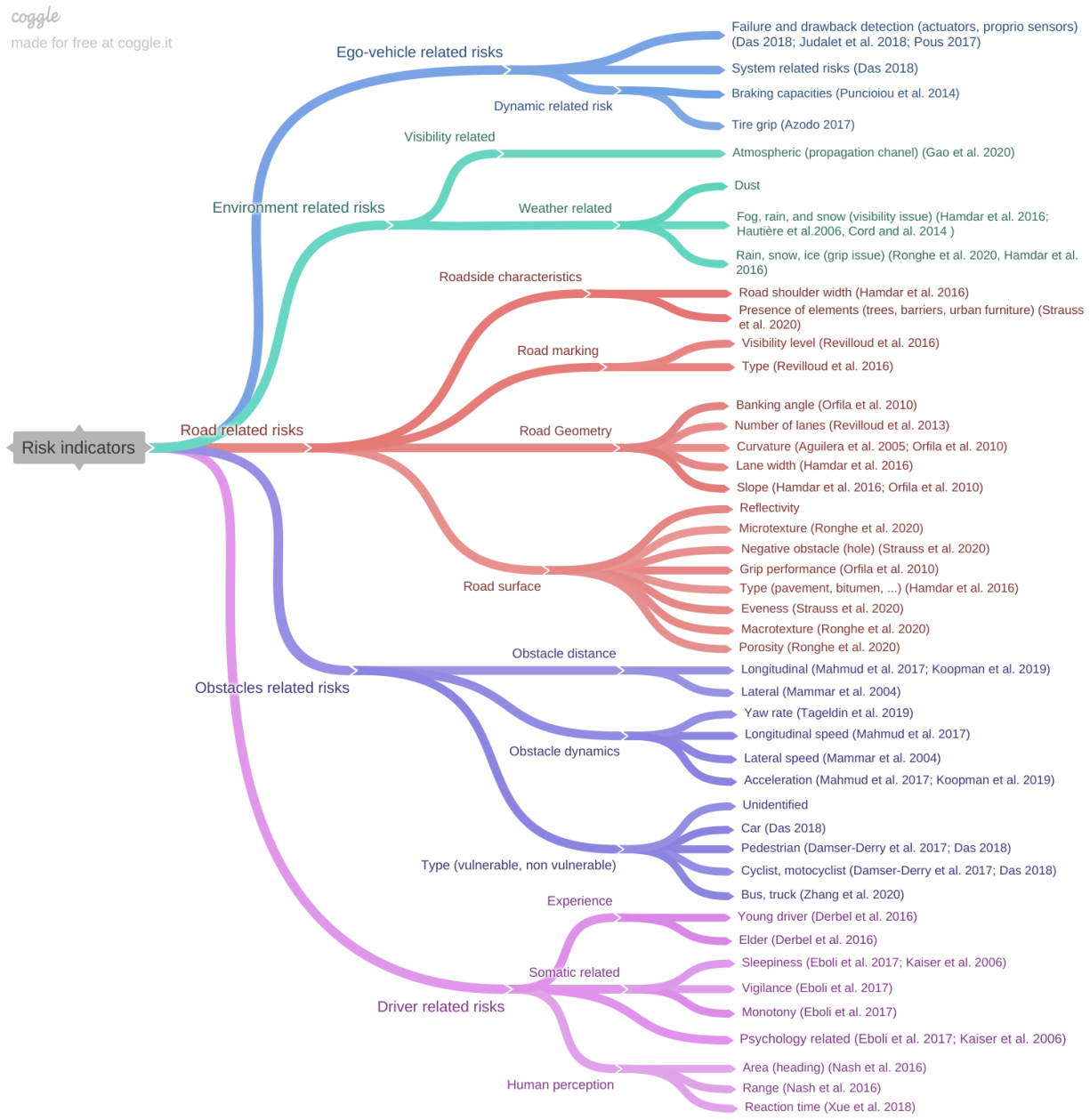

Figure 39: Ontology for risk indicators, key components and perception attributes [17]

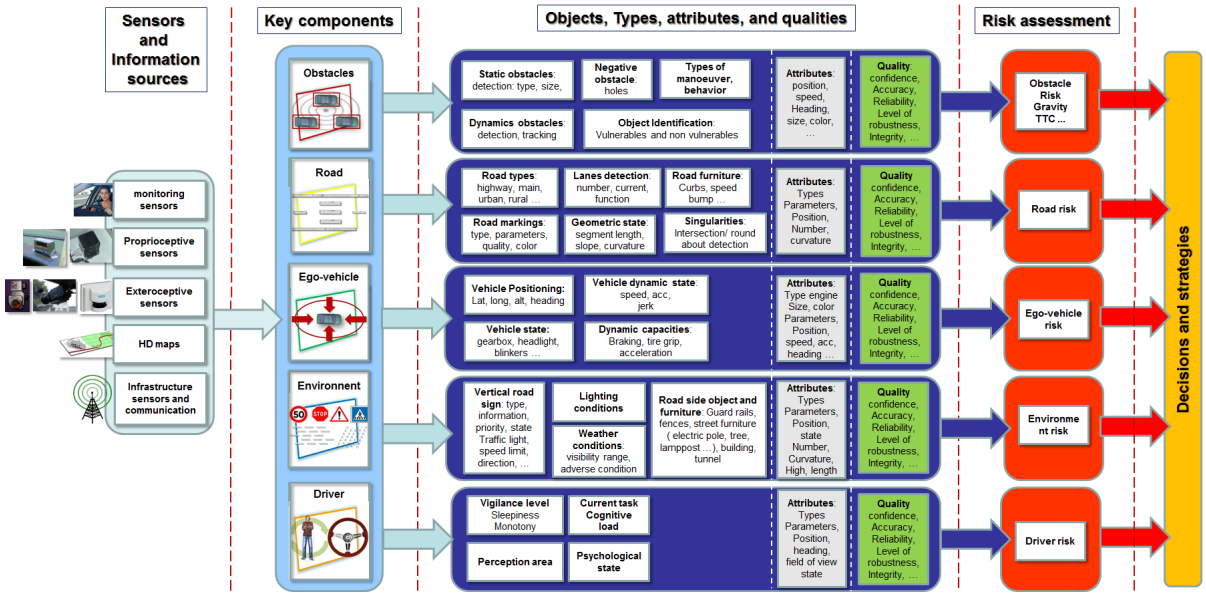

Figure 40: The different level of risk assessment depending of road key components (obstacle, road, ego-vehicle, environment, and driver) (source: University Eiffel)

Another interesting way using risk-index based sampling method is proposed by [136]. In this work, the authors propose a novel framework to generate scenarios for the evaluation of autonomous vehicle safety. The test scenarios are generated by sampling parameters from a probabilistic model based on naturalistic driving data, which consist of vehicle kinetic information and a traffic risk index. The proposed framework also allows to estimate the comprehensiveness of the test scenario with respect to the naturalistic driving dataset. In this work, the idea is not to generate scenarios and then assess the level of risk, but the opposite. The authors generated scenarios with constraint to reach a level of risk.

3.3 Metrics and evaluation operators for components and functions (i.e. Perception layers and modules)

In order to validate a perception application, it is necessary to use 3 sets of criteria. The first set contains the qualitative criteria, the second set the quantitative criteria, and the last one the semantic criteria. We mainly use these 3 sets of criteria in order to evaluate and validate the quality of a perception system using control theory approach or AI-based approach for road key components estimation. However, these criteria are applicable to cooperative systems, with extended and distributed perception.

- **Quantitative** indicators
- **Qualitative** indicators
- **Semantic** indicators

Moreover, in these 3 levels (Quantitative, qualitative, semantic), a dedicated vocabulary is used for Perception, AI-based, data fusion algorithms evaluation and validation. This algorithms, methods, and functions involve to merge information from multiple sources to improve decision making. The merged information can be direct observations, treatment results, generic knowledge (in the form of a rule for example) and be of a numerical or symbolic nature. The

level of merge depends on the merged information. The merger is said to be “low level” if values directly resulting from a measurement system are taken into account. The merging of processing results is said to be “medium level” and the merging of symbolic rules or parameters is said to be “high level”. Data fusion and processing explicitly considers the merged data to be necessarily ”imperfect”. The community recognizes that there is no consensual definition of these imperfections. However, some features of the imperfection are distinguished:

- **Incompleteness** characterizes the absence of information provided by the data source on certain aspects of the problem.
- **Ambiguity** is the capacity of information to lead to two different interpretations.
- The notion of **conflict** characterizes the fact that several different pieces of information lead to contradictory interpretations.
- The characteristic **redundancy** of data sources which each provide the same information and the characteristic **complementarity** of sources which provide information on different aspects.
- **Uncertainty** relates to the truth of information and characterizes its **degree of conformity** with reality.
- **Accuracy** concerns the content of the information and measures a quantitative lack of knowledge on a measure. It testifies to the lack of accuracy in quantity, size, duration of the data. We insist on the need not to confuse uncertainty and inaccuracy.
- **Integrity** is more complex to define. The definition of ”high level”, given by the ICAO (International Civil Aviation Organization) is: *”A measure of the confidence that can be placed in the accuracy (metrology definition) of the information provided by the system. It includes the possibility for the system to warn the user in time with valid alert messages”*. More explicitly, we can say that integrity is the (almost) certainty of not continuing to use a system that has become dangerously false. We can easily understand his interest in landing maneuvers, for aeronautic domain.
- **Disponibility - Availability** characterizes the percentage of time during which the tracking system is actually available with its nominal performance. For a perception system or a specific sensor, periods of unavailability in a specific/ constraining/ degraded/ adverse environment are mainly due to too few visible, readiness, and readable data/information.
- **Continuity (of service)**: probabilities that the defined performance of the system (in terms of accuracy and integrity) will be maintained for a given period of time, defined in advance, with the assumption that the service is operational at the start of the period.

3.3.1 Quantitative indicators

The key problem in comparing estimators is the choice of criteria and their interpretation regarding the evaluation of algorithm performance. In this section, we present a protocol for comparing estimators for local perception based on three families of evaluation criteria:

- 1. the precision [Drummond 98], [Li 01], [Drummond 99] is evaluated from the following three measurements:

  - The root-mean-square error (RMSE);
  - The average Euclidian error-AEE (Average Euclidian Error);
  - and finally the geometric average error-GAE (Geometric Average Error);

- 2. The consistency and credibility of each estimator are assessed using three criteria based on the standard of estimation error provided by each algorithm. It is:
  - The NEES (Normalized Estimation Error Square);
  - The NIS (Normalized Innovation Squared);
  - The ANEES (Average Normalized Estimation Error Square);
  - and finally the NCI (Non-Credibility Index) [Drummond 98], [Li 02], [Drummond 99], [Bar-Shalom 93], [Bar-Shalom 83].

  Other criteria based on the information content [Lefebvre 04], [Zamora-Izquierdo 08] of each filter, through 2 or 3σ envelopes, as well as uncertainty ellipses are presented;

- 3. the robustness of the estimators is evaluated by using statistical averages of algorithm parameters (ego-location, detection, etc.) on particularly aberrant point measurements, or during driving scenarios in borderline situations.

The following notations are adopted in this part. X is the exact state vector of dimension n to be estimated, its estimate is $\hat{X}$ and the estimation error $\tilde{X} = X - \hat{X}$. M defines the number of measurements. The exact mean squared error (MSE) variance-covariance matrix of $\hat{X}$ is denoted by Σ. The one provided by the estimator will be called P. During simulations, we assume that neither the evolution model nor the sensors used are biased. For each scenario, the total number of independent trials using a Monte Carlo random draw (having N samples each) is denoted by M, the index i therefore represents the ith trial. Finally, the Euclidean norm of a vector a will be denoted $\left\|a\right\|_2 = \sqrt{a^t a}$, where a^t is the transpose of the vector a.

3.3.1.1 Root Mean Square Error - RMSE [accuracy]

This is the best-known precision measurement. It is defined by:

$$RMSE(\hat{X}) = \sqrt{\frac{1}{M}.\sum_{i=1}^{M}\left\|\tilde{X}_i\right\|_2^2} = \sqrt{\frac{1}{M}.\sum_{i=1}^{M}\tilde{X}_i^t.\tilde{X}_i} \tag{3}$$

This quantity can be interpreted as the statistical parameter called “standard deviation”. For unbiased scalar parameter estimators, this is the most natural approximation of the standard deviation of the estimation error. Therefore, the RMSE is a useful parameter for probabilistic analyzes, although not having a simple physical interpretation. The popularity of RMSE stems mainly from the fact that it is generally the best approximation over a finite sample, the standard error. This is the most widely used optimality criterion in terms of errors (MSE), and remains mathematically the most elaborate.

3.3.1.2 Average Euclidean Error - AEE [accuracy]

The Average Euclidean error, denoted AEE, is defined by:

$$AEE(\hat{X}) = \frac{1}{M}.\sum_{i=1}^{M} \left\|\tilde{X}_i\right\|_2 = \frac{1}{M}.\sum_{i=1}^{M} \tilde{X}_i^t.\tilde{X}_i \tag{4}$$

This measurement is derived from the concept of Euclidean distance or Euclidean norm and is often referred to as Mean Absolute Error (MAE). While the RMSE is an approximation of the standard error, the AEE represents the average of the actual errors.

3.3.1.3 Geometric Average Error (GAE) [accuracy]

Both RMSE and AEE are based on an arithmetic mean of the error. However, it is known that this average is largely influenced by large error values, this influence being however weaker with the AEE than with the RMSE. To overcome this weakness, it is proposed to use a geometric mean: the GAE. The mean geometric error is defined by:

$$GAE(\hat{X}) = \left[\prod_{i=1}^{M} \sqrt{\tilde{X}_i^t.\tilde{X}_i}\right]^{1/M} \tag{5}$$

In order to optimize its numerical calculation, we go through the logarithm:

$$GAE(\hat{X}) = 10^{LGAE(\hat{X})} \text{ with } LGAE(\hat{X}) = \frac{1}{2.M}.\sum_{i=1}^{M} log_{10}\tilde{X}_i^t.\tilde{X}_i \tag{6}$$

3.3.1.4 NEES (Normalized Estimation Error Square) - [consistency and credibility]

This test is carried out by calculating the NEES (Normalized (state) Estimation Error Squared), denoted $\epsilon = \tilde{X}.P^{-1}.\tilde{X}^t$ Under the assumption H_0 that the filter is consistent and that the error $\tilde{X}$ follows a Gaussian law of variance-covariance matrix P, then ϵ follows a distribution of χ^2 with ηX degrees of freedom (ηX being the dimension of X). Then according to the equation giving the expectation of a random variable according to the law of χ^2

$$E[\epsilon] = \eta X \tag{7}$$

The test then consists of evaluating the acceptability of the equality to this equation. As we said before, the evaluation of the filter consistency from the statistic ϵ (the NEES) is based on simulations. Using the Monte Carlo approach with M independent samples each having the same number of elements N, we generate at each time index k, M independent values of the random variable $\epsilon(k)$, denoted $\epsilon_i(k), i = 1, 2, ..., M$. For these M samples, the mean value of the NEES is

$$\bar{\epsilon}(k) = \frac{1}{M}.\sum_{i=1}^{M} \epsilon_i(k) \tag{8}$$

It follows then that $M.\bar{\epsilon}(k)$ follows, under the hypothesis H_0, a distribution of χ^2 with $M\eta X$ degrees of freedom. Consequently, the assumption with the equation $E[\epsilon] = \eta X$ which translates the fact that the real estimation errors are consistent with those provided by the filter (criterion

C1), is acceptable if $\overline{\epsilon}(k) \in [r_1, r_2]$ where the confidence (or acceptability) interval [r1, r2] is determined with

$$P[\overline{\epsilon}(k) \in [r_1, r_2]|H_0] = 1 - \alpha_\theta \tag{9}$$

In this study, we have to choose the desired confidence level (eg at $1 - \alpha_\theta = 95\%$). We then refer to the table of values of χ^2 to determine the interval $[r1, r2]$. For instance, if we consider $\alpha_\theta = 5\%$, $\eta X = 2$ (position vector) and $M = 20$ independent Monte Carlo tests, i.e. 40 degrees of freedom, then we deduce from the χ^2 table that the interval delimiting 95% confidence probability is $[r1, r2] = [24.4/M, 59.3/M] = [1.22, 2.96]$. When $M = 1$ (one test, 2 degrees of freedom), this interval becomes $[0.05, 7.38]$. The interval is reduced with a large number of Monte Carlo trials, illustrating a reduction in the variability (error) when the trials are repeated, while remaining independent.

3.3.1.5 NIS (Normalized Innovation Squared) measurement

Criterion C2 is applied when the ground truth is known only from observations (often of the GPS coordinate type). This is particularly the case during tests on real tracks. The consistency test can therefore be performed using a new statistic called the NIS (Normalized Innovation Squared) [Bar-Shalom 93], and noted ϵ_v, such that $\epsilon_v = v^t.S^{-1}.v$ where ϵ_v represents innovation (also called the residue) of the filter and S the variance-covariance matrix of this residue. In the same way as the NEES, under the assumption that the filter is consistent, the NIS ϵ_v follows a law of χ^2 with ηY (dimension of the vector of measures) degrees of freedom. From M independent Monte Carlo samples, having a NIS ϵ_v, we calculate the mean:

$$\overline{\epsilon}_v = \frac{1}{M}.\sum_{i=1}^{M} \epsilon_v^i \tag{10}$$

The test performed is then similar to that described in the previous part dedicated to the NEES measurement. The acceptance interval is determined by assuming that $M.\overline{\epsilon}_v$ follows a law χ^2 with $M\eta Y$ degrees of freedom.

3.3.1.6 ANEES (Average Normalized Estimation Error Square) - [consistency and credibility]

3.3.1.7 NCI (Non-Credibility Index) - [consistency and credibility]

The NCI (Non-Credibility Index) is presented as the most accurate universal indicator of the credibility of an estimator. In particular, it keeps the property of invariance with respect to the dimension of the estimator. Its calculation is based on the principle that the difference between P (variance-covariance matrix of the estimation error) and Σ (exact mean square error (MSE) variance-covariance matrix) is equivalent to the difference between P^{-1} and Σ^{-1}. A well known way to compare the matrices P^{-1} and Σ^{-1} is to compare the scalars $\tilde{X}^t.P^{-1}.\tilde{X}$ and $\tilde{X}^t.\Sigma^{-1}.\tilde{X}$; where $\tilde{X}$ is the estimation error. A classic approach used to quantifying the difference between $\tilde{X}^t.P^{-1}.\tilde{X}$ and $\tilde{X}^t.\Sigma^{-1}.\tilde{X}$ is done through the variable D defined by:

$$D = (\tilde{X}^t.P^{-1}.\tilde{X}) - (\tilde{X}^t.\Sigma^{-1}.\tilde{X}) \tag{11}$$

3.3.1.8 Comparison and semantics of errors measurements

Comparison and semantics of accuracy measurements aspect

The EEA expresses the average distance between the estimate and the parameter to be estimated. In our applications, this distance will be expressed in our physical (Euclidean distance) and Cartesian space. If we consider a test set containing a sub-sample of 10 estimation errors noted in meters: $[1.5; -2.1; 2.3; -2.5; 3.1; 6.1; -2.0; 1.7; 0.5; -1.8]$, we will then obtain an AEE of 2.36m and an RMSE of 2.74m. The EEA expresses the mean error in amplitude well. However, it is not easy to give a physical interpretation of RMSE apart from its relevance in probabilistic analyzes, and in particular the characterization of the dispersion of errors. For this same error and assuming the unbiased estimator, the AEE appears to be the natural approximation of the mean deviation from the reference $E[\tilde{X}]$, for a finite sample. And for a Gaussian distribution of standard deviation σ. Therefore, AEE can be converted to RMSE if it is assumed that the error follows a Gaussian distribution, while RMSE cannot be converted to AEE without precise knowledge of the distribution of the error. For these reasons Li [Li 01] recommends, for the evaluation of the precision of estimators, to use the AEE instead of the RMSE.
The GAE error, based on the geometric mean, is more robust than these first two measurements (RMSE and AEE). Indeed, it is less sensitive to large values of real deviations. On the other hand, the GAE is never greater than the AEE, itself never greater than the RMSE value. Thus we theoretically have $GAE \leq AEE \leq RMSE$: the impact of large errors is greater on an RMSE and less important on a GAE.
If we consider the example of the ten measures of error, then we obtain a GAE of 2.004m; the theoretical order between the different averages is well respected. The main advantage of using GAE is its ability to be less influenced by very large error values, while remaining close to AEE. Equality between these errors only exists when the number of tests is limited to 1 ($M = 1$), this is the case for tests on real tracks. Also, the difference between GAE, AEE and RMSE can be used to represent the presence of large point errors. The closer these averages, the closer the errors along the path (no error peak). In the case where $GAE = AEE = RMSE$, then there is no fluctuation in these errors and the common error can be considered as a bias, and hence this bias can be compensated.

Consistency of an estimator

In many perception algorithms, we have to use state estimators. In order for these estimators to be considered consistent it is necessary that the estimation errors satisfy the two conditions presented in [Bar-Shalom 93] and [Lefebvre 04]:

$$E[X - \hat{X}] \equiv E[\tilde{X}] = 0 \tag{12}$$

and

$$E[(X - \hat{X}).(X - \hat{X})^t] \equiv E[\tilde{X}.\tilde{X}^t] \leq P \tag{13}$$

This means that for a finite number of samples (measurements), the estimation error must be consistent with the following theoretical statistical properties:

- **C1**: the real errors on the state of the system must be centered at zero and have a variance-covariance matrix greater than the real MSE of the system;
- **C2**: innovation must also satisfy this property.

Among these two criteria, only C2 can be tested with real data. Criterion C1, which is really the most important, can only be tested in simulation because we need a well-known reference to implement it. Then the ground truth availability is essential.

Comparison of NEES and NIS

From the results that will be obtained, we can deduce that if the error statistic (NEES or NIS) is beyond the upper limit of the interval $r2$, then:

- either the estimate is biased
- either the real errors are very large compared to those estimated by the filter
- or the covariance of the error provided by the filter is too low.

If, on the contrary, the calculated statistic is below the lower bound of the interval $r1$, then the covariance of the estimation error provided by the filter is too large compared to the real error. When the error statistic is within the acceptability range, then it is concluded that the filter is configured correctly, and the estimated error is consistent with the actual errors in the system.

3.3.1.9 Quantitative indicators and combinatorial science NEW

Most ML/DL applications continue to incorporate traditional measures such as MAE or RMSE as primary indicators of the adequacy of ML models (mainly those based on regression). This seems to stem from familiarity with these measures, as opposed to others, such as the Golbraikh and Tropsha criterion [137], the QSAR model of Roy [138], Frank and Todeschini [139], and objective functions specifically designed following the domain. The work of Gandomi et al [140], Golafshani and Behnood [141] and Cheng et al [142] has applied a multi-criteria verification process that integrates the use of traditional and modern measures. The use of a multi-criteria process is not only beneficial for ensuring the validity of a particular ML model, but is also recommended for overcoming some of the identified limitations of traditional measures. For example, the actual/predicted correlation, which corresponds to the linear correlation (in the statistical sense) between actual and predicted values for each value considered in a set, should be used in conjunction with another measure such as RMSE, MAE or Max Error.

3.3.2 Qualitative indicators

This first level of qualitative criteria makes it possible to obtain statistics on the general functioning of the algorithms and is mainly concerned with qualifying the overall performance of the detection algorithms and not its quantitative precision.

3.3.2.1 Simple quality metrics

The simplest and often used metrics are:

- **The detection rate** which is the number of objects detected divided by the number of objects to be detected. It is enough that the object is detected on at least one time of the scenario for it to be considered as detected in the tests. Objects may as well vehicles as road markings or other objects.

- **The detection distance** which indicates the average distance from which an object is detected. Often this criterion is associated with a detection percentage (80% detection).
- **The loss of detection**. From the moment the object is detected, this criterion will represent the sum of the duration of the detection losses divided by the duration that the detection should have had. An average of this error is provided as a result.
- **The false detection rate** which is the number of times at least one false detection was recorded divided by the total number of iterations of the detection algorithm. An average of this error is provided as a result.

3.3.2.2 Precision and recall

In order to define this precision and recall metric, it is first of all necessary to define 4 metrics (originally they are defined for classification issues):

- **TP (true positives)**: which represents the number of objects of a specific class detected with a positive test and actually belonging to that class. Correct detection situation. For instance, in a situation with 2 classes (obstacles / non-obstacles), for an obstacle detector, this value represents the number of cases of detection of an obstacle by the algorithm, actually corresponding to an obstacle.
- **FP (false positives)**: represents the number of objects of a specific class detected with a positive test when they are not part of that class. False alarm situation. For instance, in a situation with 2 classes (obstacles / non-obstacles), for an obstacle detector: number of cases of detection of an obstacle by the algorithm, when in fact it is not an obstacle.
- **FN (false negatives)**: represents the number of objects of a specific class not detected with a negative test even though we are in the presence of an object of this class. Non-detection situation. For instance, in a situation with 2 classes (obstacles / non-obstacles), for an obstacle detector: number of cases of non-detection of an obstacle by the algorithm, while an obstacle is present.
- **TN (true negatives)**: represents the number of objects of a specific class considered undetected with a negative test and not actually being an object of that class. For instance, in a situation with 2 classes (obstacles / non-obstacles), for an obstacle detector: number of cases of non-detection of an obstacle by the algorithm, and where there is indeed no obstacle.

Evaluating each of the previous metrics requires ”ground truth”, that is, having assigned the data and those to which they correspond to each scene of the scenario. For instance :

- for an obstacle detection algorithm from LiDAR data: 3D zones / points, with ”obstacle” / ”no obstacle” labels.
- for an obstacle detection algorithm from an image (from a camera): 2D / pixel area, with ”obstacle” / ”no obstacle” labels

These 4 metrics define the confusion matrix. From these different metrics, it is possible to define the recall and the precision:

- **The recall** is defined, for a binary detection (example: obstacle / no obstacle), as TP/(TP+FN). Thus, the recall is defined by the number of objects of the target class detected according to the number of objects actually present and detectable of this class during the scenario unfolded. A high recall corresponds to a high detection of all objects of the target class that exist in the scenes of the scenario. Conversely, a weak recall corresponds to a weak detection of all objects of the target class in these scenes.

- **The precision** is defined for a binary detection (example: obstacle / no obstacle) as TP/(TP+FP). Precision is the number of objects of the target class detected in relation to the total number of objects returned by the detection algorithm. The principle is as follows: when an algorithm performs a detection, we want the detected objects to correspond to the targets actually present in the scene. Any superfluous or irrelevant detected objects constitute noise. Precision opposes this detection noise. If it is high, it means that few unwanted objects (different from the target class) are offered by the detection system. Conversely, if it is low, many objects detected as a target class are in fact objects that do not belong to this target class.

An ideal detector would have an accuracy and recall of 1:

- recall of 1 (100%): all target objects are detected

- precision of 1 (100%): all detected objects are indeed only target objects

It is generally trivial to create a recall detector = 1: it suffices to create a detector which predicts all objects in the scene as a target class (example: a detector which indicates that all objects in the scene are obstacles, everything the weather). Likewise, for a precision detector = 1. What is difficult to achieve, however, is good precision with a good recall. The quality of a detector is therefore generally evaluated via a metric defining a balance between precision and recall (see F-Score).

The precision / recall value depends on the detector threshold, that is to say the numerical value α which makes it possible to define the separation of the intervals indicating the detection of an object of the target class ((for example = $[\alpha, +\infty[$) or absence of the object of the target class (for example =$]-\infty, \alpha[$).

Moreover, these 2 metrics are generally also interdependent: when one is increased, the other risks being decreased after a certain monotonic change in threshold. These metrics are originally defined for classification issues, but they are adaptable to detection issues (see IoU).

3.3.2.3 Score-F

A popular measure that combines precision and recall is their weighting, called F-measure (or F-measure in English) or F-score. It is defined as follows:

$$F_\beta = \frac{(1+\beta^2).(precision.recall)}{(\beta^2.precision + recall)} \tag{14}$$

$\beta > 0$ allowing in this equation to adjust the importance of the recall (the greater the β, the more it is favored). It is possible to be satisfied with the F1 score which is defined as the

harmonic mean of the precision and the recall, when there is no preference between recall and precision:

$$F_1 = \frac{2.(precision.recall)}{(precision + recall)} \tag{15}$$

Just as recall and precision depend on the detector threshold, F_1 depends on this threshold. However, it is possible to assess what is the threshold to have the best compromise in terms of precision / recall and therefore of F_1.

3.3.2.4 The distance of Sørensen

This measure of similarity on sets proposed by Sørensen [Sørensen 57] has been adapted in order to use the information that we construct from the quadruplet VP, VN, FP, FN. Sørensen's initial equation consists of taking the number of elements shared over 2 classes and normalizing it by the sum of the elements of the 2 classes:

$$DSC = \frac{2.\left|A \cap B\right|}{\left|A\right| + \left|B\right|} \tag{16}$$

By taking the confusion table we can extract from it 2 sets of elements: the set of objects in the obstacle class (VP + FN), and the set of objects detected by the algorithm as being obstacles (VP + FP). If we apply these 2 sets to the similarity operator then we get the following equation:

$$DSC = \frac{2.TP}{(TP + FP) + (TP + FN)} \tag{17}$$

If we plot the $DSC = f(\alpha)$, the maximum of DSC is obtained for the best threshold α. In other words: $\alpha = argmax(DSC)$. We also want at the level of the DSC peak to have something with the least steep slope possible (that our detector is the least sensitive to the selection threshold chosen to work well). For example, it is possible to evaluate only the precision/recall and F_1 for this threshold α, in order to best compare the configurations of our algorithms (comparisons of the relative to the best thresholds).

The quality of the curve obtained with this DSC operator can be quantified through two criteria. Firstly, the maximum value of the curve which indicates the optimal value of the threshold (or of the configuration of the algorithm), and secondly the support of the curve which must be extended because it tells us about the sensitivity of the detection algorithm for the threshold value. However, these criteria are mainly concerned with qualifying the overall performance of the detection algorithms and not their quantitative precision. In addition, these criteria give a general a posteriori view of the quality of an algorithm. They do not allow us to study in more detail the causes of failure or degraded operation. They also do not identify problematic configurations or situations.

3.3.2.5 Intersection over Union (IoU)

Perception and detection algorithms generally perform a dual operation: they detect the presence of a certain type of target, and then they generate higher-level modeling including the state vector and uncertainty modeling. This modeling can take several forms. The most commonly used are variance / covariance and bounding boxes (2D or 3D). Thus, in order to define the metrics of TP, FP, TN and FN precisely, it is appropriate to give the definition of

these metrics in this case. Thus from the bounding boxes generated, we can evaluate the quality of the detection from a ground truth with a homogeneous modeling (bounding box). Intersection over Union is defined precisely as:

$$IoU = \frac{T(G \cap D)}{T(G \cup D)} \tag{18}$$

with

- G: surface or volume of the target (or object) in the ground truth
- D: surface or volume of the detected target (or object)
- T: function providing the size of a surface or a volume

Thus, each target object of the ground truth returns an IoU. This IoU must then be thresholded with a second threshold β, to consider a detection that is conserved or not (in addition to considering whether this is the correct class assigned), and we can come back to the definitions of TP, FP, and FN. This metric can be used with as well oriented bounding boxes as not oriented bounding boxes, or with any polyhedra. A precision-recall curve is obtained (itself defined by changing the threshold α) for each of the chosen β thresholds. This threshold β will therefore have to be fixed beforehand in order to define an acceptable localization relevance. The procedure to be followed will then be as follows (case of detection with a binary target in the case of an obstacle). For given thresholds α and β:

- for each of the detected obstacles, check if $IoU > \beta$ (to apply to each surface/volume of the ground truth)
  - If true: detection is kept. If the object is part of the target class then TP, otherwise FP (by the way we "mark" the surfaces/volumes associated with the TP)
  - If false: detection is not kept and nothing is done.
- Then we go through and count the possible unmarked ground truths: they define the FN.

3.3.2.6 Recognition indicator

In [143], a set of metrics and a single detection score per class has been proposed in order to evaluate the performance of an RGB-based detection model in the Cityscapes 3D benchmark. This single indicator is built from the merge of several metrics that individually assess recognition performance in terms of 2D and 3D detection and localization. The mean over all classes is denoted as mean detection score (mDS) and is used for ranking the approaches within the benchmark. These indicators are:

- **2D Average Precision (AP)**: This metric is the standard Average Precision (AP) for assessing the 2D detection performance based on 2D bounding boxes. The 2D bounding box of both, ground truth and predicted 3D box, is defined as the circumscribing rectangle of all 8 vertices of the 3D box projected into the image. Matching is conducted in the 2D space and we require an IoU of 0.7 between ground truth and detection to accept the detection as true positive.

- **Depth-Dependent Average Precision (DDAP)**: This metric is the standard AP for all objects within the range $[s, s + \delta s]$. To assign a depth value to an object, it is necessary to take the ground truth depth for true positive and false negative detections and the predicted depth for false positives.

- **Depth-Dependent True Positive Metrics (DDTPM)**: This metric uses several depth-dependent true positive metrics, *i.e. BEV Center Distance, Yaw Similarity, Pitch-Roll Similarity, and Size Similarity*. The depth of the ground truth box is used to determine the applicable interval $[s, s + \delta s]$. In contrast to the regular 2D AP score, a fixed confidence threshold is used for the depth-dependent true positive metrics. The threshold cw is defined as $cw = argmax_{c\in[0,1]}p(c)r(c)$ with $p(c)$ and $r(c)$ denoting the precision and the recall for score c

- **Center Distance (BEVCD)**: Bird's-Eye View Center Distance is defined as the normalized integral of the depth-dependent distance up to the maximum depth of interest X_{max}

- **Yaw Similarity (YawSim)**: Following the same scheme than Center Distance metric, Yaw Similarity is inspired by [12]

- **Pitch-Roll Similarity (PRSim)**: Pitch-Roll Similarity is calculated analogously to Yaw Similarity but both pitch and roll orientation are combined since pitch and roll of a vehicle are not independent in realistic driving scenarios.

- **Size Similarity (SizeSim)**: Size Similarity assesses the 3D dimensions of the true positive detection.

the final Detection Score (mDS) is obtain by combining all quality measures presented above. The detection score per class is given with $mDS = AP.(BEVCD + YawSim + PRSim + SizeSim)/4$ By this definition, the 2D Average Precision is enforce to be an upper bound for the final detection score that can only be reached if all true positive bounding boxes are predicted perfectly.

3.3.3 Semantic indicators

The use of semantic indicators in the field of autonomous vehicles is a promising approach to enhance the understanding and decision-making capabilities of embedded artificial intelligence (AI) systems particularly in terms of human-machine interaction. Initially, this type of indicator was used mainly in the Natural Language Processing (NLP) field. These semantic indicators are elements that enable AI to comprehend the meaning and context of the information it perceives, which is essential for safe and effective interactions in complex and dynamic environments.

Here are a few areas where these indicators can be used in the field of autonomous vehicles:

- Object and Entity Recognition: Autonomous vehicles must be able to recognize and understand various objects and entities in their environment, such as pedestrians, other vehicles, traffic signs, etc. Semantic indicators associated with object recognition (based on classical quantitative metrics like FPR, precision, recall, MAP, IoU...) allow AI to categorize and understand these elements more finely, which is crucial for making safe navigation and driving decisions.

- Natural Language Understanding: Interactions with passengers, pedestrians, and other drivers may involve the use of natural language. Semantic indicators enable AI to understand and interpret these interactions, whether it's responding to voice commands, understanding the intentions of other road users, or communicating information to passengers. For this field of application, we can apply the classic NLP metrics:
  - Intent Recognition Accuracy: The percentage of correctly recognized intents or commands from user input.
  - Slot Filling Accuracy: For dialog systems, the accuracy of filling the slots with correct entities or values in response to user queries.
  - Word Error Rate (WER): The percentage of words incorrectly recognized in the transcribed text compared to the reference text.
  - Mean Reciprocal Rank (MRR): A measure of how well the system ranks the correct response in a list of potential responses.
- Human Behavior Analysis: Understanding human behavior is essential for predicting the future actions of pedestrians and other drivers. Semantic indicators associated with human behavior analysis allow AI to detect subtle signals such as gestures, facial expressions, and intentions, which is crucial for ensuring safe and predictive interactions. To evaluate this kind of indicator, quantitative and qualitative metrics can be used, such as:
  - Action Prediction Accuracy: The percentage of correctly predicted actions or behaviors of pedestrians or other drivers.
  - Reaction Time Estimation: Estimating the time it takes for the AI system to react appropriately to observed human behaviors.
  - Trajectory Prediction Accuracy: For predicting the future trajectory of pedestrians or other vehicles, metrics such as Root Mean Square Error (RMSE) or Average Displacement Error (ADE) can be used.
  - Sensitivity and Specificity: Sensitivity measures the proportion of actual positive cases that are correctly identified, while specificity measures the proportion of actual negative cases that are correctly identified.
- Contextualization of Environmental Information: Semantic indicators help contextualize environmental information perceived by the vehicle's sensors. For example, they can enable AI to understand that the presence of a bicycle near a bike lane is different from its presence on a highway, and adapt driving decisions accordingly. Here, we can use these kind of metrics:
  - Contextual Understanding Score: A qualitative or quantitative measure of how well the AI system understands the context of the environment based on the perceived information.
  - Semantic Segmentation Accuracy: The accuracy of classifying each pixel in an image into semantic categories, which can provide contextual information about the environment.
  - Scene Understanding Metrics: Metrics such as Scene Completion Accuracy or Scene Classification Accuracy, evaluating the AI system's ability to comprehend the overall scene and its elements.

3.4 Metrics and evaluation operators for simulated data, models and systems

The objective is not only to assess the quality of the perception functions, methods, application but also to quantify the level of representativeness of the data generated by the simulated sensors and by the simulated environment. The goal of this part is to enumerate the different concepts and metrics coming from the International Vocabulary of Metrology which be be used in order to evaluate the simulation platforms and the quality of the sensor models and the generated data. The goal in the sensor simulation or physical system modelling to have a perfect behavior, but to obtain a system modelling providing same data and behavior than the ones coming from the real similar system with the same defaults, imperfections, noises, weaknesses ...

3.4.1 Some definition about metrics concepts

Fidelity (ISO / DIS 3534 standard): “closeness of similarity/correlation/fitting between the results obtained by applying the experimental process several times under determined conditions. The process is all the more faithful as the RANDOM part of the experimental errors which affect the results is less”.

The Fidelity of the method will therefore be defined by two criteria: **repeatability** when the experimental conditions are identical and **reproducibility** when the experimental conditions are different. This aspect are really relevant in order to evaluated the data provided by simulation platforms.

Repeatability (Standard ISO / DIS 3534): ”closeness of similarity/correlation/fitting between successive results obtained with the same method on an identical material subjected to the test and in the same conditions ”, Under the same conditions, means that the determinations were made with the same operator, the same equipment, in the same laboratory and in a very short time interval. In simulation, this could mean that the experiments were carried out with the same scenarios, the same parameters, the same configurations, on the same computer.

Reproducibility (Standard ISO / DIS 3534): ”closeness of similarity/correlation/fitting between the individual results obtained with the same method on an identical material subjected to the test and in different conditions ”.

The metrological quality of a measuring instrument or of a measuring unit involving a sensor is the set of data which characterize the quality of the measurement carried out by the device in question. In simulation environment, we address the virtual sensor and the metrological quality of the data provided by this sensor. The main characteristics of virtual measuring instruments (or metrological properties of virtual measuring devices) are defined in the framework of the International Vocabulary of Metrology and include:

- **The error**: The measurement error is the difference between the “conventionally true” value or true value of a quantity and its measured value. Error characterizes accuracy. The smaller the error, the more accurate the instrument. Accuracy is characterized by 2 magnitudes of different kinds: **correctness** and **fidelity**.
- **The measuring range**: The measuring range is the range of measured values for a measurand, in which defined, agreed, or guaranteed error limits are not exceeded. It is delim-

ited by a lower and an upper measuring range limit that define the measuring span.

- **The resolution**: In metrology, the resolution is also synonymous of **uncertainty**, that is to say characterizes the reading error, for example due to the size of a needle or the number of digits or even noise of measurement. For this metric, we have an ambiguity between metrology community and data fusion and data processing community.

- **The sensitivity**: The probability that the detection will be done if an obstacle is really existing, is measured with the obstacles only. It is given by TP/(TP + FN). A sensitivity measurement is always accompanied by a specificity measurement. The sensitivity is also called the recall. In statistics, the sensitivity of a test measures its ability to give a positive result when a hypothesis is verified. It is opposed to specificity, which measures the ability of a test to give a result negative when the hypothesis is not verified.

- **The specificity**: The specificity, or the probability of obtaining a negative detection in non-obstacles, is given by TN / (TN + FP).

- **The uncertainty and resolution**: The uncertainty is an indicator used to describe a confidence interval (qualitative value) in which the reference value (ground truth) of the measurement device (sensor) has a probability (noted p) close to 1 of being there. Nevertheless, in metrology, measurement uncertainty addressed also the **resolution** and is also synonymous of resolution, that is to say characterizes the reading error, for example due to the size of a needle or the number of digits or even noise of measurement. For this metric, we have an ambiguity between metrology domain and data fusion and data processing Community.

- **The correctness or accuracy**: The correctness is expressed by the bias, which is the difference between the experimental mean of the series of measurements and the reference value.

- **The fidelity or precision**: Fidelity is usually expressed as standard deviation of the series of measurements, a significant index of the width of the dispersion. Fidelity is the capacity of a measuring instrument to give the same indication for the same measured value. Fidelity usually defines the dispersion of results (standard deviation of the series of measurements). If only one measurement is made, the precision represents the probability that it is representative of the average result. The latter would have been obtained by carrying out an infinity of measures. From a probabilistic point of view, if the measurement error is considered as a random variable characterized by its law of probability density, the trueness is represented by the deviation of its mean from the "conventionally true" value and the reliability is represented by its variance or standard deviation.

Sensitivity and specificity are two evaluation criteria which allow to produce a qualitative evaluation in the form of a graphical function of the efficiency of an obstacle or marking detector (also called the operating characteristic of the detector or the performance characteristic or even the sensitivity /specificity curve). This function, more frequently called the **ROC curve** (for Receiver Operating Characteristic) will make it possible to graphically study the variations in the sensitivity and the specificity of a detector for a set of values with a threshold having an influence on the quality of the result. This function is a measure of the performance of a binary classifier (obstacle / non-obstacle detector, marking / non-marking detector), i.e. a

system which aims to categorize entities into two distinct groups on the basis of one or more of their characteristics. Graphically, the ROC measurement is often represented in the form of a curve which gives the rate of true positives (sensitivity: fraction of positives that are detected (correctly)) as a function of the rate of false positives (fraction of negatives that are detected (incorrectly)) for this same group. More and more, these curves are used to know if a model is significant, to give an idea of the performance of a system and to estimate its reliability, to compare several models or several configurations in order to determine the best performing model or configuration according to a set of objectives. They are also often used in statistics to show the progress made with a binary classifier when the discrimination threshold varies. If the model calculates a score s which is compared to the threshold S to predict the class (i.e. $(s < S)$ gives positive evaluation and $(s >= S)$ gives negative evaluation), and we then compare the result with the real classes ("*Positive*" and "*Negative*"), the sensitivity is given by the fraction of the "*Positives*" classified as positive, and the anti-specificity (1 minus specificity) by the fraction of "*Negatives*" classified as positive. We put the anti-specificity on the abscissa and the sensitivity on the ordinate to form the ROC diagram. Each value of S will provide a point on the ROC curve, which will go from (0, 0) to (1, 1) and which will take the following meaning:

- **At (0, 0)** the detector always declares "*negative*": there is no false positive, but also no true positive.
- **At (1, 1)** the detector always declares "*positive*": there is no true negative, but also no false negative.
- **At (0, 1)** the detector has no false positives or false negatives. This means that the algorithm is perfectly correct and that it is never wrong.
- **At (1, 0)** the detector has no true negative or true positive. This means that the algorithm is completely inaccurate and that it is always wrong. In this case, by inverting the outputs of the algorithm, it is possible to obtain a perfectly accurate detection algorithm.

A detector giving random results will draw a line going from (0, 0) to (1, 1). By measuring the area under the ROC curve, we can then determine the quality of our algorithm. The larger this area, the more the curve deviates from the line of the detector producing random results and the closer we get to the "ideal" detector (which goes from (0, 0) to (0, 1) to (1, 1)). This curve is also used to estimate the optimal threshold value that can be used in a detection algorithm based on a set of parameters. This is to identify the configuration that maximizes the performance of an algorithm. The optimum threshold (or configuration) will be the one which corresponds to the point closest to the point (0,1).

The ROC curve is also interesting for the following reasons:

- Its construction is independent of the misallocation cost matrices.
- It also works in the case of very unbalanced distributions without suffering from the negative effects of the confusion matrix (need to make an allocation).
- It offers (as we have just seen) a graphical tool which makes it possible to evaluate the performance of a detection or classification system.

A device or a measurement system (a sensor) is qualified according to the values it gives when it is applied to a standard/reference/ground truth (value assumed to be true) under conditions of repeatability or reproducibility. From the figure 41 the different notion of fidelity and correctness are presented.

- **1**: The system is neither correct (correctness) nor faithful (fidelity) in the event of significant systematic and random errors.
- **2**: The system is faithful (fidelity) but not correct (correctness) in the case of low random error (uncertainty) and large systematic error.
- **3**: The system is correct (correctness) but not faithful (fidelity) in the case of negligible systematic error in front of a large random error (great uncertainty).
- **4**: The system is correct (correctness) and faithful (exact) in the event of negligible systematic error and low random error (uncertainty).

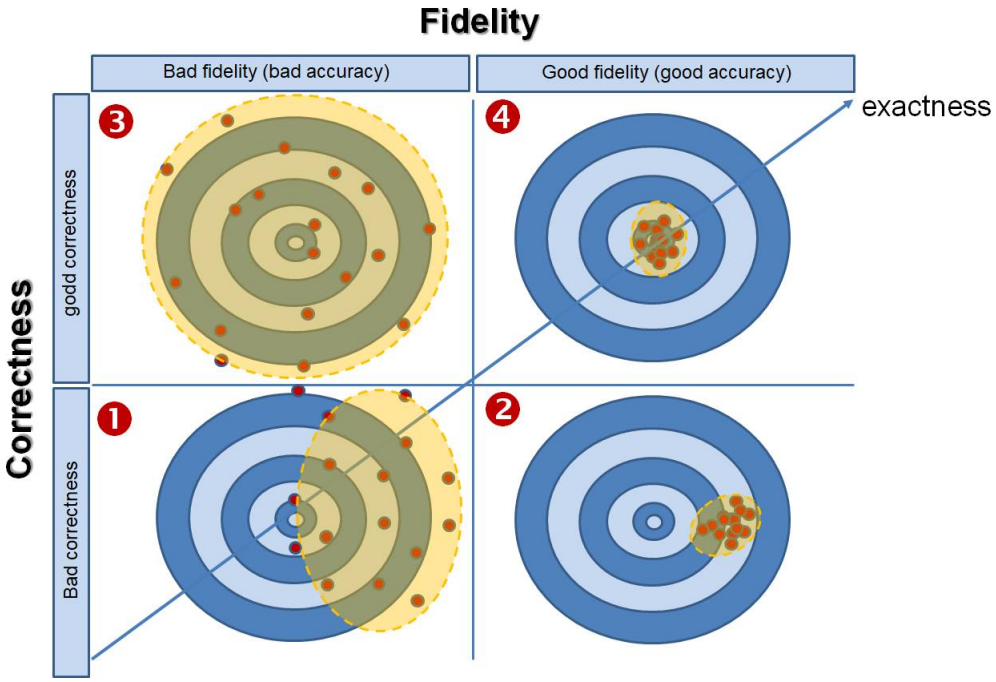

Figure 41: Correctness, fidelity, and exactness

The exactness is the combination of the fidelity and the correctness. In ISO/TS 22176:2020 (https://www.iso.org/obp/ui/#iso:std:iso:ts:22176:ed-1:v1:fr) about the analytic methods and the development of global approach for the validation of the analytic quantitative methods, an overview is given about the definitions and link between a part of the concepts mentioned above. To achieve a high level of exactness, it is necessary to carry out a specific validation process to demonstrate the validity of the analytical method, as well as the accuracy of the measurement obtained by the system in question (virtual sensor in our case). As part of this approach, it is necessary to determine a tolerance interval within which falls a given proportion (β) of future values produced (on average) by the sensor model (sensor measurement). If this tolerance interval is located within an acceptance limit defined a priori, taking

into account several parameters related to the data produced by the sensor model, to the parameters of the environment used, to the type of analysis, and under the specific conditions of the experiments and measurements, in this case, the modeling of the sensor or the system will be considered valid. If it does not comply with this acceptance limit, the modeling of the sensor or the system will be deemed invalid.

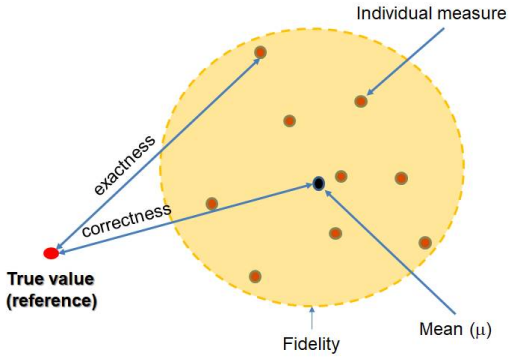

Figure 42: Representation and relationship between the concepts of Correctness, fidelity, and exactness

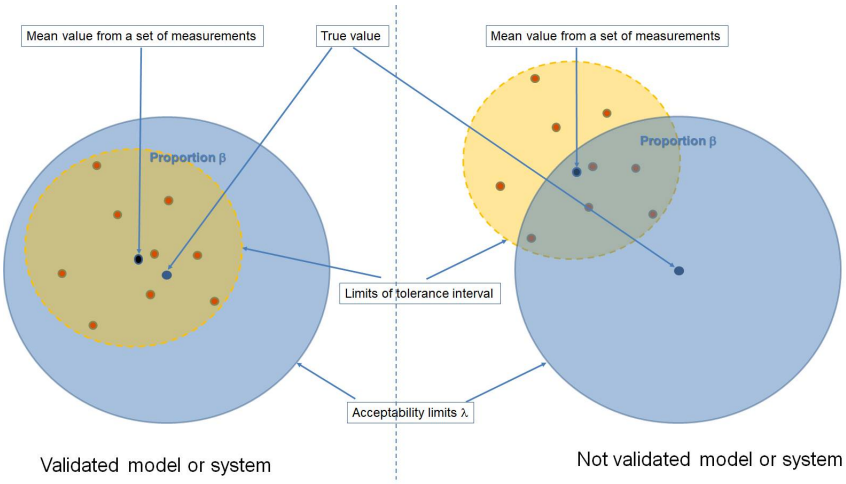

Figure 43: Representation of the validation process

If we apply this evaluation stage to a collaborative inter-sensor study based on the NF ISO 5725-2: 1994 standard; at the end of the tests carried out under repeatability conditions in each sensor or system models, with the same measurement method on the same object or on similar objects (corresponding to a given level), the processing of the measured values is carried out according to three steps.

- 1. Calculation of the mean and the experimental standard deviation (si) for each sensor model i in the same context.
- 2. Analysis of the si and yi values by a centralized method and rejection of sensor models with outliers according to suitable statistical tests: for example the Cochran test (standard deviation analysis) and the Grubbs test (analysis of means).

- 3. Calculation of the repeatability standard deviation sr and the reproducibility standard deviation sR of the series of measured values corresponding to the selected sensor models.

These three steps are shown in Figure 44.

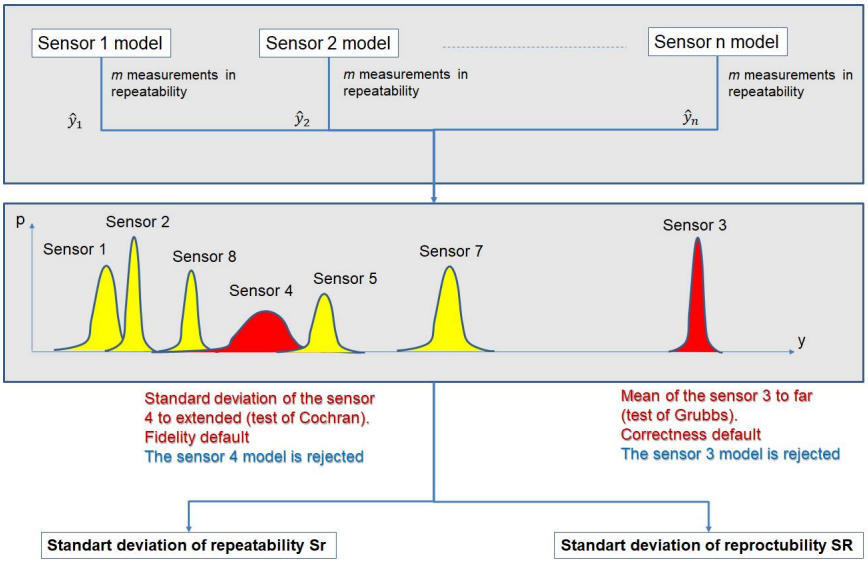

Figure 44: Process for sensor model validation

3.4.2 Accuracy in metrology equals uncertainty in perception/fusion

In metrology, uncertainty characterizes the resolution of the measurement and in Perception/data and information fusion/data processing, the degree of conformity with reality, or even confidence.
Considering that the **reality** of the "*Perception*" corresponds to the true value of metrology (which seems an acceptable hypothesis), the degree of conformity of a data to reality corresponds to the difference between the measured value and the true value. The uncertainty of the perception therefore corresponds to the accuracy of the metrology.

3.4.3 Uncertainty in metrology equals inaccuracy in perception/fusion

In fusion and perception systems (AI-based systems), the inaccuracy translates measurement errors (on a continuous discernment framework, this is the measurement noise, or on a discrete framework), which is entirely compatible with the term uncertainty in metrology.

3.5 Verification and validation of simulated models

In the previous section, we addressed the different metrics allowing to assess the quality, the performance, and the fidelity of a simulation model. In our case, it is mainly focused on sensor models, vehicle models, road users models, and environment models. This section is more dedicated to the definition of the process allowing to have a verification stage of the models, then to validate theses models, and finally to provide a level of representativeness and an accredition on a specific model in an accurately define domain of operating.

Definitions: *Verification is the process of determining that a model implementation and its associated data accurately represent the developer's conceptual description and specifications. Validation is the process of determining the degree to which a simulation model and its associated data are an accurate representation of the real world from the perspective of the intended uses of the model*

The development of a simulation model that has undergone a formal **verification, validation, and accreditation (VV&A)** process is not only desirable, but essential.

Consider the following definitions for the phases of the simulation model VV&A process [144] (see figure 45):

- **Verification**: "The process of determining that a model implementation and its associated data accurately represent the developer's conceptual description and specifications.". Verification process consists in determining if a simulation system or model performs a physical system as intended. This stage quantify if the simulated system or model is built sufficiently well to fit properly with the physical model.
- **Validation**: "The process of determining the degree to which a simulation model and its associated data are an accurate representation of the real world from the perspective of the intended uses of the model."
- **Accreditation**: "The official certification that a model, simulation, or federation of models and simulations and its associated data are acceptable for use for a specific purpose."
- **Simulation Conceptual Model**: "The developer's description of what the model or simulation will represent, the assumptions limiting those representations, and other capabilities needed to satisfy the user's requirements."

A simulated model is **credible** when its results are accepted by the user and can be used as an help in the decision process with a high level of accuracy. Animation is an effective way for an analyst to establish credibility of a model.

3.5.1 Verification

Verification answers the question "*Have we built the model right?*" whereas validation answers the question "*Have we built the right model?*" [145]. In other words, the verification phase of VV&A focuses on comparing the elements of a simulation model of the system with the description of what the requirements and capabilities of the model were to be.

Verification is an iterative process aimed at determining whether the product of each step in the development of the simulation model fulfills all the requirements levied on it by the previous step and is internally complete, consistent, and correct enough to support the next phase [146]. The overall process of comparing the model and its behavior to the real system. In this stage, calibration is needed. The calibration consists to apply an iterative process of comparing the model to the real system and making adjustments.

Comparison of the model to real system is shared in several sub stages:

- **Subjective tests**: People who are knowledgeable about the system
- **Objective tests**: Requires data on the real system's behavior and the output of the model

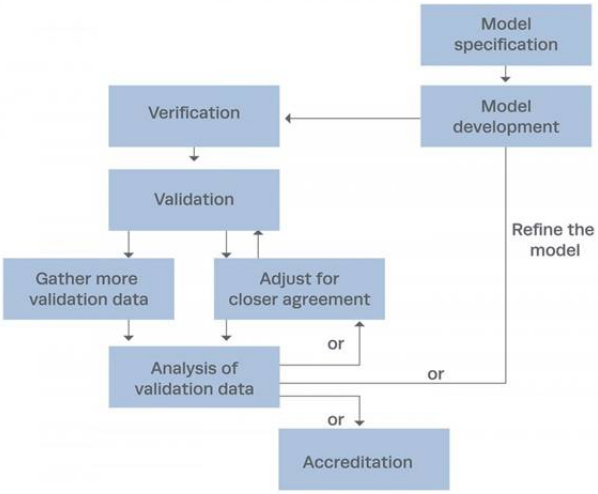

Figure 45: Simulation Model Development and the VV&A Process

3.5.2 Validation strategies, techniques and attributes

The validation phase of VV&A focuses on the agreement between the observed behavior of elements of a system with the corresponding elements of a simulation model of the system and on determining whether the differences are acceptable given the intended use of the model. If a satisfactory agreement is not obtained, the model is adjusted to bring it in closer agreement with the observed behavior of the actual system (or errors in observation/experimentation or reference models/analyses are identified and rectified).

Validation techniques (see figure 46) can be applied across a wide range of simulation systems and models. A set of attributes can be identified and should be considered when assessing the utility of any validation technique ([18]):

- **Applicable to scalar data**: the suitability of a validation technique to be applied for comparing scalars. A scalar is a single numerical quantity observed/calculated in one or multiple repeated experiments/computations.
- **Applicable to vector data**: the suitability of a validation technique to be applied for comparing vectors. A vector is a finite collection of scalars.
- **Applicable to scalar time series**: the suitability of a validation technique to be applied for comparing scalar time series, comprising a sequence of scalars recorded at successive time points. Unlike scalar and vector data, time series data often have serial dependence, in which there is statistical dependence between a value observed at time point ti and the value observed at another time point t_j.
- **Applicable to vector time series**: the suitability of a validation technique to be applied for comparing vector time series which are a sequence of vectors recorded at successive time points. Vector time series can be considered as a collection of multiple scalar time series; consequently, they too often have serial dependence.
- **Consider multivariate correlation**: the ability of a validation technique to use the correlation information of multivariate data. Although a validation technique suitable only for

univariate data could be applied to each response of the multivariate data, the validation results for each response might be in conflict.

- **Include objective criteria**: the status of a validation technique to have objective criteria to accept/reject a model. An objective criterion is developed based on mathematical or statistical reasoning.
- **Quantify model confidence**: the ability of a validation technique to provide a quantitative assessment of the validity of the model in terms of model confidence. For example, in hypothesis testing, the null hypothesis is set up to support the fact that the computer model is accurate. Model confidence is the probability of this null hypothesis being true.
- **Incorporate SME opinions**: the ability of a validation technique to utilize information provided by Subject Matter Experts (SME) in the process of validating a computer model.
- **Normality assumption independence**: the independence of a validation technique on the use of normality assumption for the distribution of either test data or CAE (Claim-Argument-Evidence) data. More generally, it is desirable that a validation technique does not require any particular distribution model.
- **Insensitivity to type-I error**: the insensitivity of validation results to the type-I error level specified for classical hypothesis testing validation techniques. Type-I error level, or the rate of type-I error, is the probability of rejecting the null hypothesis when it is true. It is known that specifying the type-I error at different values can lead to different validation results (i.e. from accept to reject the model).
- **Low computation cost**: the time needed to execute the validation technique.
- **Sample size independence**: the insensitivity of the validation results to the selection of sample size. Sample size is the number of observations in a sample which is a subset of the population. Validation results should be similar if data of different sample sizes are used.

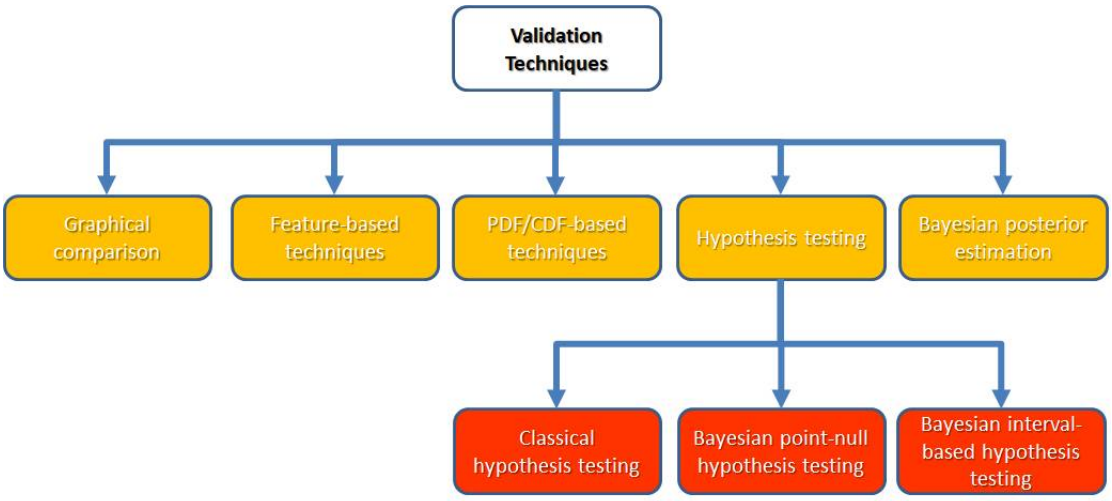

Figure 46: Validation techniques for simulated systems and models ([18])

In [66], a taxonomy of virtual validation strategies for automated mobility means. The figure 47 presents this taxonomy.

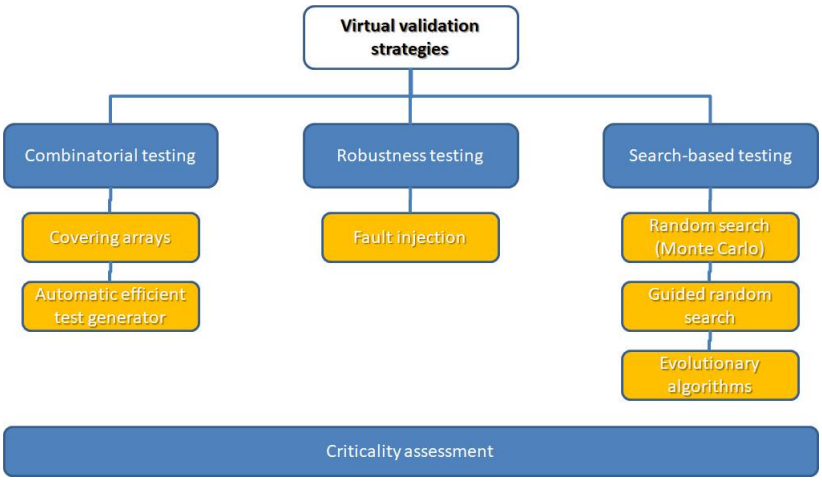

Figure 47: Taxonomy of virtual validation strategies ([**?**])

In [147], the authors proposed an ontology-based test generation for automated and autonomous driving functions with a general framework for testing, verification, and validation for automated and autonomous driving functions (see 48).

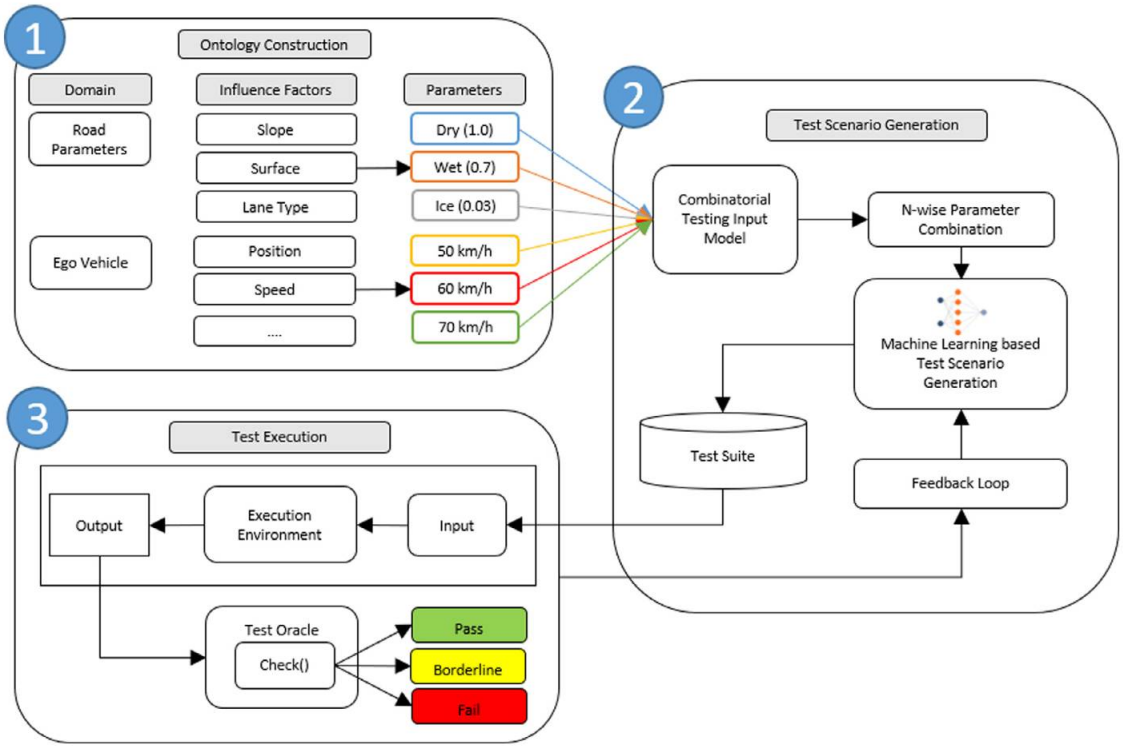

Figure 48: A general framework with 3 main pillars for testing, verification, and validation for automated driving functions ([19])

3.5.3 Confidence Interval Testing

Confidence interval testing: evaluate whether the simulation and the real system performance measures are close enough. If Y is the simulation output and $\mu = E(Y)$, the confidence interval (CI) for μ is:

$$[\hat{Y} - t_{\frac{\alpha}{2},n-1}.\frac{S}{\sqrt{n}}, \hat{Y} + t_{\frac{\alpha}{2},n-1}.\frac{S}{\sqrt{n}}] \tag{19}$$

with n the fixed sample size. α is the level of significance and corresponds to the Type I error which represent error of rejecting a valid model. This type of error is controlled by specifying a small level of significance α.

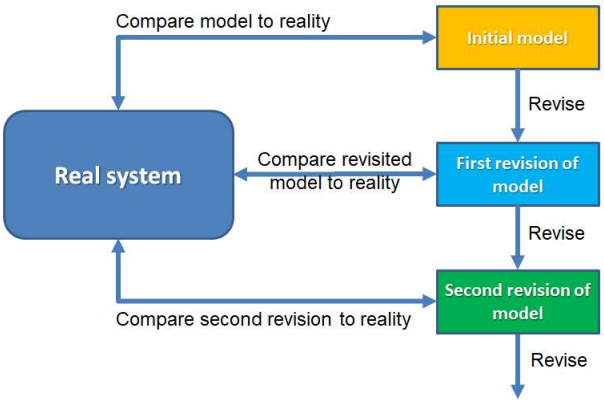

Figure 49: Stages of validation process for simulated models

Model Validation Method	Description
Comparison to Other Models	Various results (e.g., outputs) of the simulation model being validated are compared to results of other (valid) models. For example, simple cases of a simulation model are compared to known results of analytic models, and the simulation model is compared to other simulation models that have been validated.
Face Validity	Asking individuals knowledgeable about the system whether the model and/or its behavior are reasonable. For example, is the logic in the conceptual model correct and are the model's input-output relationships reasonable?
Historical Data Validation	If historical data exist (e.g., data collected on a system specifically for building and testing a model), part of the data is used to build the model and the remaining data are used to determine (test) whether the model behaves as the system does.
Parameter Variability – Sensitivity Analysis	This technique consists of changing the values of the input and internal parameters of a model to determine the effect of output on the model's behavior. The same relations should occur in the model as in the real system. This technique can be used qualitatively—directions only of outputs—and quantitatively—both directions and (precise) magnitudes of outputs. Parameters that are sensitive (i.e., cause significant changes in the model's behavior or output) should be made sufficiently accurate prior to using the model.
Predictive Validation	The model is used to predict (forecast) the system's behavior, and then the system's behavior and the model's forecast are compared to determine if they are the same. The system's data may come from an operational system or be obtained by conducting experiments on the system, e.g., field tests.

Figure 50: Common Simulation Model Validation Methods

In [148], the authors proposed a model validation and scenario selection for virtual-based homologation of automated vehicles.
In [149], Validation of the simulation models.
In [150], Integration, Verification, Validation, Test, and Evaluation (IVVT&E) Framework for

System of Systems (SoS)
In [151], A Framework for Automated Driving System Testable Cases and Scenarios (chapter 5: Preliminary tests and evaluation methods and APPENDIX B. Modeling and simulation for scenario testing)

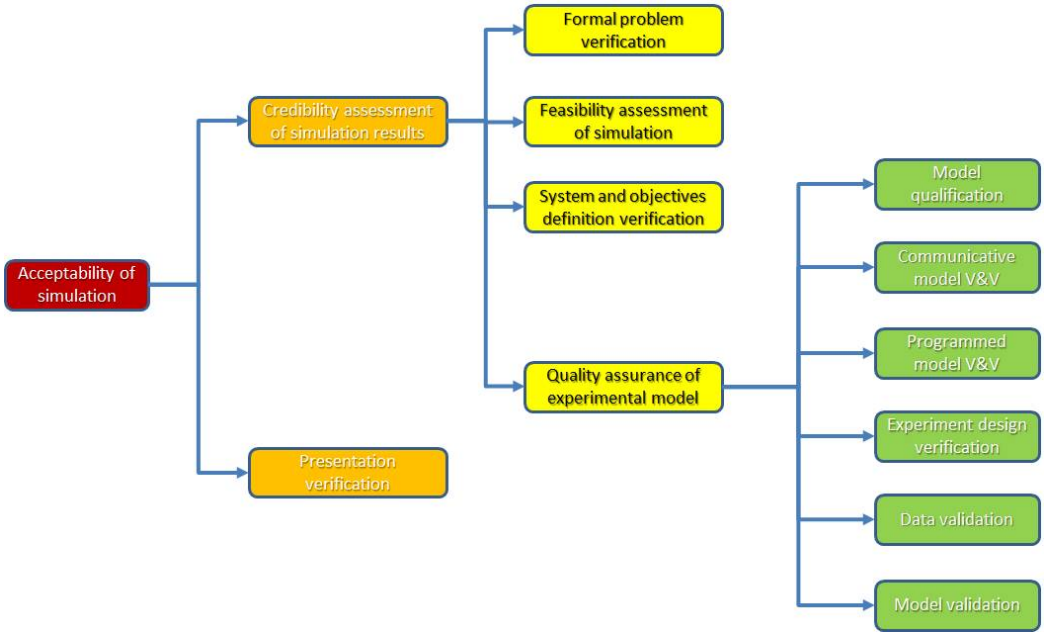

Figure 51: Indicators hierarchy for validation process of simulated systems and models

3.5.4 Literature review of validation methods for AI systems

Society today is dependent on AI systems to some extent, such as intelligent service systems, manufacturing robots, and autonomous driving vehicles, etc. The awareness of the complexity related to the validation of AI systems began with the arrival of the first systems that initiated the AI methods, such as Advanced Driver Assistance Systems (ADAS) of Autonomous Driving vehicles [152]. The rapid development of AI provide more chances to the users, which is manifested in that they do not need to have a good command of(even no) knowledge in these fields to use the existed tools and library of AI methods. Coupled with the difficulty of testing AI systems with traditional methods, the credibility of system has become an urgent issue. The difficulties are main from the characteristics of AI itself [153] and can be generalized as follows: 1) unpredictability and unexplainability of individual outputs [154], 2) difficulty of maintaining consistency and weakness against slight changes in inputs [155], 3) unanticipated, emergent behavior, and unintended consequences of algorithms [156], 4) complex decision making of algorithms [157], etc. From the result of the systematic literature review (SLR) of AI systems validation [20], there are several approaches to validate AI systems, including simulation-based, trials, model-centered and expert opinion methods (as in figure 52). Besides, software-based methods, formal methods (e.g., formal proofs, model checking, and probabilistic verification), and some specific methods are also applied to the validation of AI systems.

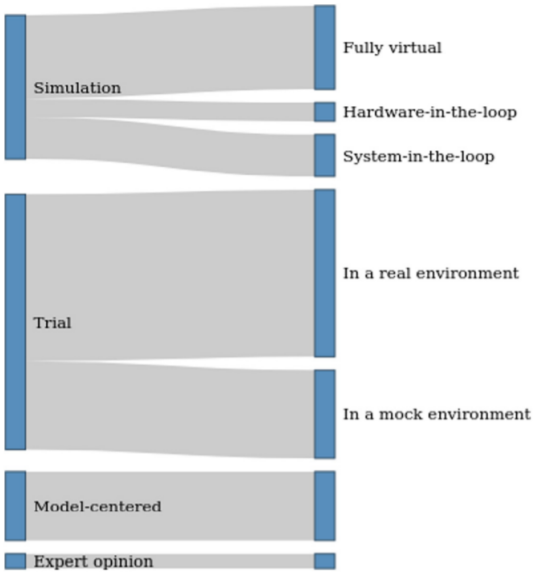

Figure 52: The taxonomy of the validation methods [20]

3.5.4.1 Fault injection validation

Software-based fault injection methods are designed to emulate at the software level the faults that would have occurred either in the hardware components or during the software execution [158]. By modifying the constants, operators, variables, and/or software source code, the logic and arithmetic expressions in the program will be revised. The process of validation can be fully automated and target both application and operating system. [159] developed the Autonomous Vehicle Fault Injector (AVFI), where the faults are injected into AI neural network by inputting damaged sensor data into the neural network, and the vehicle behavior is simulated in CARLA [160] during the experiment.

3.5.4.2 Functionality-based Validation

Functionality-based method is a type of software testing that validates the software system against the functional requirements/specifications. In the autonomous vehicle case, this validation method decomposes the scenarios into various operational components that can be tested individually, in which the intelligence of a system has been categorized into three parts [161, 162] : 1) recognition, 2) decision, and 3) action functionalities. The basic principle of this kind of method is that autonomous vehicles should be able to retrieve various functionalities for a given task analogous to human beings. For example, vision-based recognition functions can be further decomposed into some specific intelligent functions [21], including road surface recognition, lane line recognition, vehicle recognition, traffic sign recognition, traffic light recognition, pedestrian recognition, obstacle recognition (as shown in figure 53). These recognition information will be the inputs of decision functionality. Finally, the combination of several outputs of decision functionality will lead to corresponding action functionality.

Figure 53: An illustration on a category of vehicle vision based recognition functions: including road boundary detection, lane detection, traffic sign detection,vehicle detection, and etc. [21]

3.5.4.3 Simulation validation

Simulation as a validation method is heavily favored in cyber-physical systems [20], since the simulators are closed, indoor cubicles that act as substitutes for physical systems [163]. They can replicate the behavior of any system by using hardware and a software model, and validate the systems in an artificial environment virtual or real) simulating the actual deployment environment.

- **Fully virtual simulation validation**: In the fully virtual simulation validation, the input from the real world will be replaced and provided by the virtual simulator, which contains the simulation of the final environment and the possible physical components [164, 165, 166]. As a result that the actual danger has been avoided in the real scenarios [147], this validation method is able to make the validating process much safer, and it is also efficient in some highly complex system cases [167]. Although simulation attempts to replicate the real world as closely as possible, the inherent limitations always create some deviation between the two, which may cause serious consequences.

- **X-in-the-loop simulation validation**: X-in-the-loop (XIL) stands for the seamless integration of all relevant component and systems (software, hardware, and model) in the development, integration and tests loop [168]. In terms of automated vehicles, not only the demand for simulation is yet higher but also the complexity of XIL approaches due to static and dynamic environmental influences, such as driving environment and traffic scenarios. XIL simulation validation focuses on the modeling of the sensor models, the vehicle with its actuators, the definition of driving scenarios, as well as autonomous driving functions, which provides a novel approach and test architecture to validate the perception systems, planning and control logic of autonomous vehicles using simulation and virtual techniques [169]. Mode-in-the-loop(MIL) simulation is simulated in closed-loop with model of vehicle dynamics, sensors, actuators, and the traffic environment. The new simulation concept PRESCAN [170] allows a reliable MIL simulation of ADASs, which use validated physical sensor models for radar, lidar, and camera vision in a virtual environment.

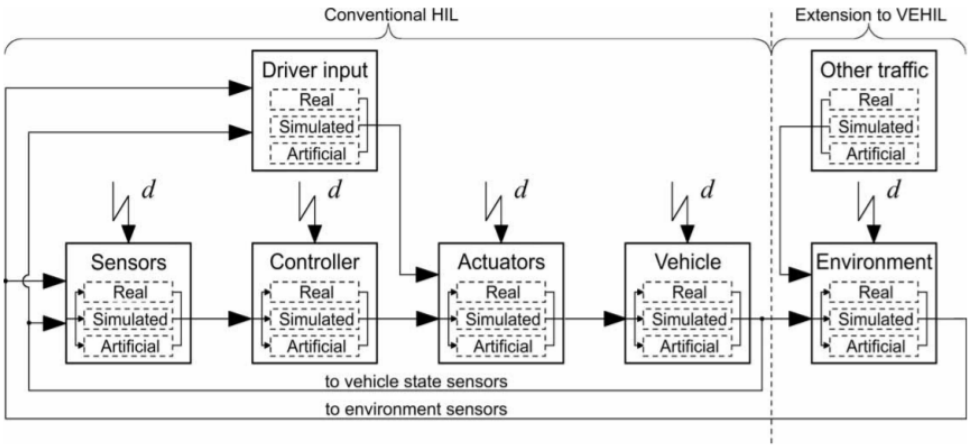

Figure 54: Possible configurations for HIL simulations

Hardware-in-the-loop (HIL) fixes the deviation problem a little by consisting of simulated and physical components (camera, some agents) for certain aspects of simulation (as shown in figure 54). The usage of real hardware provides a high level of reliability. At the same time, HIL offers the flexibility of a simulation since every component can be replaced by real, simulated, or emulated components. As the result in that any vehicle components can be simulated in HIL simulation, the validation of the physical component is able to start in an early development phase, even without the complete prototype. For these reasons, HIL simulations are more efficient and cheaper than test drives and are extensively used for the development of vehicle control systems, such as ABS [171], engine control systems [172]. Comparing to the fully virtual simulation, hybrid-faults originating from the combination of hardware and software can be monitored more easily [173]. However, physical components are not required in simple model-centred systems, HIL simulation validation is impractical in this situation.

For some definite tasks, System-in-the-loop (SIL) simulation validation has a more precise picture of how the system would behave in the validation scenarios, which is dependent on the whole system being under validation. The full system is placed into an artificial environment (virtual or physical), which replicates some features instead of the whole real environment [174, 175].

In some application [176, 177] of the Vehicle-in-the-Loop (VIL) simulation validation, the complex system of tested vehicles is embedded into an artificially controlled, synthetic and virtual testing environment. Therefore, the real hardware and software framework of the investigated vehicular system is tested combinedly, which gain the advantages of both methods (as shown in figure 55). VIL simulation validation is also able to consider interaction with human, and provide a safer test to the driver at the same time. AV's environment is simulated by a comprehensive software framework, covering the most relevant decision factors and sensor types, such as traffic simulation, graphical representation, or point cloud generation [178]. The vehicles and simulation modules can be connected in real-time, even the real vehicle can be used instead of simulated vehicles in this such safe environment, where the driver is shown simulated feeds of the external environment to capture his interaction with the car, and the car travels across a ground devoid of obstacles, simulating inertial effects and simultaneously responding to the external feed. The

systems will be tested directly in the vehicle instead of driving into real traffic, which provides special advantages arise with safety assistance systems handling emergency situations [22].

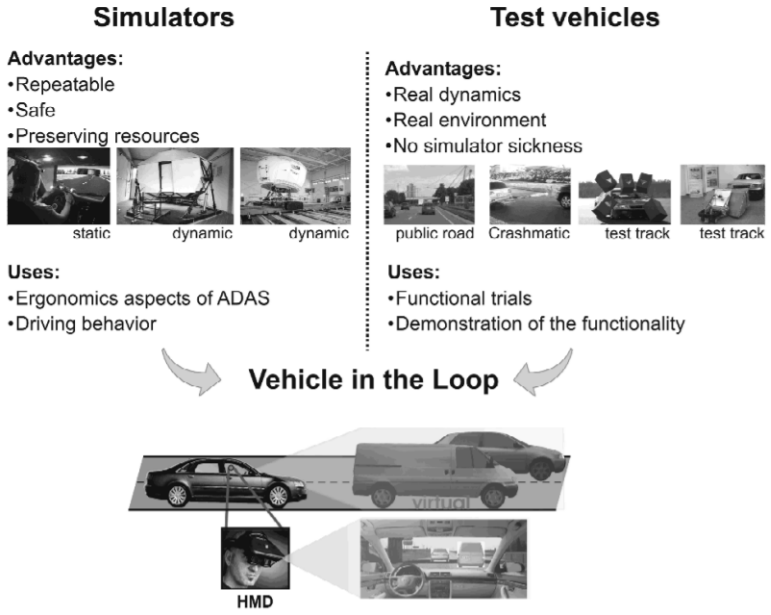

Figure 55: Vehicle in the Loop – combination of the real hardware and software framework [22]

3.5.4.4 Trial validation

Nothing is closer to the real world than the real world itself. Experiments in the real environment is also a method to validate the AI systems [174, 179]. The AV systems in this method will be driven onto a road with actual traffic [180, 181]. Sensor data is recorded and recorded to capture behavior in critical situations. Analyze them to adapt and fine-tune the system according to daily scenarios. Although the system is used in an actual use case, the validation settings can be arranged in a way that ensures that all the key scenarios are covered. However, sheer amount of test data is generated followed by it. In addition, the mock environment [182] could be replicated or duplicated from the real one, by which the system does not have to be brought into potentially dangerous environments in the trial.

3.5.5 Validation of sensor models in Pro-SiVIC^{TM}

3.5.5.1 Camera model validation ([183])

In order to validate the optical sensor modeling developed in the pro-SiVIC^{TM} platform, a set of tests has been made using a real laboratory for characterization of optical sensors. This real platform is described in [184]. The validation stage is mainly focused on focal length, distortion [185], vignetting [186] and sensor linearity [187]. Thus, two types of test targets have been used: Dot charts and Retro-lighting chart. Figure 56 shows the pictures of the test targets given by three real and simulated optical systems. In the order, downward, are illustrated the pictures obtained from SPC1030NC webcam, CM040MCL sensor and GE040CB sensor. For each of these three sensors, real system output is located to the left and simulated system output

is located to the right. These cameras have been modeled in Pro-SiVIC using real calibration information, and the simulated images of these charts have been recorded. Then, for each system, the correspondence between the real and simulated system has been analyzed. In the following figures (57), the results obtained are presented according to the position on the half-diagonal (center - corner) of the test target, named thus radial field position. The blue curves present the real system while the red dashed curve refers to the simulated system resulting from the Pro-SiVIC software. The relative error is represented by the green curve.

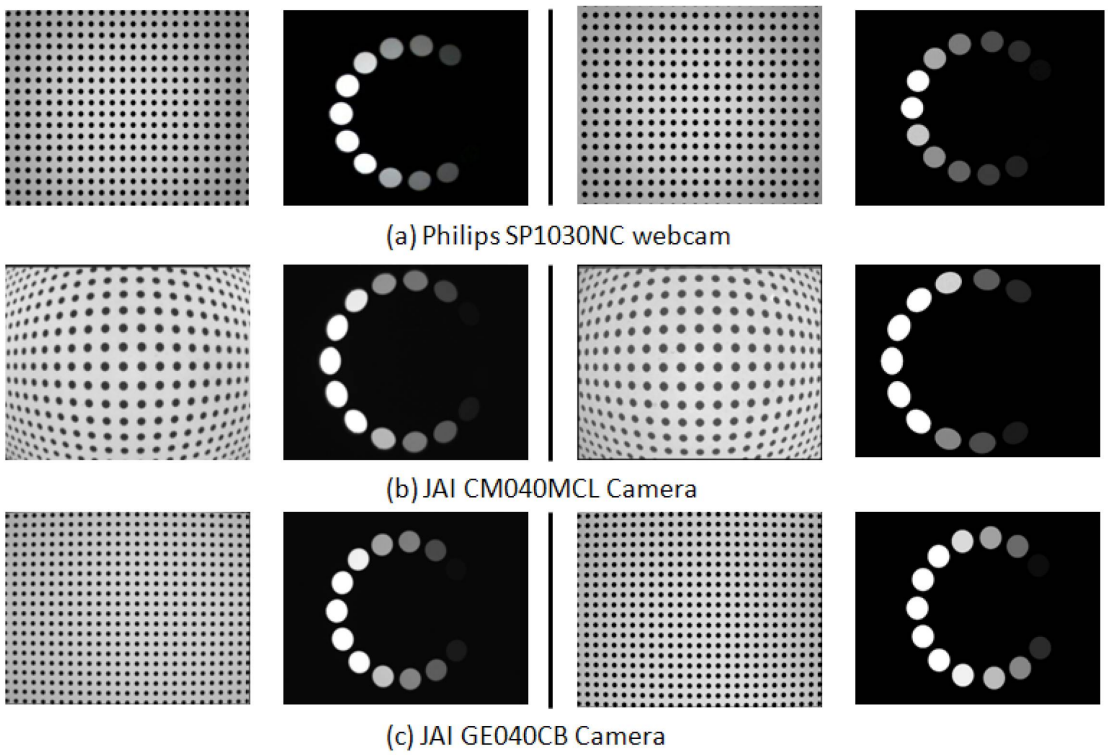

Figure 56: Pictures of the dot and retro-lighting charts for the real system (on the left) and simulated system (on the right).

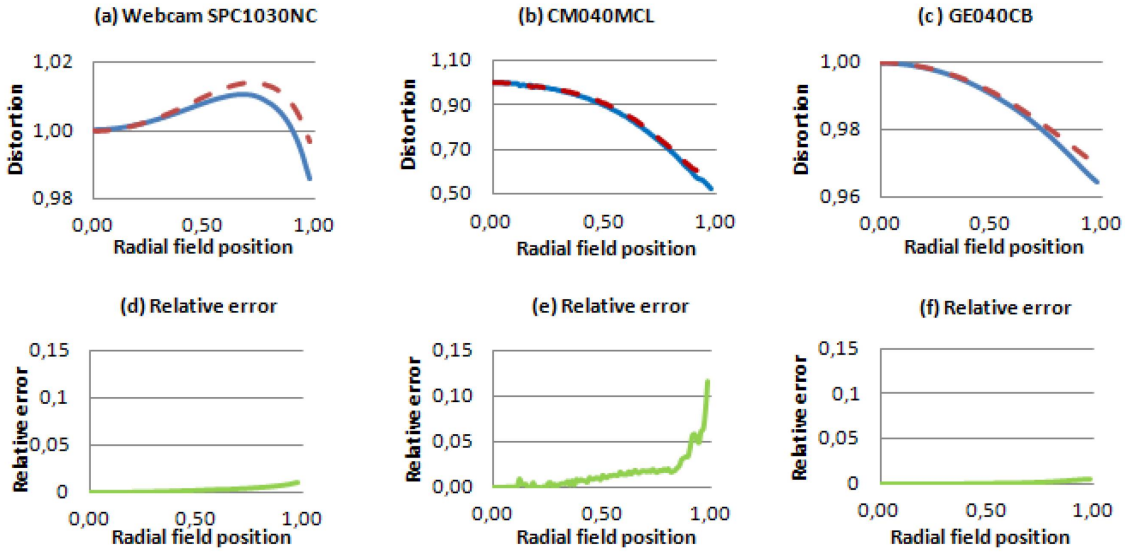

Figure 57: Measured distortion with (a) Philips SPC1030NC webcam; (b) JAI CM040MCL Camera; (c) JAI GE040CB Camera

3.5.5.2 LiDAR model validation

The simulation of a LIDAR sensor is complex due to the physics involved. A LIDAR sensor usually performs the detection of targets, and measures some characteristics of those targets, such as the distance, speed, angular position. The LIDAR system uses lasers pulses with a selected wavelength from ultraviolet to infrared range. The emitter composed by a LASER sends the light pulses and triggers a timer. As shown in Fig 58, the objects which are in the LIDAR Field Of View (FOV) reflecting LASER light to the detector which is composed by an electro-optical system that transforms the light signal into an electrical signal. The electrical signal is then processed by an electronic chain to obtain the targets information (distance, speed and reflectivity). The detected object then appears as a point cloud in the LIDAR display. Then, the LIDAR images quality depends mainly on its resolution. Indeed, the image resolution can vary from one LIDAR to another depending on its horizontal and vertical aperture angles, the number of LIDAR layers and the number of LASER impacts per degree. In addition, the frequency at which images can be developed is affected by the speed at which it can be scanned into the system and create a high-resolution picture. Finally, the modeling of the LASER beams with a point model is not enough. The LIDAR image quality depends mainly on LASER beam propagation, energetic modeling of the received signals, geometry of the illuminated objects with LASER as well as their powers of reflection, diffraction, absorption, different weather condition (fog, rain, snow, sun intensity, . . .).

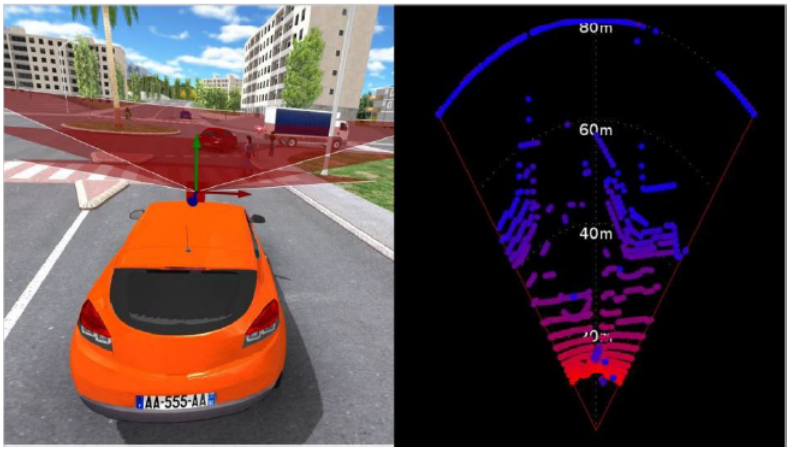

Figure 58: Automotive LIDAR illustration and 2D viewer. (source: ESI group)

3.5.6 Validation of PROSIVIC lidar model

PROSIVIC lidar model is used in multiple R&D and industrial projects. These different project helps to improve the fidelity of the sensors and estimate the gap with real LiDAR. The current version of the PROSIVIC lidar allows to simulate 2D and 3D LiDAR taking into account the scanning mechanism (Mechanical scanning, Solid state). In order to test the reliability of lidar simulation model, we virtually reproduced experiments published in literature, and we compare the numerical result with the published experimental data. Among these experiment we quote the works done in [188]. This experiment shows the detection of a car target by 2D LIDAR one long-range and the second is a short-range LIDAR both from SICK [**?**]. The two LIDARs are mounted in the bumper of a vehicle to perform several test sequences to compare the capabilities of each sensor. Several detection tests are described in the reference [188], and here we represent one of these tests. The chosen test consists in measurement of the detection points returned by a gray car target towards the detector as a function of the distance between the LIDAR and the target. The experimental and numerical results are shown in Figure 59 for the short-range LIDAR and Figure 60 for the long-range LIDAR.

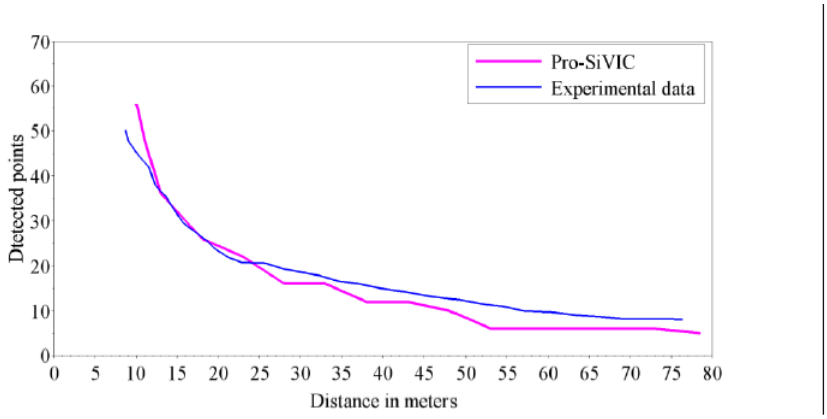

Figure 59: Detected point as a function of the distance between lidar and car target with short range SICK lidar LMS 291

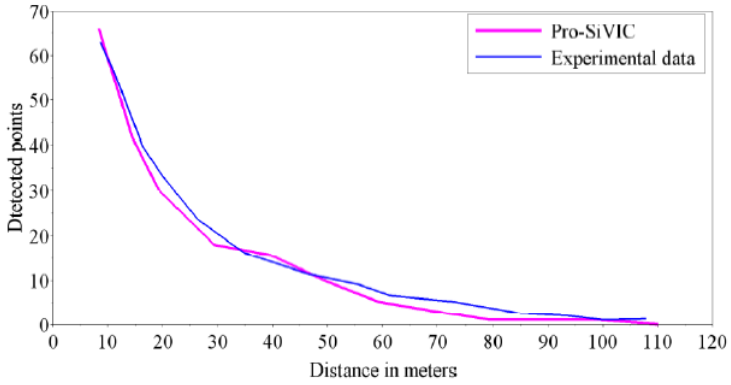

Figure 60: Detected point as a function of the distance between LIDAR and car target with short range SICK LIDAR LRS 1000

As advanced as the LIDAR may be, no current single sensor system guarantees a perfect accuracy of the measurement. The performance of LIDARs decreases in adverse weather conditions like rain, fog [189], snow and dust [190] or when the sensor gets blocked by e.g. dirt. The scenario illustrated in figure 61 represents the car to car Automated Emergency Breaking NCAP test in clear weather, medium and strong fog density.

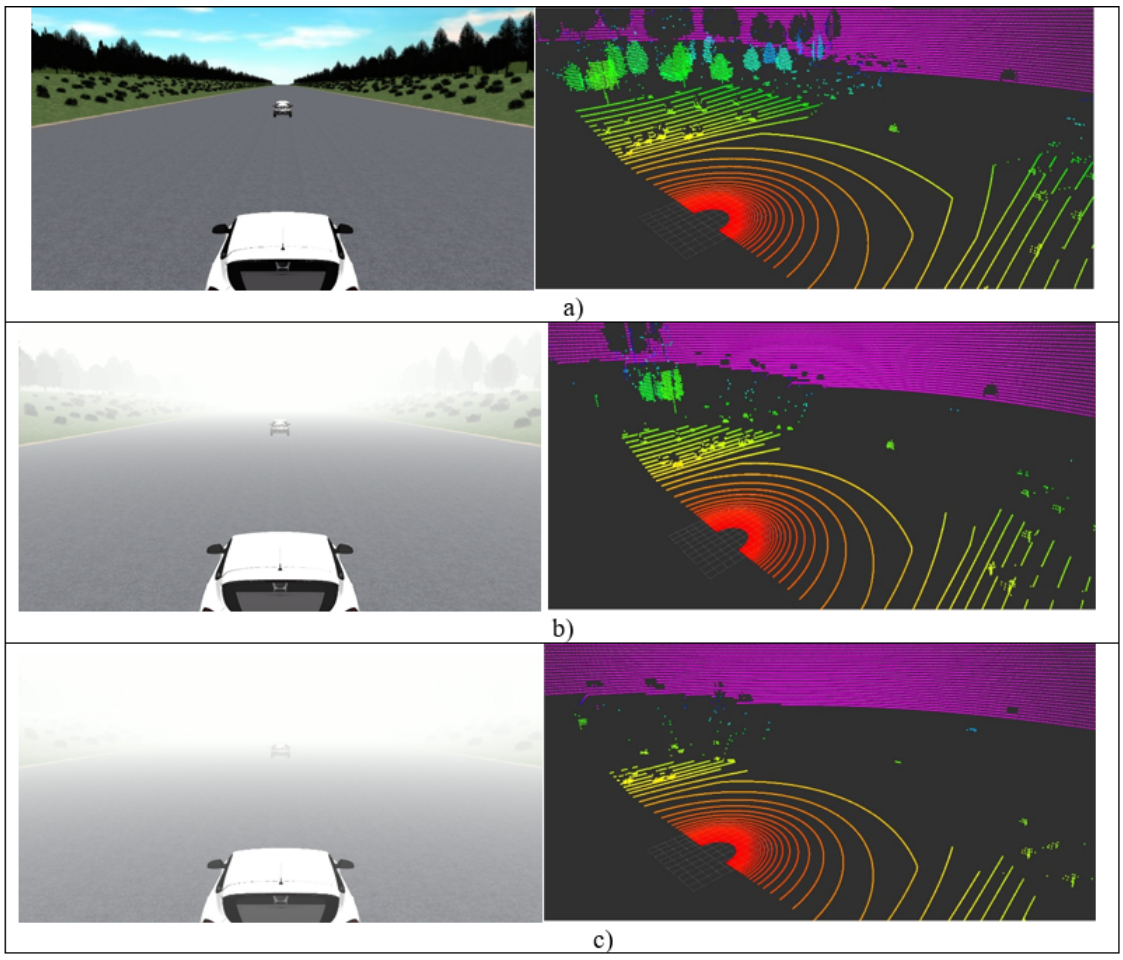

Figure 61: Simulation of the LIDAR sensor in Pro-SiVIC and representation of the laser points cloud in ROS. a) clear weather; b) low fog density; c) strong fog density (Source: ESI group)

The lidar data are represented by a 3D points cloud in ROS where LASER impacts with objects are represented with a gradient color as a function of distance and reflectivity. Regarding to the pink dots, these represents the LASER non-impacts whose are projected at the maximum range of lidar and it is equal to 120 m in this simulation. We can see that the lidar images (raw data) deteriorate as the fog density increases. This degradation in lidar images is related to attenuation of laser energy by fog particles. Indeed, the moist air acts as a screen for the infrared radiation where fog particles reduce laser intensity by absorption and diffusion phenomena. These simulations shows that harsh weather condition may cause lidar system outage due to the attenuation signal attenuation. The ProSiVIC helps to simulate all NCAP test in extreme situation and different weather conditions with an independent sensor manufacturer software [191, 192]. In addition, ProSiVIC can be applied for detecting different object (cars,pedestrians, animal, road signs,...), road marking. The method used to test lidar performance under harsh weather condition and perturbation can be used as a method to compare different sensors issued from different manufacturers. Assigning a score to the different sensors based on the number of detections will this allow to classify sensors according to their performances.

3.5.6.1 RADAR model validation

Early generations of driver assistance systems typically used a single environment perception sensor for each driver assistance function. Today's higher levels of automation often deploy sensor fusion concepts or semantic grids to detect also stationary objects [193]. The role of radar for driving automation has become increasingly important; with further developments towards higher resolution [194] and fully polarimetric devices [195], radar is considered a key sensor for autonomous driving [196]. Among the primary reasons for the success of radar are that it is more robust against adverse weather conditions compared to lidar or camera, and that it is able to measure the target's velocity via the Doppler-effect in addition to its range [197]. The virtual validation of automated driving functions requires meaningful simulation models of environment perception sensors such as radar, lidar, and cameras. There does not yet exist an unrivaled standard for perception sensor models, and radar especially lacks modeling approaches that consistently produce realistic results.

Holder & al had presented in [198] a measurements that exemplify challenges in the development of meaningful radar sensor models. They highlight three major challenges: multi-path propagation, separability, and sensitivity of radar cross section to the aspect angle. They also address challenges and suggest further research directions towards meaningful automotive radar simulation models

As mentioned earlier, various types of sensors are proposed within the ESI PROSIVIC platform with different levels of modeling. The simulation of electromagnetic sensors was previously achieved through dedicated rendering process tuned to detect RADAR targets or using ray tracing techniques combined with the modelling of the RADAR antenna (see figure 62 and 63. The strategy proposed is relying on a preliminary 3D characterization of the RADAR devices obtained using the ESI CEM One solutions [199]. RADAR sensors and/or devices are embedded in a simulation chain featuring three different stages,

1. the modeling of the emitting/receiving antenna itself,
2. the targets to be identified in the near-by environment and finally
3. the so-called propagation channel allowing both to interact.

The propagation of electromagnetic waves is usually handled through ray tracing or other SBR techniques (Shooting and Bouncing Rays). Same comment applies to the targets handling while dedicated developments will be presented in forthcoming papers (RCS, far field hot spots, etc.).

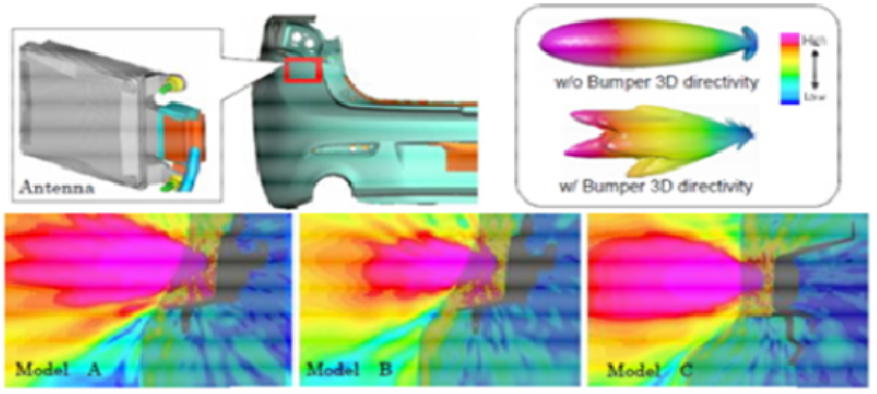

Figure 62: Electromagnetic design of a plastic bumper (Courtesy MAZDA Motor Corporation)

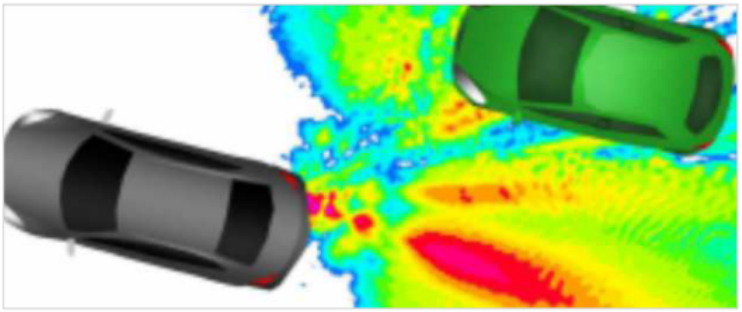

Figure 63: Near radiated fields with 24 GHz Blind Spot Detection (Courtesy MAZDA Motor Corporation)

The Radar in PROSIVIC represents the sensor system itself, including the antennas, the signal generation and the signal processing. The radar model is interacts with propagation channel, but has additional properties regarding the definition of waveform used.It consider multiples radiating device characterized with a preliminary 3D computation as illustrated in figure 62. User can choose to use an ideal (isotropic) radiating source, or real radiating source according to the direction being considered using the antenna directivity. This model allows reaching quite easily a much more realistic modeling of the sensor performance. The electromagnetic radiation can be assumed in free space or with the RADAR sensor located behind the plastic bumper and interacting with near-by metallic parts or metallized paint coatings

3.5.7 Validation of perception systems in Pro-SiVIC^{TM}

Some tests have been done with Pro-SIVIC in order to test the validity and the rate of detection of a great set of primitives (road markings, pedestrian, building ...). In the figure 64 the Univ-Eiffel buiding in Satory has been modelled. The same point of view and camera parameters have been applied. The goal was to assess the quality of the simulated camera comparatively to a real one for the egdes extraction.

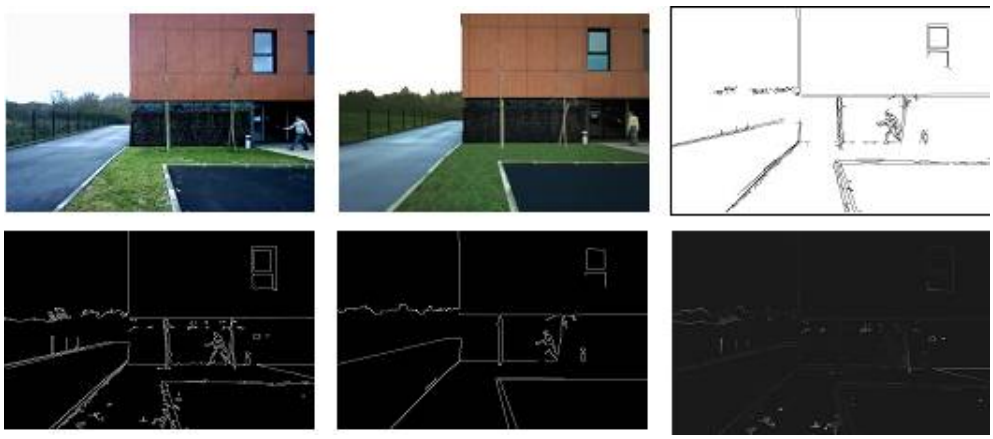

Figure 64: Validation of primitives detection with Pro-SiVIC (left: real image, middel: Pro-SiVIC image, rigth: differences) (source: Univ. Eiffel)

Another environment have been developed with a digital twin of the Satory test track, an OpenDrive modeling of the main track, and real emergency braking application (obstacle detection and tracking with LiDAR point cloud, decision stage, control of braking actuator). This scenario was a twin of the real same experiment with the modeling of the existing Univ-Eiffel's dummies.

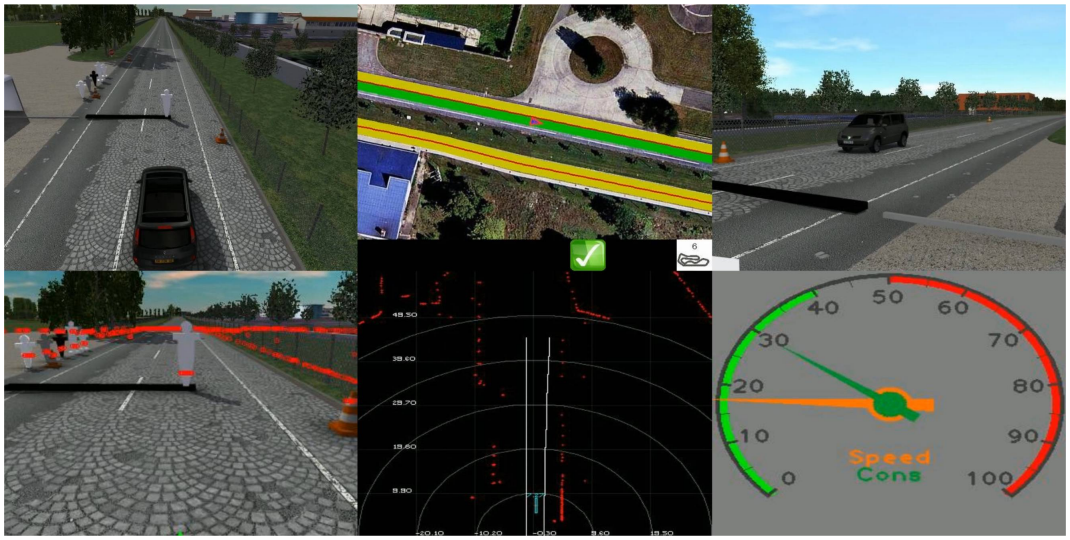

Figure 65: Quasper and ABV projects: platform for emergency braking system prototyping, test, and evaluation (source: Univ. Eiffel)

3.6 Formal proofs

In the validation of AI embedded in autonomous vehicles (AVs) through simulation, formal proofs hold a significant position. They provide a rigorous method to ensure the safety and reliability of AI systems, offering mathematical assurance of their behavior across various scenarios.

Regarding the types of formal proofs utilized, several approaches are commonly employed:

- Model Checking: This method involves exhaustively examining all possible states of a system model to verify whether certain desired properties are satisfied. It is often used to detect errors in specifications or AI models.

- Program Verification: This approach aims to mathematically prove the correctness of a computer program by demonstrating that it satisfies certain specified properties. This may include proofs of absence of critical defects such as buffer overflows or infinite loops.
- Temporal Logic: Temporal logic formalisms are used to specify and verify properties related to the temporal behavior of systems. For example, Linear Temporal Logic (LTL) and Computation Tree Logic (CTL) are often used to express safety and liveness properties.
- Automated Theorem Proving: These methods employ algorithms to automatically search for formal proofs. They can be used to prove properties of complex systems, including AI systems, using techniques such as constraint solving or proof by induction.

3.7 Ground truth

In order to evaluate and validate function, methods, algorithms in simulation environment, it is necessary to generate in same time a set of reference data use as accurate ground truth. Only these accurate reference allows to assess the level of quality of the components, systems, or system of systems. Among the necessary data, we can quote:

- accurate obstacles and VRU annotations
- 2D & 3D bounding boxes with attributes and classification for object that an autonomous system might encounter
- Pixel semantic and instance segmentation data-sets
- Depth data-sets
- Comprehensive scene and object attributes (environment segmentation)

In order to provide and to compute accurate environment segmentation and semantic reference, it is essential to segment the road environment in specific area with specific meanings. Some recent works proposed by the MIT (https://agelab.mit.edu/driveseg) gives efficient results. In order to be usable, a data-set need to involve such a data. About this segmentation aspect, several types of data annotation exist and include:

- 2D bounding boxes
- Lane marking
- Semantic segmentation
- Video tracking annotation
- Annotation of points
- 3D object recognition
- 3D Segmentation Sensor Fusion: Sensor Fusion Cuboids/Sensor Fusion Segmentation/Sensor Fusion Cuboid Tracking

The types of pixel segmentation are the following (see figure 66):

- **Instance Segmentation**: Each instance of an object(thing, eg: person) is labeled separately and segments without any instance(stuff) are ignored.
- **Semantic Segmentation**: Here all the regions of the image(thing+stuff) are considered and objects of the same type are considered as a single label.
- **Panoptic Segmentation**: It is a combination of Semantic Segmentation and Instance Segmentation such that all pixels are assigned a class label and all object instances are uniquely segmented. Here, each pixel is assigned a class label(example: person) and if a segment belongs to the thing category, each instance of it is labeled individually. Example annotation, syntax for each annotation: [< class label >, < instance id >], for thing: ["person", "1"], for stuff: ["building", "None"]

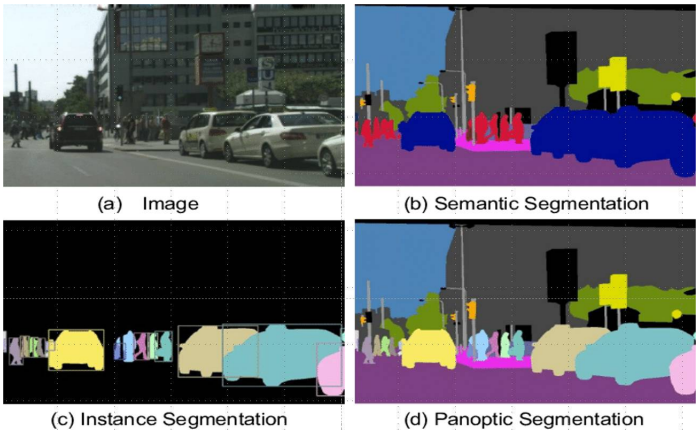

Figure 66: Types of image segmentation in an evaluation and validation process

Several types of image segmentation datasets (2D, 2.5D, 3D) are available like:

- **2D Datasets**: PASCAL Visual Object Classes (VOC), Common Objects in Context (MS COCO), Cityscapes
- **2.5D RGB-D Datasets** : NYU-D V2, SUN RGB-D, ScanNet
- **3D Datasets**: Stanford 2D-3D, ShapeNet Core, Sydney Urban Objects Dataset

Moreover, traditional Image Segmentation approaches involved:

- **Region-based Algorithms**: Thresholding, region growing
- **Edge detection algorithms**: Sobel operator, Laplacian operator
- **Clustering algorithms**: K-means
- **Other approaches**: active contours, conditional random fields, Markov random fields

If we look at the figure 67, it is easy to remark segmentation issues due to either large scale object with reflective material, or small size object difficult to share as several people. In this context, it is easy to understand the main advantage of the simulation platform. Effectively, in

these simulation platform, intern mechanism and the perfect acknowledge of the environment allow to generate accurate ground-truth and environment segmentation.

(a) Inconsistent labels due to large object size

(b) Missing labels due to small object size

Figure 67: Segmentation issues due to the objects size

In Pro-SiVIC, ground truth are generated by several mechanisms ([200]). The first one use ”observer” sensor concept which allows to generate in real time the stage vector of the different objects in the scene (vehicle, pedestrian, static object, road configuration). For instance, the vehicle observer provide a state vector with 40 dynamic parameters of the dynamic vehicle model during the simulation. For the pedestrian, only 16 fields are necessary in the state vector. The second mechanism is based on visibility layers. This mechanism offers a powerful tool to generate “ground truths” for optical processing with an instance segmentation method. Indeed, for the validation of the obstacle detection and road marking detection algorithms, it is possible, simultaneously with image generation, to obtain labeling images (of the obstacles and markings).

Figure 68: Pro-SiVIC with ground truth generation: obstacle annotation and depth map (source: Univ. Eiffel)

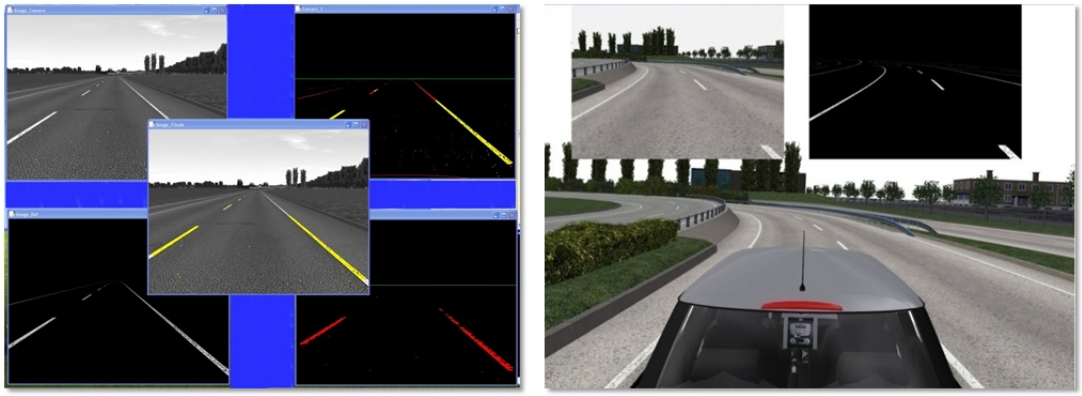

Figure 69: Pro-SiVIC with ground truth generation: road marking annotation ([23])

4 Modelling and simulation tools

A simulation, in the context of the validation of autonomous driving function or systems, is an implementation and execution of one, or a set of scenarios. It can be done by one or several tools working together depending on the complexity of the scenario and the expected outcome. Those simulation tools are composed of generic elements, each tool having some, or all of them based on their specialty.

4.1 Generic interfaces and data propagation

The simulation tools cannot generally manage all the aspect of a simulation, and for complex, detailed scenario, the interconnection between several tools is essential. The simulation block used together should have the same interface definition to share data efficiently. The simulation system can be composed of software pieces but also some hardware component. In the case hardware component are used, all the elements should be running at real time, meaning the

simulated time is the same than the real time. The cases that use only software component can be running faster or slower than real time depending on what is being simulated. If the goal if to produce massive data quickly it can run faster than real time, but if the simulation contains complex mathematical computation it can be slower than real time, the simulated time depend on the model performances. When several tools are used together, they can require to be synchronised, meaning they need to share the same simulated time so that the faster component wait the other one to have result coherent with the reality. Some cases can be Asynchronous, it is possible in particular when there is no feedback loop in the simulation and that processing some data faster than the rest does not impact the realism of the results. The data sharing interface, and communication protocol generally use standard such as FMI, but can also specific connector for a tool, that requires specific development.

4.1.1 Data exchange architecture

4.1.2 The need for open architectures

The need for data exchange architectures arose from the rapid evolution of electronic components for critical applications. During the 80s and 90s, the processors used in military systems were already obsolete before the systems were deployed in their operational environments ([201]). Open architectures were defined to make the software as independent as possible from the hardware, in order to allow the hardware to evolve to keep up with the increasing power of the market, without having to rework the entire software.

These architectures are based on more or less powerful hardware abstraction layers (HAL) to make the software independent of the hardware, even if the latter is based on standards such as PCI, PCIExpress, RS232, ATA, SATA. An HAL is defined as all the software that is directly dependent on the underlying HW. The examples of HAL include boot code, context switch code, codes for configuration and access to HW resources, e.g. MMU, onchip bus, bus bridge, timer, etc. In real HAL usage, a practical definition of HAL can be done by the designer (for his/her HW architecture), by OS vendors, or by a standardization organization like VSIA. Though HAL is an abstraction of HW architecture, since it has been mostly used by OS vendors and each OS vendor defines its own HAL, most of HALs are OS dependent ([202]).

4.1.3 The need for distributed architectures

Despite the improvement achieved by the standardization of the execution nodes with hardware components or broader usage of HAL, this standardization has shown limitations:

- Thee use of standards in the hardware and hardware abstraction layer was not enough to enable real reuse of software across different platforms.
- As system capabilities are moreover distributed over networked components developed by different suppliers, the standardization and openness of each equipment's technologies is not sufficient to maintain system capabilities despite the technological evolution of the components that constitute them.

The key is therefore not only the standardization of individual computation nodes, but also the standardization of the distribution of tasks accross multiple nodes

4.1.4 The network OSI layer as a foundation

Network of computation nodes basically falls into 2 main categories

- Broadcast (e.g. bus or ring topology) This mode of operation consists in using only one transmission medium. The principle is that the message is sent on the network, thus any network unit is able to see the message and to analyze according to the address of the recipient if the message is intended for him or not.
- Point to point (for example star or mesh topology) In this mode, the physical medium connects only one pair of units. For two network units to communicate, they must pass through an intermediary (the node).

Broadcast mode (e.g. bus or ring topology) This mode of operation consists in using only one transmission medium. The principle is that the message is sent on the network, thus any network unit is able to see the message and to analyze according to the address of the recipient if the message is intended for him or not. Point to point mode (for example star or mesh topology) In this mode, the physical medium connects only one pair of units. For two network units to communicate, they must pass through an intermediary (the node).

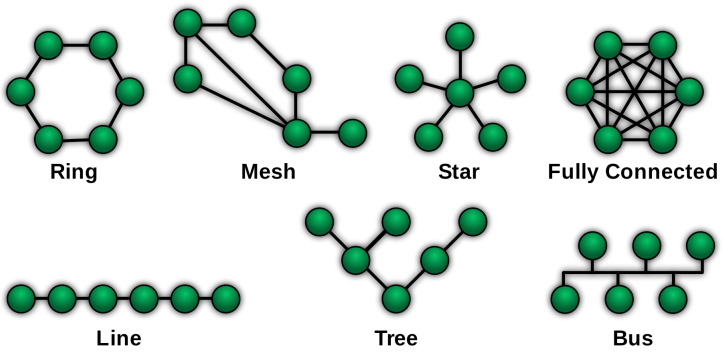

Figure 70: Main network topologies

Depending on the circumstances, each topology may have its advantages and disadvantages. While the ”fully connected” topology may seem the most interesting because it allows all the nodes to communicate with each other, it has disadvantages if it is implemented at the physical level: it requires a very large quantity of electrical wires, which means a significant cost and weight, particularly penalizing for an aircraft, for example. The OSI ISO/IEC 7498 model has made it possible to separate the various connectivity concerns of the different equipment in a system according to different layers.

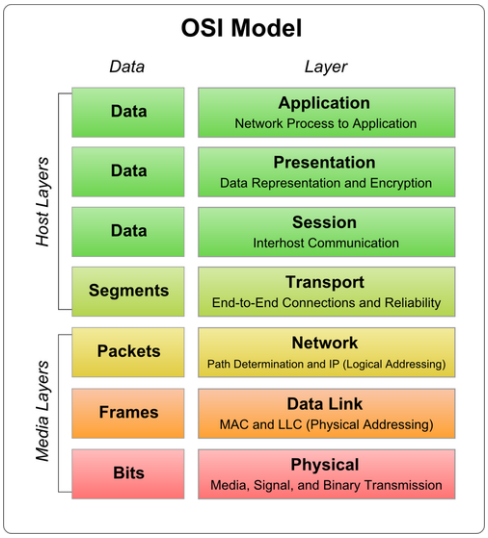

Figure 71: OSI ISO/IEC 7498 model

4.1.5 The rise of the middlewares

The standardisation of the different layers of inter-connectivity of software-intensive electronic computing resources has allowed the emergence of ”middleware” that allows the decoupling of the emerging behavior of all components from their implementation by each component. A middleware is a software that enables communication and management of data in distributed applications, originally defined as ”those services found above the transport (i.e. over TCP/IP) layer set of services but below the application environment”. In this more specific sense middleware can be described as the dash (”-”) in client-server, or the -to- in peer-to-peer.

Middlewares can be of different kinds ([203])

- Transactional: Processing of multiple synchronous/ asynchronous transactions, serving as a cluster of associated requests from distributed systems such as bank transactions or credit card payments.
- Procedural: Remote procedure call to connect, pass, and retrieve software responses of asynchronous systems communications such as a call operation.
- Message-oriented: Message queue and message passing architectures, which support synchronous/asynchronous communication.
- Object-oriented: Similar to procedural middleware, however, this type of middleware incorporates object-oriented programming design principles. Analytically, its software component encompasses object references, exceptions, and inheritance of properties via distributed object requests.

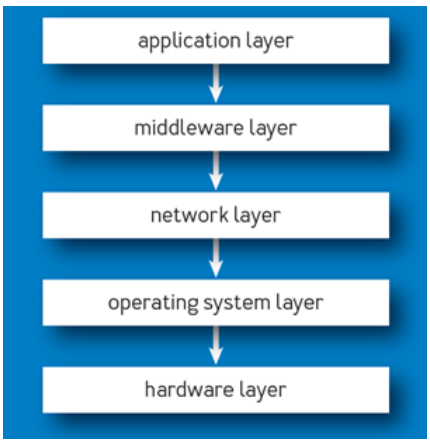

Figure 72: Layered models with middleware

The architecture of any technical system consists in determining which assembly of components (yellow) allows to fulfil the identified functions of the system (green). In an ideal framework, the system functions are executed by the application layer. The more powerful a physical platform is, the easier it is to implement the architecture task and the lower the realization costs.

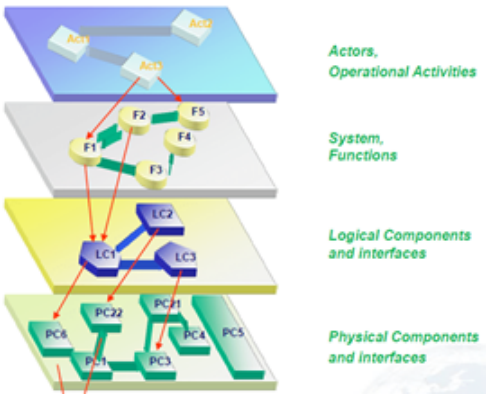

Figure 73: Allocation of functions to implementation

4.1.6 Tests systems architectures

In the VHTNG reference architecture the exchanges among the different components are grouped into three different categories (/citemartinen2017modular):

- The physical system under test communication channels, which regroups the communication of real components using real communication channels.
- The virtual system under test communication channels, based on the [ED247] standard simulate the communication between virtual components.
- The VHTNG Instrumentation networks which regroups the communication between the components of the test system itself.

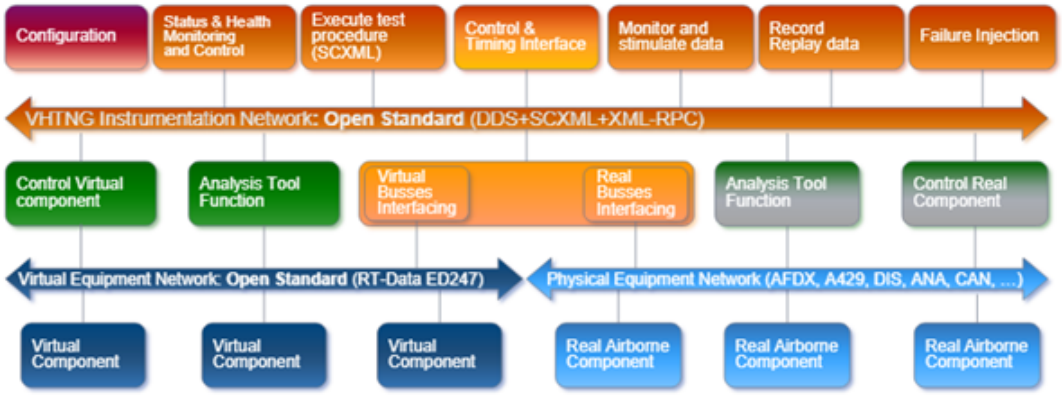

Figure 74: Allocation of functions to implementation

By enforcing standard interfaces for the Virtual Equipment network and the Instrumentation network, the VHTNG project aims at enabling the setup of distributed test systems provided by multiple suppliers. In the /cite9043010 article, the authors proposed a referenced implementation of their simulation platform using DDS middleware for real time distribution of simulation data. In addition to the DDS middleware, the FMU/FMI standard has been used for integrating simulation to this platform. In both VHTNG and this article we see the advantages of using standards for designing open architectures: the integration of new components adding features can rely on their compatibility on these standard for easier integration.

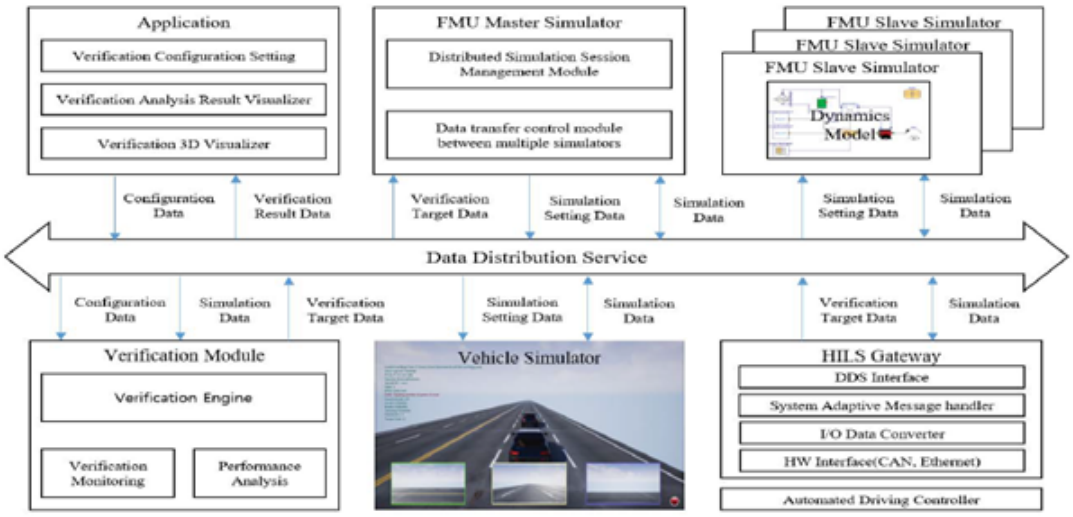

Figure 75: Simulation platform based on DDS FMI

4.1.7 Data exchange library

EPICS is a set of Open Source software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as a particle accelerators, telescopes and other large scientific experiments.

MQTT is a dominating protocol in the Internet-of-Things world. It is a “publish and subscribe” protocol, but as a very lightweight protocol it is very well adapted to transmit sensor data but not video clips.

XMPP is fast and real time Protocol, based on Extensible Markup Language (XML). Whereas MQTT doesn't define a message format; XMPP enables to define the message format and get structured data from the nodes of the network. XMPP provides a solid, flexible foundation for security features. XMPP facilitates identity management, authentication, authorisation, Off-the-Record Messaging (OTR), and encryption—including end-to-end encryption.

AMQP defines an efficient, binary, peer-to-peer protocol for transporting messages between two network processes (usually a client and a broker). AMQP separates the structure of a message, from its manner of delivery, with explicit and implicit meta-data which the using architecture can use. It supports multiple messaging protocols. It can be deployed in distributed configurations to meet high-scale, high-availability requirements

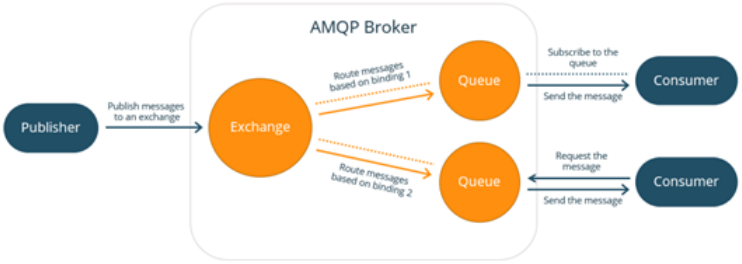

Figure 76: AMQP Broker

DDS implements a publish–subscribe pattern for sending and receiving data, events, and commands among the nodes of a distributed system. Nodes that produce information (publishers) create "topics" (e.g., temperature, location, pressure) and publish "samples". DDS delivers the samples to subscribers that declare an interest in that topic. DDS allows the user to specify quality of service (QoS) parameters to configure discovery and behaviour mechanisms up-front. By exchanging messages anonymously, DDS simplifies distributed applications and encourages modular, well-structured programs.

The ED247 virtualisation protocols provide possibilities for standardising the exchange of simulation data across virtual components. The standard eases interconnecting benches in different locations in a simple and non-intrusive way for the virtual data.

4.1.8 Data exchange format

The Functional Mock-up Interface (FMI) defines a ZIP archive and an application programming interface (API) to exchange dynamic models using a combination of XML files, binaries and C code: the Functional Mock-up Unit (FMU). The API is used by a simulation environment, the importer, to create one or more instances of an FMU and to simulate them, typically together with other models. The FMI defines three interface types:

- Co-Simulation (CS) where the FMU typically contains its own solver or scheduler
- Model Exchange (ME) that requires the importer to perform numerical integration
- Scheduled Execution (SE) where the importer triggers the execution of the model partitions

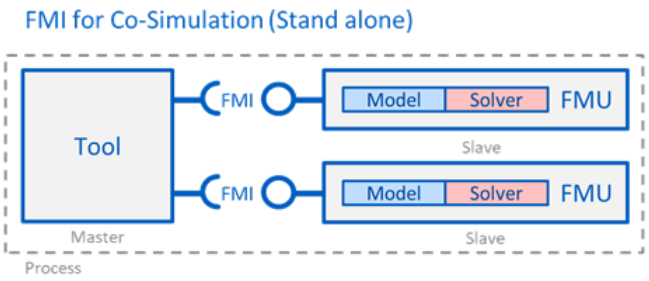

Figure 77: FMI usage

4.2 Simulation Engines: Physical and graphical engines

The simulation engine has the responsibility of representing the Scene described in the scenario. The scene is the set of environments, static and dynamic elements. The Simulation Engine level of detail can greatly vary in both graphics and physics which are its two main components. A basic scene representation can be schematics with low graphics and no physic with aim to only be a PoC that the scenario is correctly setup. The more the simulation require precise output, the more detail and effort are put in the scene representation. A detailed scene is essential to have good result from advanced sensor simulation, and most of the effort is on having detailed graphics with correct physical properties setup. A very high-level scene can be a digital twin, that is an exact representation of a real environment, with very accurate detail and physic elements. This allows direct comparison between real driving data and simulated data.The scene can be imported from a data set. The Simulation Engine can also have weather simulation capability which can provide variation on a same scene that are hard to reproduced exactly in real driving condition such as rain, fog, or lighting.

Simulation engine needs to provide some essential key features in order to be usable and efficient for robotics and autonomous vehicles researches. A part of these key features are listed below:

- **Physical fidelity**: Realistic simulation, suitable for virtual-reality environments, such as a sensor and environment simulator; most recent game engines feature both rigid and soft body dynamics, some of them even use a new dedicated hardware named Physics Processing Unit (PPU). Cutting-edge lightning effects, polygon rendering, and realistic destructible environments are also present/considered. Finally, meta data are needed in order to simulate the waves interactions with the objects and their specific materials.

- **Distributed architecture**: Support for multiple processor cores is included in earlier frameworks for maximum computational resources exploitation. It is possible to simulate multiple entities in multiple networked computers, distributing the processing power over all nodes. More and more, the use of cloud architecture or remote computers architecture is essential in order to guarantee real-time processing and sharing of processing on several dedicated units.

- **Cutting-edge graphics**: Use of game engines will significantly increase the level of detail and realism of the environment; relating to camera sensor simulation, higher resemblance from the virtual to the real world can be achieved.

- **Scriptable environment**: Featuring simple but powerful scripting languages, game engines can be rapidly extended to support a new type of sensor, or an optimized statistics module.

- **Events and ground truth generation**:

The main problem with game engine simulators is that they may not be high fidelity. A game engine allows to test the dynamics based on behaviour, but when a high-fidelity simulation is required, it is necessary to use models or software that contains a mathematical modelling of the subsystems to achieve realistic calculations. This software is often validated with HIL tests, used to a large extent in the evaluation of computer-based test equipment. The high-level algorithms for trajectory planning, vision processing and interactions of multi-agent systems are examples of suitable fields for use with simulators based on game engines. The principal game engines used in the development of autonomous vehicle simulators, or for some of its subsystems, are:

- **Unity 3D** [69] is an open source Game Engine, which is primarily used to develop video games and simulations for computers, consoles and mobile devices. The Unity graphics engines use OpenGL, Direct3D, OpenGL for Embedded Systems (OpenGL ES) for mobile platform (iOS, Android) and various APIs. The Unity engine provides built-in support for PhysX physics engine with real-time cloth simulation on skinned meshes, collision layers, and thick ray casts.

- **Unreal Engine 5.0** [70], is, like Unity, a popular general-purpose games development engine. It provides a scripting engine, physics engine, and highly realistic video capabilities.

- **Blender** [71] is an open source 3D modelling and rendering application whose main purpose is the creation of computer generated images and animations. Though it is not designed as a tool for simulation, it provides many features that facilitate the development of such an application. A community of robotics researchers who use Blender for some simulations already exists, and there is a drive to improve on this functionality. Blender has BlenSor [72], a Free Open Source Simulation Package for Light Detection and Ranging (LiDAR/LADAR) and Kinect sensors.

- **Cry Engine** [73]: Since version 5.2, CryPhysics has supported multiple physical entity grids In the case of simulation for a perception system, the most important component is the physics engine, which will allow modeling of the perception system of an autonomous vehicle with less fidelity. Physical simulators work according to the detection of collisions. These differ in the way they react in a collision. They usually work in two ways, where the collision is detected a posteriori or a priori. Collision detection refers to the problem of calculating the detection of the intersection of two or more objects.

As possible physics engines, we found:

- **Open Dynamics Engine (ODE)**[74]: A high-performance, open source library for dynamic simulation of rigid bodies. It is fully equipped, stable and with an easy to use C/C + + platform. It has some very useful methods, such as the method of approaching friction. An advantage is that it is a free and open source. ODE uses a Euler integrator and fixed time stepping. It provides an additional 2D constraint, and has been ported to a large number of platforms.

- **Bullet physics** [75]: A powerful open source physics engine. It differs from other physics engines such as Box2D, Chipmunk, or Sprite Kit's physics engine. This physics engine is 3D and includes 3D collision detection, soft body dynamics, and rigid body dynamics. It also includes a partial graphics processing unit (GPU) for physics implementation.
- **NVidia PhysX**[76]: A very powerful and free physics engine. PhysX is a proprietary middleware or middleware layer engine and a development kit designed to perform very complex physical calculations. Physical middleware engines allow videogame developers to use abstraction during development, as PhysX provides specialised functions in complex physical simulations, which results in high code writing productivity

4.3 Different simulations tools

4.3.1 Sensors

The sensors are the core of the simulation, they are generating data based on the scene and their own configuration. They are covered in chapter 5.2 in more details. The sensors exploit the ground truth (see chapter 3.4) that are generally provided by the Simulation Engine. Low fidelity are "ideal" sensors exploiting directly the ground truth data with no or little added values. More advanced sensor simulations require the scene description to be filled with the correct level of detail in particular the physical properties of the materials. The lack of a correct scene description will render the sensor simulation useless if the sensor requires them. Moreover as describe in the JAMA report [24], the sensor models need to consider a great part of the main disturbances encounter in their operating. These main disturbances, for the 3 main types of automotive sensors, are given in figure 78 for cameras, in figure 79 for LiDAR, and in figure 80 for RADAR.

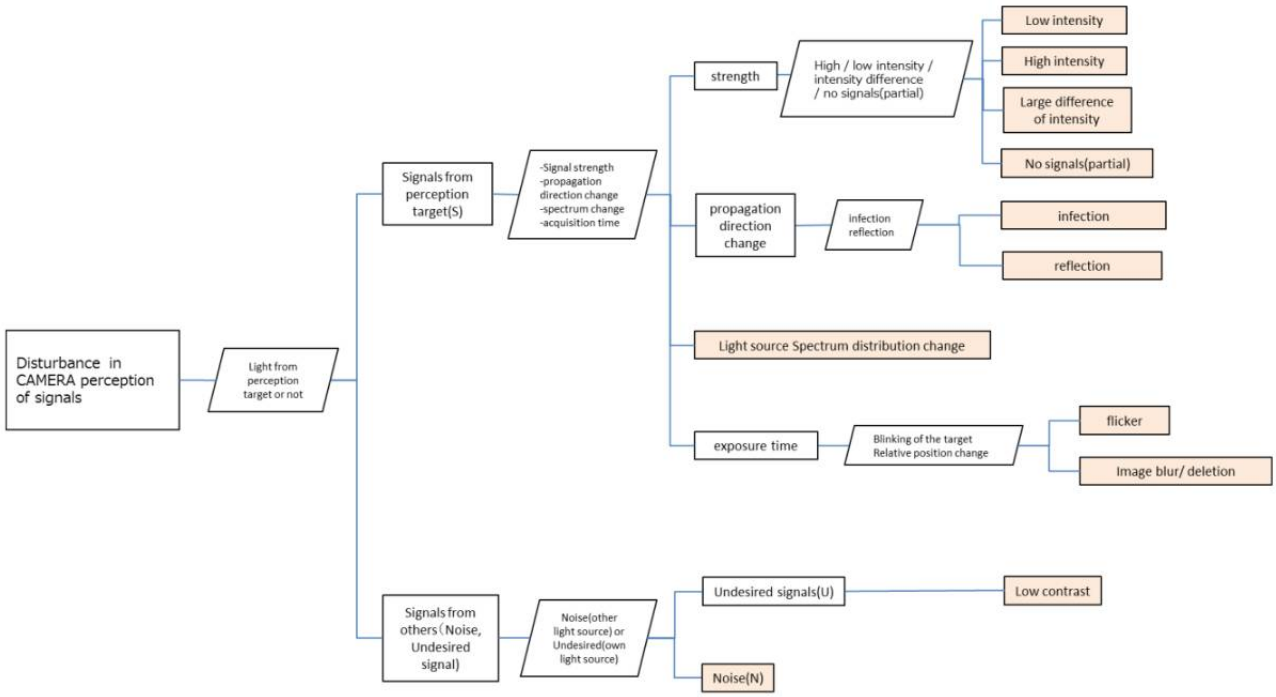

Figure 78: Generation principle of disturbance in camera perception processing (source: JAMAL's report [24])

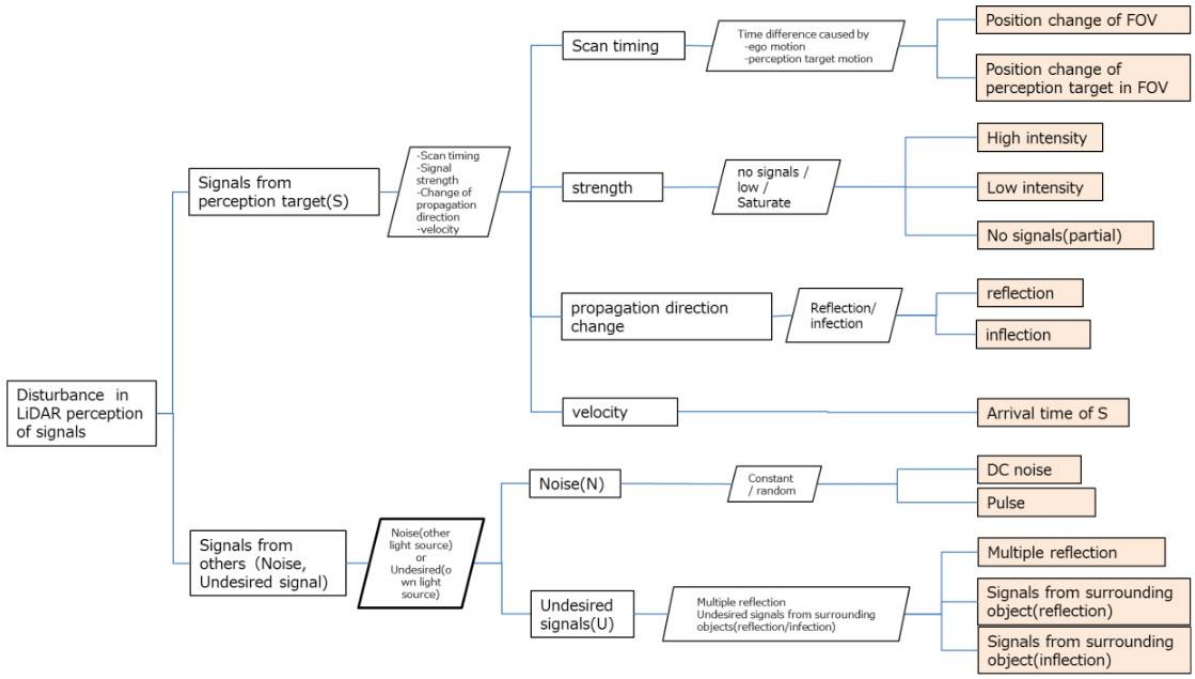

Figure 79: Generation principle of disturbance in LiDAR perception processing (source: JAMAL's report [24])

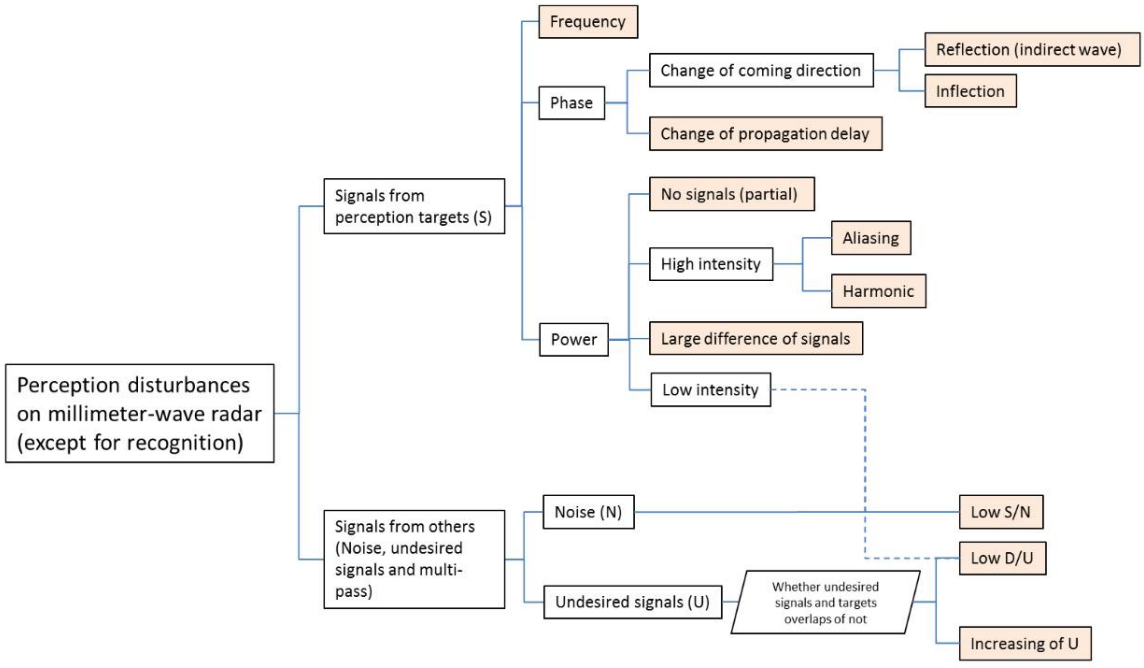

Figure 80: Generation principle of disturbance in millimetre-wave radar perception processing (source: JAMAL's report [24])

4.3.2 Decision-making module

Decision-making module is essential to handle different scenarios in the simulation, it is in charge of the connection between perception of the environment and control of the Autonomous Vehicles(AV) like a vehicular "brain". After the "brain" receives various stimuli and perception information, the decision layer analyses them to understand the current environment and then apply strategies to move according to constraints (safety, energy, comfort, etc.). Then, the decision-making module generates instructions (path, trajectory, orders, manoeuvres, or advice) to the control module. Decision-making algorithm design is required to be tested and validated in the complex and interactive simulation environment(like a multi-agent simulation environment), in view of the fact that the level of complexity of scenarios that a decision-making module can handle is a core indicator for measuring and evaluating autonomous driving capabilities [204].

Different decision-support architectures have been applied to the simulation of autonomous driving, and in the meanwhile, a variety of AI methods have been employed for the different scenarios(intersection, roundabouts, highway, etc.). In a controlled environment, rule and knowledge-based methods have been applied with assuming that the traffic environment and other road users' intent are fully aware. In opposite, the tactical decision task is usually modelled as a Partially Observable Markov Decision Process [205] while the environment is uncertain. The core of the planning-based method is trajectory planning, i.e., generating a trajectory for a given scenario based on the trajectories of other vehicles, but its applicability is still limited in a highly interactive environment. The rapid development of machine learning allowed using learning-based methods for autonomous driving decision-making in highly interactive environments, such as the imitation learning method [206, 207], or more recently a combination between traditional methods and reinforcement learning methods [208].

A virtual platform, which is essentially a simulator, is a key component of Electronic System Level design and verification (ESL). Virtual platforms can not only help making hardware/software architectural trade-offs but also allow early hardware/software integration, with the ability to test autonomous driving software way before real hardware is available. There are languages and tools to assist designing such virtual platforms. SystemC TLM, essentially a Design Specific Language (DSL) based on C++, is a well established languages for that purpose. A non-exhaustive list of tools for ESL includes Cofluent Studio, Synopsys Virtualizer tool set (Platform Creator, Platform Analyzer, SystemC Explorer), Synopsys COMET-METeor (formerly VaST), Imperas OVP (Open Virtual Platform), UNISIM-VP, Qemu, Wind river Simics. Generally, when dealing with simulation, design and decision criterion include the desired level of detail, precision, and representativeness (to make realistic simulation), the flexibility, the speed, the development cost, and the maintenance cost. Indeed, modelling and technological choices have great impacts and implications. Simulation speed impacts the amount of tests, and thus the quality of product because the time budget for testing is limited. While a low level of detail implies high flexibility, high simulation speed, low development and maintenance cost of simulation models, there are drawbacks: are the key characteristics of modelled system still captured? And is behaviour still simulated? Conversely, high level of detail implies high development and maintenance cost, low flexibility, and low simulation speed. Instruction set simulators, and even more virtual platforms (electronic board, System-on-Chip), have generally way too high level of detail to be useful for AI system V&V. Also, in virtual platforms, the emulation of real devices such as CAN, serial, SPI, I²C, and network controllers, have high development costs. However, virtual platform involving virtual devices (instead of truly emulated

devices), such those of Linux VirtIO (Virtual Input-Output) and AUTOSAR MCAL (Microcontroller Abstraction Layer), are more cost effective and can significantly reduce overall development cost. Instruction set simulators and virtual platforms can still have usefulness when AI models, deployed on CPUs or GPUs, deserve a floating-point accuracy analysis. Indeed, some AI models may be sensitive to error introduced with small floating-point binary representation (e.g. Float16 or BFloat16) and more generally, floating-point rounding and absorption, which is tightly geared to underlying hardware implementation and binary code generated at compilation time. Similar techniques, involving instruction set emulation, such as Valgrind with Verrou plugin can also serve for such analyses. As instruction set simulators and virtual platforms are mostly single threaded tools, speed is a common issue, and data-set should be as minimal as possible while maintaining the representativeness of the analysis.

4.3.3 Controlling software

The controlling software covers multiple aspect of the simulation, it can be everything that is not graphical or physical rendering, nor sensor simulation or scenario management. It covers the dynamic model of the various element of the scene, the way the EGO vehicle is moving according to its instruction. It can also include the traffic generation and the control of multiple vehicles in the scene. The pedestrian moves differently from a vehicle and therefore require their own dynamic model. Each dynamic model for car or pedestrian can be more or less advance, and close to the "real" movement of the elements. The goal can be to reproduce exactly a movement and properties of a specific car model from a manufacturer for the more detailed simulations. Another part of Controlling Software is the decision algorithm that process the data generated by the sensor to take decision such as emergency brake. Controlling Software also include the "event" generation described by the scenario that are triggered by data else than a predictive dynamic model, or by the sensor data. This can be a weather change after some time, or when the EGO vehicle reaches a specific position.

4.3.4 Applications and software bricks

Listing of the tools used in autonomous driving simulation and their capabilities. The interface capability covers the scenario import or creation. The data access for EGO vehicle, non-vehicle and pedestrian are covered by the ground truth. Sensor detail and variety cover the sensors emulation. Rendering performance and Physical engine or in the Simulation Engine part. figure 159 shows the list of simulation tools

4.4 Mobile dynamics

In order to develop realistic dynamic vehicle model, it is often necessary to use Physical engine with solvers (ODE), ray-tracing library, and mechanic properties in order to manage physical interaction between several dynamic objects. Moreover, as presented in the figure 81, a dynamic vehicle modelling must involve at least this several modules:

- Car body and chassis with differential system, aerodynamic coefficient
- shock absorbers with either linear, or non linear modelling (potentially with use of LUT)
- Wheels and Tires involving tire grid modelling (longitudinal and lateral)
- Braking system

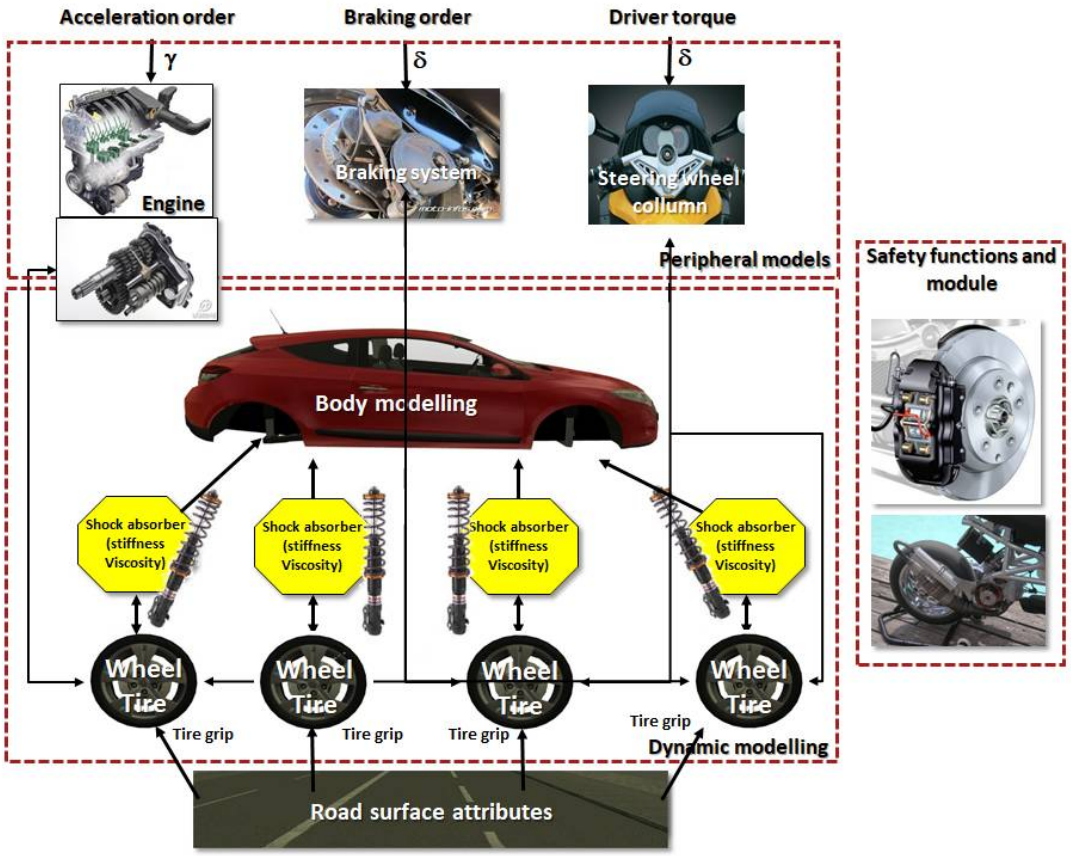

Figure 81: Different parts to consider in a realistic and dynamic vehicle modelling (source: Univ. Eiffel)

- Thermal engine and/or electric engine (involving battery modelling) with power-train and gearbox (automatic or manual)
- Steering wheel column with auto alignment, driver, and controller torques
- main ADAS systems like EPS, ABS, ACC and Stop&Go

Moreover, as mentioned in the JAMA report [24], a set of disturbances must to be considered in order to obtain a physical realistic behavior of a vehicle in realistic and constrain configurations. The figure 84 shows some of these disturbances applied in the ODD of a real vehicle.

4.4.1 Realistic vehicle modelling with Unreal and Unity

For instance Both Unreal Engine 5 and Unity have simplistic vehicle physics based on the standard PhysX implementation. Unity oversimplifies the wheel collider and gives no access to important information (current lateral and longitudinal slip forces) or a way to modify it. In Unity Engine 5, it is possible to have access to the source code, including the complete PhysX engine. So it is possible to modify the vehicle update method, but for the cost of losing sanity when we see another repeated code for each vehicle variant depending on the number of wheels (Vehicle4W, VehicleNoDrive, VehicleTank). Things can get messy if it is necessary to use another engine and to not use the standard engine and gearbox. Unreal Engine does not focus on reliable physics (at least when this information has been collected in end of November 2020). By default, physics simulation is frame rate dependent, so a vehicle may move slower

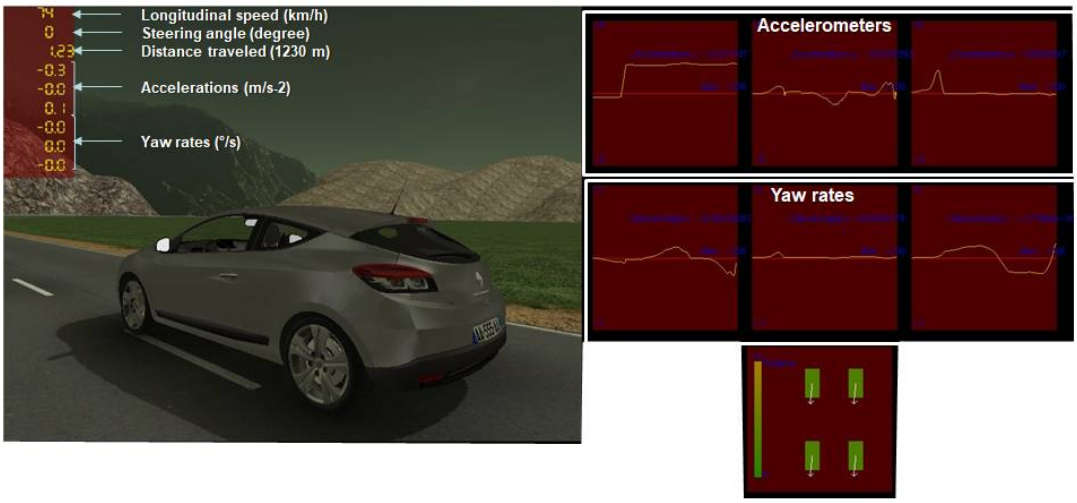

Figure 82: Dynamic vehicle modelling in Pro-SiVIC with car body, shock absorbers, wheels and tires, and power-train (source: Univ. Eiffel)

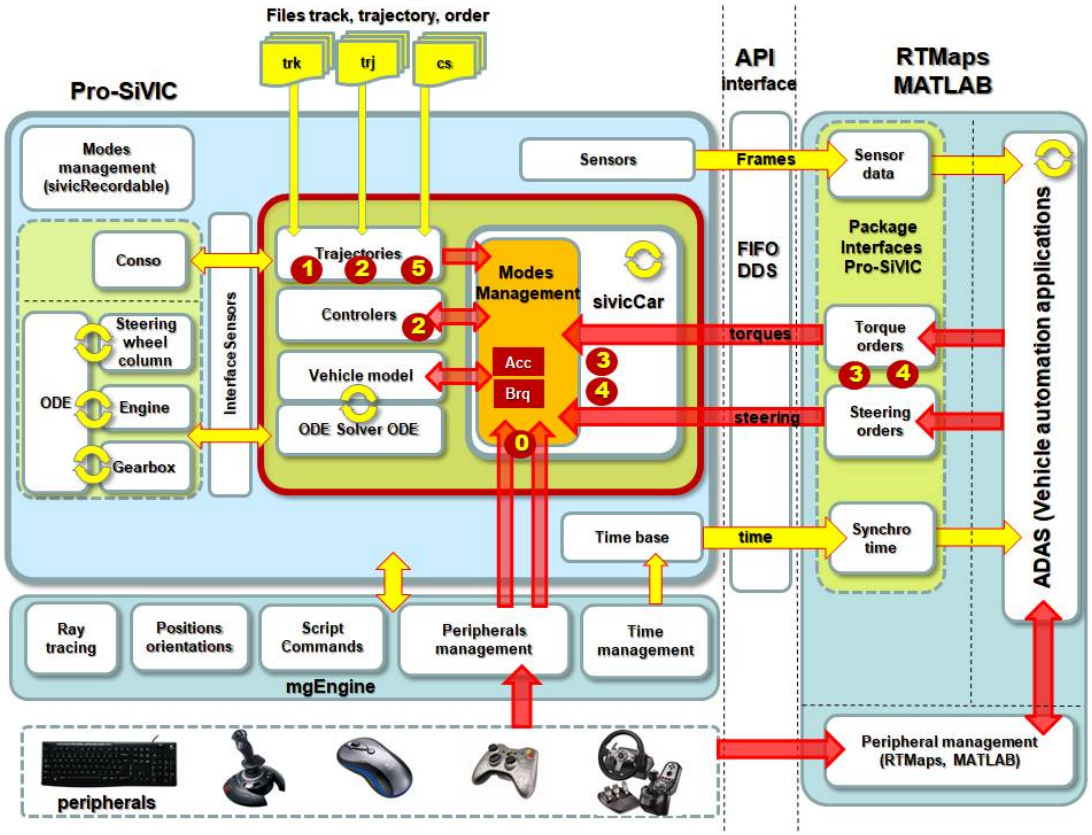

Figure 83: Dynamic vehicle modelling and implementation in Pro-SiVIC (source: Univ. Eiffel)

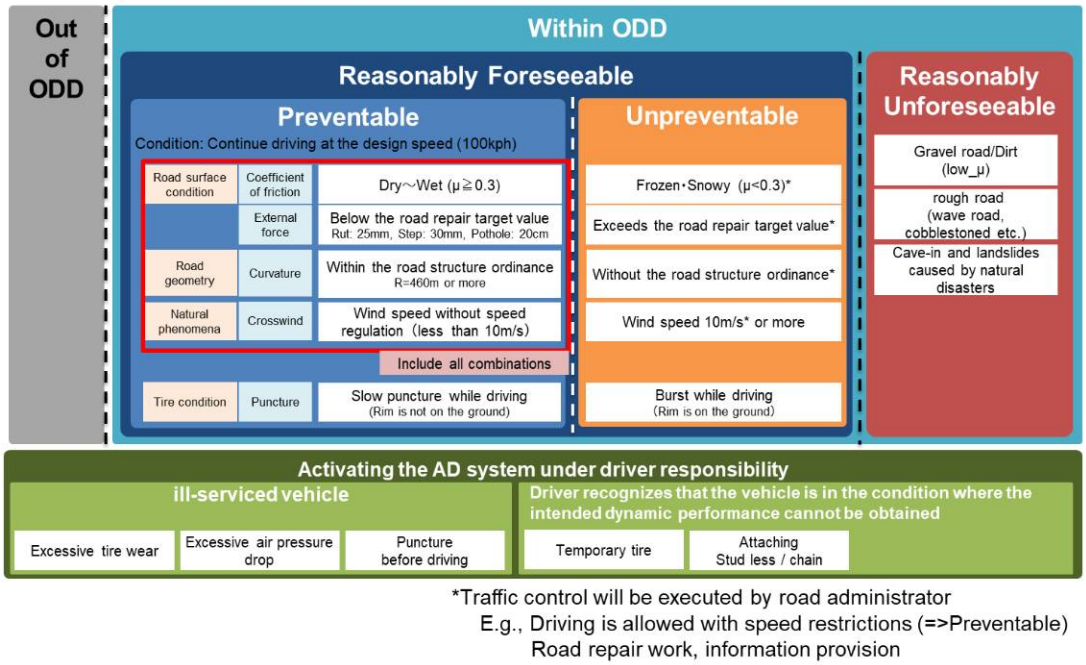

Figure 84: Preventability/Unpreventability boundary conditions in vehicle movement disturbance (source: JAMA's report [24])

on a slower computer. For a project using a vehicle model, it is necessary to modify the Unreal Engine core because the available sub-stepping option does not solve the problem for serious applications. Another interesting information: eXpanSIM is developed in Unity instead of Unreal Engine. eXpanSIM is a universal vehicle simulator with realistic physics of cars, trucks, construction machines, and military vehicles. The simulator is designed for professional applications, which hardcore simulation fans will find interesting. It is available on the Steam Store. (https://store.steampowered.com/app/1015370/eXpanSIM/) For some time now, Unity allows for executing the physics step manually. Managing the physics objects is easy because the integration with the physics engine is tight, and the lifecycle of an object is simple. Just recently, Unity announced that soon it would be possible to switch between the current physics engine and Havok (information from end of November 2019), which is a great plus for Unity. In summary, working with vehicle physics is a challenge itself. The possibility to immediately test a code and human-friendly error logging offered by Unity saves the day. It is essential for fast prototyping and developing a new vehicle physics engine. Unreal Engine may be an option if we are looking for a base implementation of vehicle physics that we only want to modify just a little bit.

4.4.2 Industrial dynamic vehicle modelling

- **Drive (Sate-italy)**:Drive is a dynamic car simulator. It simulates the behaviour of cars during acceleration along their longitudinal axis of symmetry. The coupled dynamics of the suspended and unsprung masses are obtained by calculating the vertical movement and the pitch of the vehicle. The transmission is carefully modeled from the clutch to the tires, including the gearbox. The interaction between the tires and the ground is expressed by a non-linear force and slip diagram, which takes into account the correct calculation of tensile forces, including those with high values of slip. Sate designed other

complementary tools such as *Bench* for suspensions, *Clutch* for clutch, *Condiz* for air conditioning ... These products are mostly based on a MATLAB / Simulink architecture.

- **TruckSim, CarSim and BikeSim**: Tools from CarSim, BikeSim, and TruckSim (Mechanical Simulation Corporation) simulate and animate the dynamics of cars, motorcycles, scooters, racing cars, and trucks, using standard Windows PCs. Mathematic models based on 30 years of university research in vehicle dynamics, accurately simulate braking, driving, stability and acceleration. These tools were designed to communicate with SIL and HIL technologies. An interface with other software such as Simulink, LabVIEW, ASCET and Visual studio is possible. The mathematical model of vehicles is a multi-body model that interacts with the environment through wheel / ground contact and aerodynamic forces. The system has degrees of freedom related to: chassis (longitudinal, lateral and vertical movement, roll, pitch and yaw), wheels (rotation, steering, ...), suspensions (stiffness, damping coefficient, ...) , braking and acceleration (torque, temperature, ABS system, fuel consumption, etc.), tires (lateral and longitudinal sliding, etc.). Several models of tire interaction with the ground are available: Pacejka 5.2, MF-tire, MF-Swift, coupled longitudinal and lateral sliding, external shear with camber.

- **IPG Carmaker**: Carmaker (IPG) is a set of simulation tools for virtual road vehicle driving tests. It can be used offline, in real time as well as in HIL applications. Vehicle components are modelled as rigid or flexible multi-body systems that are non-linear. The multi-body system forms the main model and the integration platform for all other sub-components like suspensions, steering systems, tires, brake system, drive-train and aerodynamics. The models are grouped into sub-assemblies which can be modified or exchanged by internal models developed in Simulink or in language C. The parameters can be done via a graphical user interface. The three-dimensional wheel-ground contact interface uses several models such as IPGTire, MF-5.2, MF-Swift, ...

- **ASM Vehicle Dynamics Simulation Package**: The Vehicle Dynamics Simulation Tool (ASM-VDSP 2007) is a Simulink model for real-time simulation of the behaviour of vehicle dynamics in a given environment. The model can be used on a dSPACE simulator for HIL testing of electronic control units (ECUs) or during the design phase of control algorithms for pre-validation. All blocks are visible, so it's easy to add or replace components with specific models to adapt vehicle properties in different types of projects. Standardised interfaces allow the vehicle dynamics model to be easily extended to meet specific requirements or even create a virtual vehicle. The physical model of the vehicle is represented by a multi-body system with 11 degrees of freedom (6 for the chassis, 4 for the wheels and 1 for the steering). The non-linear model of the vehicle takes into account the geometry and kinematics of the suspension, aerodynamics, a steering model, a gearbox with transmission flexibility, an engine, two semi-empirical tire models (Pacejka and TMEasy) and a hydraulic braking model.

- **Ve-DYNA**: Ve-DYNA (TESIS DYNAware) is software specially designed for the rapid simulation of vehicle dynamics in real-time applications (HIL, SIL) and in computer design studies. The nonlinear vehicle model is based on a modelling concept developed by Rill (Rill 2007) [209] : Vehicle modelling by subsystems. It is modular with separate blocks for the chassis, engine, transmission and the wheels in Simulink. The interaction of tires with the ground is described by the empirical models TM-Easy, Pacejka 5.2, TNO tire and FTire. (http://tesis-dynaware.com/en/products/

vedyna.html). In [210], Georg Rill proposes a new book on Road Vehicle Dynamics: Fundamentals and Modeling with MATLAB.

- **VDL (Dymola)**: VDL, vehicle dynamics library, (Dassault Systèmes) is a tool for modeling, simulating and analyzing the behavior of a road vehicle. It is a tool unique to DYMOLA, an environment of several fields of engineering which contains mechanical, thermal, electrical, pneumatic, hydraulic, thermodynamic components. VDL is based on Modelica, an open source modelling language. The system is described as a multi-body system composed of the chassis, front and rear suspensions, wheels, tires ... Real-time simulation with Dymola gives the ability to run HIL tests on common platforms like dSPACE, RT-LAB, xPC Target and Cramas.

- **ESI group (Pro-SiVIC)**: Pro-SiVIC involves several dynamic vehicle modelling. The first one is based on [211] model with some new functionalities like a powertrain (with engine mapping), a manual and an automatic gearbox, the aerodynamic effects, the fuel tank and consumption sensor, the steering wheel column, and the coupling with a dynamic driving platform. In pro-SiVIC, oscilloscope are available in order to manage the intern variable of the dynamic model. The other dynamic models come from external models with a thin coupling. Pro-SiVIC proposes interfaces with Intempora RTMaps, Effidence AROCCAM, Mathworks Matlab, Simulink, Sylab, and LMS Amesim (now Siemens). For a couple of year, ESI group also propose a cognitive model of driving (COSMODRIVE) interconnected with the dynamic vehicle model of Pro-SiVIC.

- **RaceSim (DATAS)**: The vehicle model is divided into sub-groups. The global model is defined by 1600 parameters, 480 calculated dynamic variables, 15 DoF including a non-linear system of suspensions, tire forces, ... This tool exists in three versions: Expert (F1, CART, GP2, F3000, IRL, NasCar, GT, WRC, Super Touring Car, DTM, Tarmac Rally ....), Standard (Touring Car, advanced F3), special (Dallara F301 –F306).

- **SCANeR ^{TM} studio Motorsport (AV Simulation)**: SCANeR^{TM} studio Car is a tool for assisting in the design of passenger vehicles, racing vehicles, trucks, buses, with or without a trailer. In addition, it is benchmark software for military applications. Developed for automotive experts, it is designed to meet the specific needs of dynamic simulation professionals. The software encompasses, thanks to its modularity, all the varieties of the different components of a vehicle. This platform is based on CALLAS Motorsport. CALLAS (automobile) is a specific application of PROSPER (multi-wheel, multi-modules) for four-wheel vehicles.

- **RACER**:

- **Simcenter Amesim - SIEMENS**: Simcenter proposes a Vehicle System Dynamics Simulation platform and library. Simcenter provides functionalities to design robust chassis components and subsystems like steering and braking components, shock absorbers, active roll stabilizer bars and any mechatronic system related to chassis. Simcenter is a scalable and multi-disciplinary modelling platform. (https://www.plm.automation.siemens.com/global/fr/products/simulation-test/vehicle-dynamics.html)

- **BeamNG.drive**: This platform is not a real industrial software but it is sell as a game. Nevertheless it is an very interesting software with very realistic vehicle behaviors (car,

4DW, bus, truck ...) with both realistic dynamic vehicle modeling and realistic interactions between vehicle meshes and environment (physical engine). BeamNG.drive is a physico-realistic driving game with very wide possibilities. The ”soft-body” physics engine developed in this platform simulates each component of a vehicle in real time, which allows to obtain a result faithful to reality. BeamNG's physics engine is at the heart of the highly detailed vehicle simulation and seems to be one of the best physics engines dedicated to ”soft body” dynamics in a game. The effects of crashes are very realistic this software uses a damage model extremely precise.
(https://store.steampowered.com/app/284160/BeamNGdrive/)

4.4.3 Academic dynamic vehicle modelling

About Academic dynamic vehicle modelling, the PhD thesis of Laetitia Li (2021) [212] gives a good overview of car body, wheels, and tires modelling. In this work, cinematic models are firstly presented then after, a set of 4 different dynamic model are enumerated and presented:

- Dynamic bicycle modelling with 3 DoF
- Dynamic 4 wheels modelling with 7 DoF
- Dynamic 4 wheels modelling with 10 DoF
- Dynamic 4 wheels modelling with 14 DoF

About the tire grip modelling, the main well-known models are:

- Model of Brosse
- Model of Gim
- Model of Dugoff
- Model of Kiencke/Burckhardt
- Model of Pacejka (full and simplify (order 3))

In [213], an overview is given about the modeling of drive-train components:

- Combustion engine
- automatic gearbox
- Limited slip differential
- braking system
- Steering column

In the same work, the main stability controllers are presented:

- Anti-lock Braking System
- Traction Control System

- Electronic Stability Control

In [211], Sébastien Glaser propose a full enough modelling of the dynamic of a vehicle for the simulation in limit conditions. In this model implemented in Pro-SiVIC platform, the following parts are implemented:

- The car body using the fundamental principles of dynamics
- The shock absorbers
- the anti-roll bars
- The wheels and tire with Dugoff and Pacejka models

This model has the capability to be adapted to some other types of vehicles like truck and bus. This model has been validation with experimental data and with the CALLAS platform. Now in Pro-SiVIC some other functionalities have been added like a powertrain (with engine mapping), a manual and an automatic gearbox, the aerodynamic effects, the fuel tank and consumption sensor, the steering wheel column, and the coupling with a dynamic driving platform. In pro-SiVIC, oscilloscope are available in order to manage the intern variable of the dynamic model.

4.5 Physical and realistic simulation of adverse and degraded conditions

To reach fully automated vehicles, which represent the level 5 of the Society of Automotive Engineers (SAE) classification of driving automation, the sensors must be tested and validated to ensure that all required conditions, including all kind of adverse weather conditions, are statistically met during the test phases. Regardless of the degraded weather conditions, current vehicle perceptive sensors are more or less impacted by adverse weather conditions and may not correctly detect objects or pedestrians in their path and the surroundings of the vehicle, leading to potential fatal accidents.

Hence it is crucial to be able to test those different sensors in harsh weather conditions to get rid of the actual limitations of advanced driver-assistance systems due to degraded sensors performances.

In Pro-SiVIC, the main set of weather conditions are modelling and provides. Among this models, different clusters of sensors can be done and can be mentioned depending of the type of sensors and the wave length used by a specific sensor (see 85). For instance, camera will be clearly impacted and disturbed by light sources, rain drop, and fog (figure 86). RADAR will be disturbed by interference, problems of material structure, and conductivity of material. Moreover depending of the RADAR frequency, it could or not see objects with specific size. About IR sensors, glasses and material heat will provide adverse conditions or data error (invisible objects). For the LiDAR, the particles in the air, depending of their size and their features, will cause a significant modification of the beam state (energy).

4.5.1 Adverse weather conditions

Even though adverse weather conditions do not represent the majority of the weather part of scenarios an autonomous vehicle may encounter, their impacts on driving automation are major and not yet sufficiently documented.

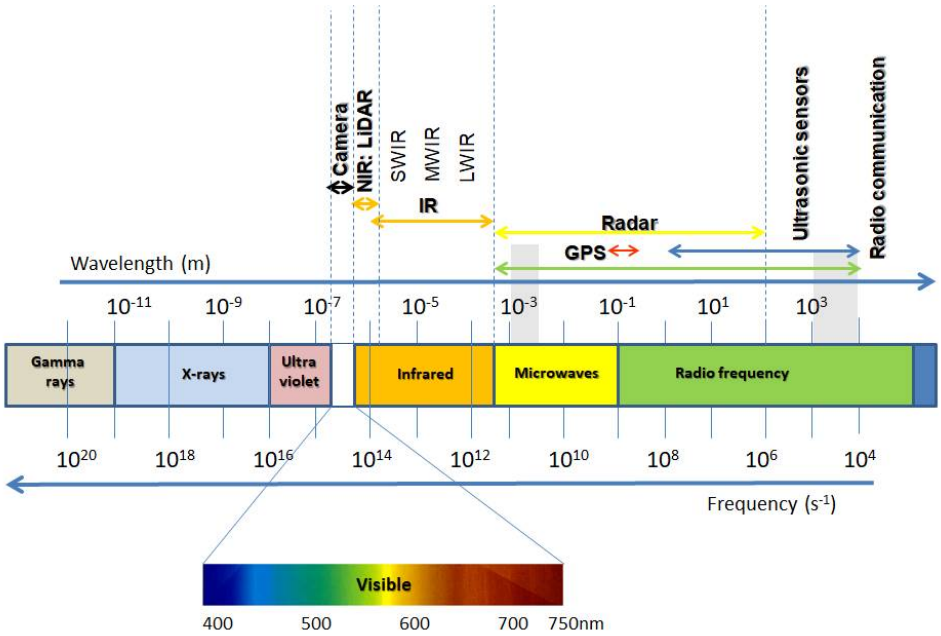

Figure 85: Automotive sensors wave length.(source Univ. Eiffel)

Figure 86: Camera with daylight with adverse and degraded conditions: homogeneous and non homogeneous fog, rain drops on the camera glass (source: Univ. Eiffel)

A list and a description of the different adverse weather conditions is presented in this section, followed by a summary of their impact on the perceptive sensors of the autonomous vehicles.

∗ **Different types of adverse weather conditions**

Adverse weather conditions mainly consist in the presence of particles and droplets in the field of view of surround sensors. In addition to atmospheric attenuation, the electromagnetic waves are more or less scattered and absorbed depending on the amount of hydrometeors they meet. Scattering and absorption vary in relation with the characteristics of the particles encountered (i.e. droplet size, solid or liquid particles, droplet distribution).

Here is a none exhaustive list of adverse weather conditions that an autonomous vehicle can encounter [214]:

- *Mist and fog* are the result of the condensation of water vapour on atmospheric nuclei remaining in suspension close to the ground. Mist and fog are classified based on the visibility, or Meteorologic Optical Range (MOR). This latter corresponds to the length of path in the atmosphere over which the light, from a known source, is reduced to 5% of its original intensity. We talk about fog when the visibility is below 1 km. When the visibility is greater than 1 km we talk about mist. Fog droplets size range from a few tenths of a micron to a few tens of microns [215]. The visibility decreases with the increase of droplets number in the medium and also depends on the droplet size distribution [216]. A same visibility value can correspond to different droplets size distribution.

- *Dust, natural haze or smog* occur by the accumulation of particles of dust, smoke or any air pollutant, in relatively dry air. The extinction coefficient of haze is wavelength dependant. Zhang et al. (2021) [217] pointed out an adverse weather condition typical in East Asia during spring months, named Asian dust, which is made of mineral dust from crustal sources transportation from desert areas eastward.

- *Precipitations, including rain, snow, hail and sleet*, consist in liquid or frozen water drops falling to the ground. It is the result of water vapour condensation on particles in the colder atmosphere. The intensity of the precipitations is defined by the size and the distribution of the droplets in the medium. The visibility impairment due to precipitations is also related to the speed of fall of droplets which causes the appearance of streaks linked to the camera shutter. The diameter of raindrops can range between 0.1 to $\sim$ 6 mm. In scattering and absorption calculations, raindrop shape is often assumed to be spherical even though large droplets typical of heavy rains look like oblate spheroidal [218]. Unlike rain drops, snow grains cannot be considered as spherical. Their shape and size are complex and hence difficult to simulate numerically. Räisänen et al. constructed a reference phase function for the scattering of snow grains, taking into account the diversity of snow grains shape and size [219].

∗ **Impacts on perceptive sensors performances**

The purpose of the sensors in ADAS (Advanced Driver Assistance Systems) is, among other things, to control the speed of the vehicle in its environment, to recognise the road and traffic signs and also to avoid collisions with any obstacles. Ideal sensors would not be impacted by adverse weather conditions, but actual sensors performances are impacted with different level of severity. Mohammed et al (2020) present spider charts for each type of sensors performances

scores including the impact of adverse weather conditions. The sensors with good performances in adverse weather conditions are radar, far infrared and ultrasonic sensors, on the opposite, LiDAR and camera have weak performances in adverse weather conditions [220].

As previously mentioned, the impact of adverse weather conditions on electromagnetic waves depends on the characteristics of the hydro-meteors and particles encountered but also on the wavelength range of the different sensors of the autonomous vehicle (See 85). Figure 85 highlights the different spectral ranges of the sensors. Each sensor has a specific task and will be impacted differently when exposed to adverse weather conditions such as mist, dust and rain.

An electromagnetic wave interacting with particles in the atmosphere is attenuated. It is characterised by the extinction coefficient of the medium. This extinction coefficient is the addition of several phenomena : scattering and absorption. Scattering can be defined in three different ways depending on the particle size and the wavelength range of the sensors:

- Rayleigh scattering for particle with very small size compared to wavelength. (Ex : In case of fog conditions for RADAR).
- Mie scattering theory : particle size is similar to wavelength. (Ex : Fog conditions for LiDAR and Camera, and in rainy conditions for RADAR).
- Optical geometry : the wavelength is small compared to particle size. (Ex : For rainy conditions for LiDAR and Camera).

Thus, the attenuation is also related to rainfall rate and drop fall speed, snowfall rate in the case of precipitations based on recommendations of the ITU-R (International Telecommunication Union) [221], and the size distribution and size of droplets/particles.

The problematic of quantification of the effects of adverse weather conditions on ADAS sensors is the topic of numerous papers in the literature. Zang et al (2019) [222], Vargas et al. (2021) [214] give an overview of the effects of adverse weather conditions on the most representative sensors of autonomous vehicles ecosystem, such as LiDAR, global positioning system (GPS), camera and radar.

- ***CAMERA***

  The main effect of adverse weather conditions on camera is the loss of contrast making it more difficult to identify or detect objects in the images.

  Hasirlioglu (2020) [223] conducted a study of the influence of rain and fog on the detection of objects by camera, LiDAR and RADAR. Fog and rain were simulated in real world, using CARISSMA indoor facility, and in virtual world, using noise models. The results highlight that the three sensors are affected by rain with more or less severity. Camera images are affected by a reduced contrast.

  Xique et al. (2018) [224] collected data from three different sensors including a camera in different weather conditions. Their results show a degradation of the performances of the camera by 55 to 60 % for rain and snowfall, respectively, compared to empirical measurements.

  Garg and Nayar (2005) [225] highlight that the effect of rain on camera images is associated with both rain drop characteristics and camera parameters, such as depth of field and

exposure time.

- ***LIDAR***

  LiDAR main wavelengths are 905 nm, with a limited detection range of $\sim$ 100 m due to eye safety power restrictions, and 1550 nm, with a detection range from 200 to 300 m. Wojtanowski et al. (2014) [226] conducted a study on the influence of adverse weather conditions on LiDAR rangefinder's performance deterioration. When 1550 nm LiDAR gives better performances than 905 nm one for object detection in dry situation, its performance is much more affected when the humidity increases. In addition to the intensity reduction of the target signal in adverse weather conditions, backscattering from raindrops may lead to the detection of droplet as an object which is named false positive detection. False negative detection are correlated to laser beam size and aperture of the LiDAR system.

  Adverse weather conditions such as rain and fog represent a soft target for LiDAR. The pulse is scattered in fog or rain and the sensor gets the distributed scattering of the initial pulse. The extinction efficiency and the backscattering efficiency depend on the diameter of the droplets. The presence of powder snow, heavy rain and strong fog modify the information received by active sensors by creating a screen that can lead to wrong interpretations like unwanted detection (i.e. false positive) for LiDAR sensors in the near field (less than 10 m from the observer) or missing true positive scan points([227], [228]).

  LiDAR are "suitable for environments with transmissions exceeding 92%–93%/m for targets closer than 25 m" [214].

- ***RADAR***

  RADAR technology uses microwaves to detect objects in the surrounding of the vehicle within a certain range. There are two kinds of automotive RADAR : long-range radar to detect long range objects, such as fast-moving vehicles, thanks to a large angular resolution but a small field of view, while short-range RADAR will have large angular resolution and field of view. Short-range RADAR are used to detect vulnerable road user or objects close to the autonomous vehicle.

  The performances of RADAR are evaluated in term of maximum range, maximum Doppler velocity and field of view. The frequency of the RADAR has an influence on the perturbation of the signal in adverse weather conditions. An electromagnetic wave travelling through rain is absorbed, depolarised, scattered, and delayed in time but the main two effects are attenuation and backscattering.

  RADAR is overall the sensor least affected by adverse weather conditions, especially for fog conditions [223]. Its range profiles detect more near-field reflections and less far-field reflections. Nevertheless, Zang et al. (2019) [222] suggest an attenuation of the radar signal of 55% in the presence of heavy rain (150mm/h). Vargas et al (2021) [214] concluded that RADAR are relatively unaffected for rainfall of 50-70 mm/h or dust with a visibility lower than 10 m.

$*$ **Sensors surface and secondary effects of adverse weather conditions**

	Short Range Radar	Long Range Radar	LiDAR	Camera	IR Camera
Operation in Rain	++	+	O	O	O
Operation in Fog or Snow	++	++	-	-	O
Operation if Dirt on Sensor	++	++	++	- -	- -
++: Ideally suited, +: Good performance, O: Possible, drawbacks, -: Bad, - -: Impossible					

Figure 87: Influence of adverse weather conditions on automotive sensors [25].

Aside from the impact of particles in the medium in the field of view of the sensor, adverse weather conditions such as rain and snow modify the characteristics of surrounding materials. Water and snow remaining on the roadway modify the perception of the lane lines of the road. Water layer on the road can produce some spaying effects caused by surrounding vehicles. Ice and rain can cover the road signs making them impossible to identify. Rain and snow can partially occlude the lenses and hence block part of the emission from LiDAR beams for example [229] or change the focus of camera [222].

From all the automotive sensors, cameras are the less suitable in adverse weather conditions. RADAR appears to be the most robust sensors in adverse weather conditions [230]. Nevertheless RADAR can still be disturbed by the presence of water on lenses or targets and wet ground.

In Figure 6 of Song et al. (2020) [25] a summary of the influence of adverse weather conditions based on a study of Rasshofer and Gresser (2005) [230] is presented :

4.5.2 Impact of adverse weather modeling

In order to model the impact of adverse weather conditions on perceptive sensors performance, one needs to model the path of a light ray interacting with the particles in the air and the materials of the road scene. The propagation of electromagnetic waves in participating media, such as fog, rain, snow or dust, is governed by the radiative transfer equation (RTE) in which the optical parameters (related to scattering, absorption and extinction) of the medium are considered. The RTE serves to simulate the radiance $L_\lambda(t, r, u)$ corresponding, for a wavelength λ, to the intensity of the electromagnetic energy flux (in W) of the radiation propagating in the direction u, per unit of area (in m^2) perpendicular to the direction of propagation, per unit of solid angle (in sr) and per unit of wavelength (in microns), and expressed in $W.m^{-2}.\mu m^{-1}.sr^{-1}$. The RTE can be expressed as [231]:

$$\frac{1}{c}\frac{\partial L_\lambda}{\partial t}(t, r, u) + u \cdot \nabla_r L_\lambda(t, r, u) =$$
$$-\sigma_\lambda(r)L_\lambda(t, r, u) - \kappa_\lambda(r)L_\lambda(t, r, u) + \frac{\sigma_\lambda(r)}{4\pi}\int_{\mathbb{S}^2} L_\lambda(t, r, v)\Phi_\lambda(r, v, u)dv + q(t, r, u), \quad (20)$$

where c is the speed of light, t, r, u, σ_λ, κ_λ, Φ_λ and $q(t, x, u)$ denote, respectively, the time, the position in space, the wave propagation direction, the scattering coefficient, the absorption

coefficient, the phase function for the wavelength λ and light sources (including thermal sources if thermal imaging is of interest). The three-dimensional unit sphere is denoted by $\mathbb{S}^2$. Optical passive objects and local light sources are taken into account thanks to the boundary conditions of Equation (20). For each light source occupying the space region S and emitting light from its surface ∂S, we have:

$$\forall\, r \in \partial S,\ \forall\, u \in \mathbb{S}^2,\ u \cdot n_r^S > 0,\ L_\lambda(t, r, u) = E_\lambda^S(t, r, u), \tag{21}$$

where n_r^S denotes the outward normal vector of S at point $r \in \partial S$, and E_λ is given. For each passive object occupying the space region O with a surface denoted by ∂O, we have:

$$\forall\, r \in \partial O,\ \forall\, u \in \mathbb{S}^2,\ u \cdot n_r^O > 0,\ L_\lambda(t, r, u) = \int_{v \in \mathbb{S}^2,\ v \cdot n_r^O < 0} L_\lambda(t, r, v) B_\lambda^O(r, v, u) dv, \tag{22}$$

where n_r^O denotes the outward normal vector of O at point r, and $B_\lambda^O(r, \cdot, \cdot)$ is the *Bidirectional Reflectance Distribution Function* (BRDF) of object O at point $r \in \partial O$. When the time can be removed from the physics and under the assumption of a phase function depending only on $v \cdot u$ (scalar product), the following stationary case of Equation (20) can be considered:

$$u \cdot \nabla_r L_\lambda(r, u) = -\beta_\lambda L_\lambda(r, u) + \frac{\sigma_\lambda}{4\pi} \int_{\mathbb{S}^2} L_\lambda(r, v) \Phi_\lambda(v \cdot u) dv + q(r, u), \tag{23}$$

where we note $\beta_\lambda = \sigma_\lambda + \kappa_\lambda$ the extinction coefficient at the wavelength λ. Boundary conditions related to this stationary case are given by Equations (21) and (22) in which the time t is removed.

The particle models are important to determine the optical properties of the medium (fog, rain, snow, dust, smoke): σ_λ, κ_λ, $\beta_\lambda = \sigma_\lambda + \kappa_\lambda$ and Φ_λ. Particle models are based on the particle size distribution (PSD) and the complex refractive index of the particles. A PSD is a function N (cm^{-3} $\mu\mathrm{m}^{-1}$) such that $N(r)\, dr$ represents the number of particles contained in a volume of 1 cm^3 whose radii belong to $(r, r + dr)$. In the case of spherical particles, the Lorenz-Mie scattering model is largely used. The Lorenz-Mie theory [232] solves the electromagnetic equations of Maxwell for a spherical particle of radius r with a given complex refractive index $m_p = n_p + ik_p$ embedded in a host medium with refractive index $m_h = n_h + ik_h$. Under the assumption of non-dependent scattering between particles, the extinction and scattering coefficients are then expressed in terms of the PSD N as follows:

$$\sigma_{ext}^\lambda(N) = \int_0^{+\infty} Q_{ext}^\lambda(r)\, \pi\, r^2\, N(r)\, dr \qquad ; \qquad \sigma_{sca}^\lambda(N) = \int_0^{+\infty} Q_{sca}^\lambda(r)\, \pi\, r^2\, N(r)\, dr. \tag{24}$$

We also note that the absorption coefficient is defined as:

$$\sigma_{abs}^\lambda(N) := \sigma_{ext}^\lambda(N) - \sigma_{sca}^\lambda(N) = \int_0^{+\infty} Q_{abs}^\lambda(r)\, \pi\, r^2\, N(r)\, dr, \tag{25}$$

where

$$Q_{abs}^\lambda(r) := Q_{ext}^\lambda(r) - Q_{sca}^\lambda(r).$$

Similarly, the phase function can be expressed by the following form:

$$\sigma_{sca}^\lambda(N)\, \Phi_\lambda(\mu, N) = \int_0^{+\infty} Q_{sca}^\lambda(r) \psi_\lambda(r, \mu)\, \pi\, r^2\, N(r)\, dr, \tag{26}$$

where the scattering efficiencies and extinction efficiencies are given by:

$$Q^{\lambda}_{sca}(r) = \frac{\lambda^2}{2\,\pi^2\,r^2}\sum_{n=1}^{+\infty}(2n+1)\left(|a_n(r,\lambda)|^2 + |b_n(r,\lambda)|^2\right), \tag{27}$$

$$Q^{\lambda}_{ext}(r) = \frac{\lambda^2}{2\,\pi^2\,r^2}\sum_{n=1}^{+\infty}(2n+1)Re\left(a_n(r,\lambda) + b_n(r,\lambda)\right), \tag{28}$$

and ψ_λ given by

$$\psi_\lambda(r,\mu) = \frac{\lambda^2}{2\,\pi^2\,r^2\,Q^{\lambda}_{sca}(r)}\left(|S_1(\mu)|^2 + |S_2(\mu)|^2\right). \tag{29}$$

S_1 and S_2 are the scattering amplitude functions given by:

$$S_1(\mu) = \sum_{n=1}^{+\infty}\frac{2n+1}{n(n+1)}\left(a_n(r,\lambda)\pi_n(\mu) + b_n(r,\lambda)\tau_n(\mu)\right), \tag{30}$$

$$S_2(\mu) = \sum_{n=1}^{+\infty}\frac{2n+1}{n(n+1)}\left(b_n(r,\lambda)\pi_n(\mu) + a_n(r,\lambda)\tau_n(\mu)\right), \tag{31}$$

where the sequence of polynomials $(\pi_n)_{n\geq 0}$ and $(\tau_n)_{n\geq 0}$ are defined by the recurrences:

$$\begin{cases}\pi_0(z) = 0,\ \pi_1(z) = 1,\\ \forall\, n \geq 2\,, \pi_n(z) = z\dfrac{2n-1}{n-1}\,\pi_{n-1}(z) - \dfrac{n}{n-1}\,\pi_{n-2}(z),\end{cases}$$

$$\begin{cases}\tau_0(z) = 0,\ \tau_1(z) = z,\\ \forall\, n \geq 2\,, \tau_n(z) = z(\tau_n(z) - \tau_{n-2}(z)) - (2n-1)(1-z^2)\,\tau_{n-1}(z) + \tau_{n-2}(z).\end{cases}$$

The coefficients a_n and b_n in equations (27) and (28) are complex numbers called the Lorenz-Mie coefficients, which are defined thanks to spherical Bessel functions. For more details on a_n and b_n, we refer to [232].
The numerical computations of the series introduced above require a truncation. The most commonly used truncation, taking into account the numerical difficulties encountered with Bessel functions, is that of Wiscombe [233]:

$$E(v) = \begin{cases} v + 4v^{1/3} + 1 & \text{if} \quad 0.02 \leq v \leq 8,\\ v + 4.05v^{1/3} + 2 & \text{if} \quad 8 < v \leq 4200,\\ v + 4v^{1/3} + 2 & \text{if} \quad 4200 < v \leq 20000,\end{cases} \tag{32}$$

where $E(v)$ is the truncation function of the size parameter $v = 2\pi r/\lambda$.

The physical significance of the phase function Φ_λ is important. Indeed, for a photon moving at the speed of light in a medium, the phase function gives the probability of the resulting direction of the photon when it interacts with a scattering particle (water drop, snowflake, dust aerosol or molecule of the air). The phase function Φ_λ for six radii of spherical water droplets and different wavelengths from the visible to the thermal infrared range is shown in polar coordinates in Figure 88. One angle $\theta = u \cdot v$ is sufficient to represent this phase function due to

the spherical symmetry. We can notice a very weak influence of the wavelength on the phase function for small spheres (r = 0.05μm and r = 0.2 μm), which is in accordance with Rayleigh's theory under unpolarized incident light [231, 234]. On the other hand, for sphere radii beyond 0.5 μm, the influence of the wavelength is noticeable, and backscattering gradually disappears. Finally, it should be noted that all of the curves presented in Figure 88 consider the variation in the complex refractive index of water according to the wavelength.

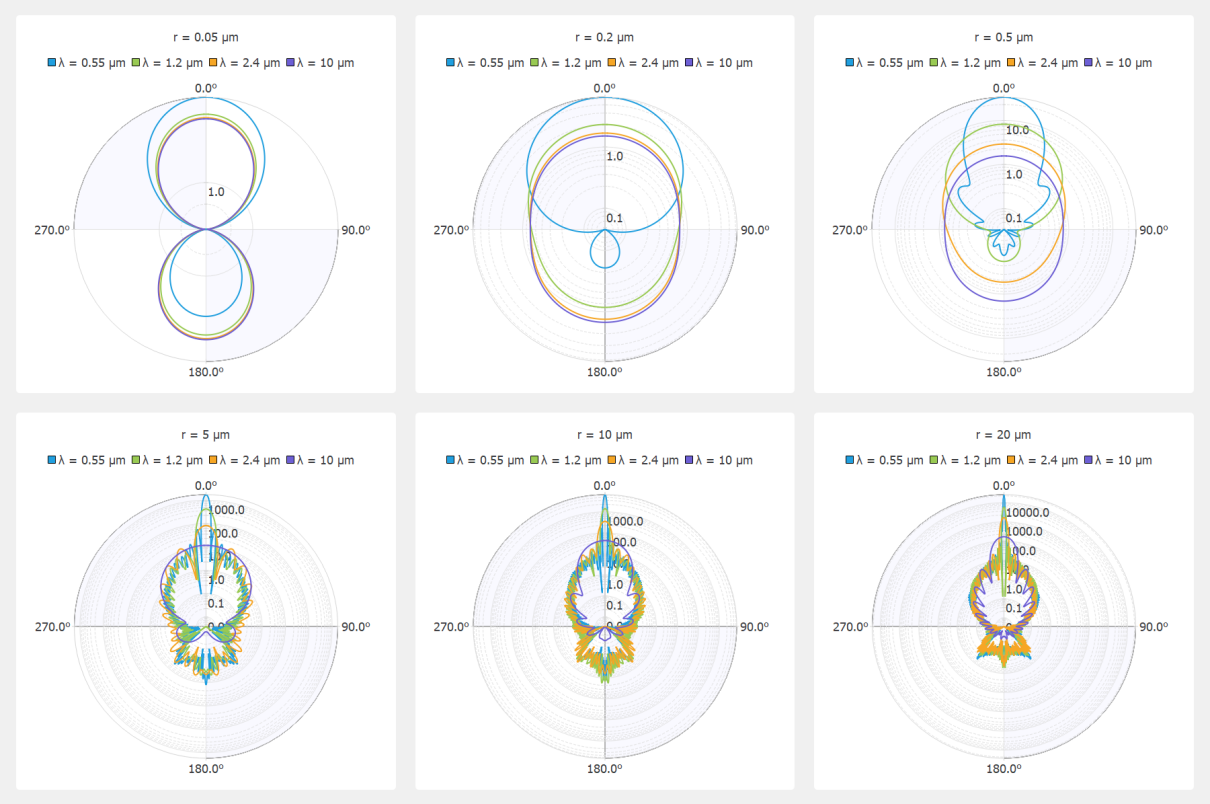

Figure 88: Polar representation of the phase function for six radii (r) of spherical particles and different wavelengths (λ).

An important remark concerns the dependence of the Mie coefficients a_n and b_n on the size parameter $v = 2\pi r/\lambda$. When $v \ll 1$, Mie scattering is well approximated by the Rayleigh scattering. For $v \ll 1$, the laws of optical geometry (Snell-Descartes laws) can be used and in the case of $v \sim 1$, the Mie theory has to be employed. Note that the Mie regime for a rain drop whose size ranges in (0.5 mm,7 mm) corresponds to electromagnetic frequencies between 40 Ghz and 600 GHz. Automotive radars operating around 77 GHz, Mie scattering theory is then relevant for the radar performance assessment in rainy conditions.

The RTE (20) can be simulated thanks to a Monte-Carlo based algorithm (ray tracing) which is reputed to require a lot of computing time. A particular case of Equation (23) is often used in a way to achieve an analytical solution. It consists of eliminating the collision (integral) term in (23) and assuming a constant source q. In this case, assuming there is no object and no local source between points r_0 and $r = r_0 + xu$ for x a real and $u \in \mathbb{S}^2$, we have:

$$L_\lambda(r_0 + xu, u) = L_\lambda(r_0, u)e^{-\beta_\lambda x} + \frac{q}{\beta_\lambda}\left(1 - e^{-\beta_\lambda x}\right), \tag{33}$$

leading to the Beer–Lambert solution if $q = 0$:

$$L_\lambda(r_0 + xu, u) = L_\lambda(r_0, u)e^{-\beta_\lambda x}. \tag{34}$$

The simple case presented above corresponds to the framework of the Koschmieder theory [235, 236] allowing the contrast between a black object and a sky background to be evaluated based on visibility attenuation due to the extinction of the medium between the object and the observer. This theory is used in image processing to artificially add fog to an image: the intensity $I(x, y)$ of a pixel (x, y) is linked to the intensity $I_0(x, y)$ without fog and an air–light intensity I_s:

$$I(x, y) = I_0(x, y)\, e^{-\beta d(x,y)} + I_s \left(1 - e^{-\beta d(x,y)}\right), \tag{35}$$

where $d(x, y)$ is the real-world distance between the observer (camera) and the real point associated with the pixel (x, y), and β is the extinction coefficient of the medium for the visible range ($\lambda \simeq 550$ nm). The use of this simplified modelling of (23) needs only the knowledge of the extinction coefficient (or equivalently the meteorological visibility), without having to know the PSD. We illustrate how the use of PSD (and then of the phase function Φ_λ) can be necessary for the simulation of radiation propagation in fog. The choice of the RTE modelling is important for the simulated fog added to a clear image. The images in Figure 89 are obtained with the Cerema Monte-Carlo based SWEET simulator [237] without/with fog (a and b) and with the Koschmieder model for the same visibility and the same airlight radiance (c). A blur effect can be noticed in the SWEET image, which is not the case for the Koschmieder image. As we can notice in Figure 89c, Koschmieder's model brightens the foggy image much more than SWEET (Figure 89b). This can be critical because some objects in the scene are not even visible with the SWEET simulation and are partially visible with the Koschmieder model (e.g., the two pedestrians on the right).

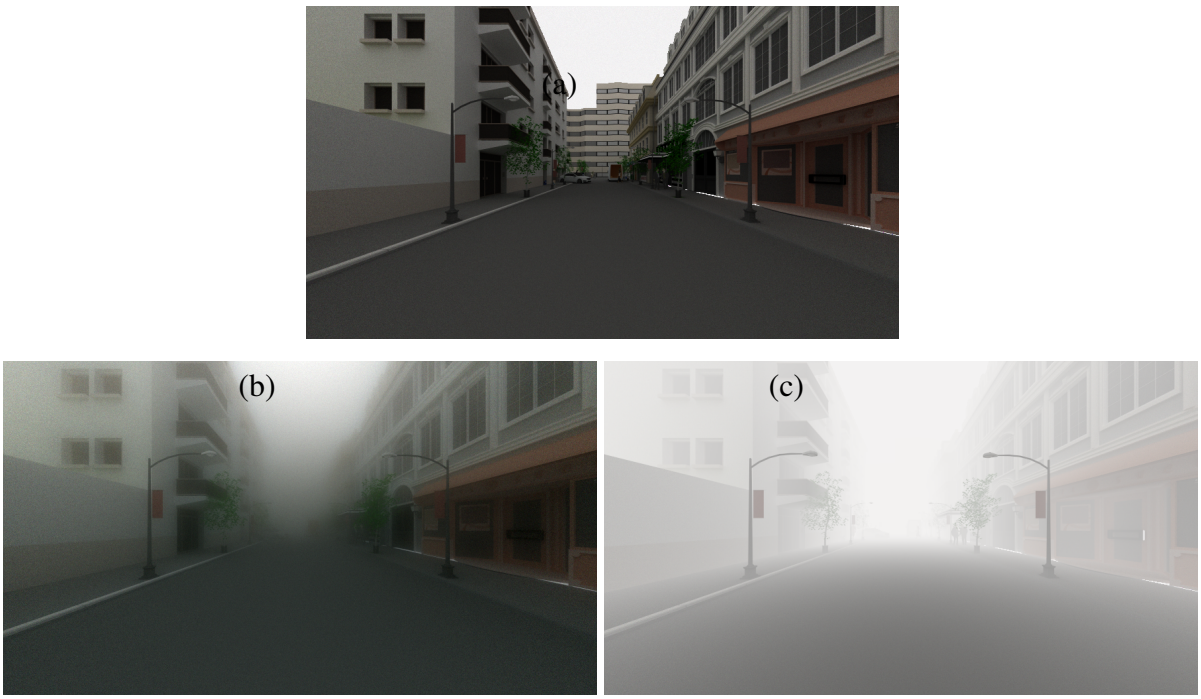

Figure 89: Simulated images for the intra-urban scene with the SWEET simulator without fog (**a**) and with fog (MOR = 20 m, (**b**)) and with the Koschmieder model (**c**) in day conditions.

In [238], results on experiments made in the Cerema Fog and Rain PAVIN platform show the sensitivity of extinction to the PSD of fog droplets. The simulated radiance (or intensity) thanks to the RTE (23) with two different PSD are showing in Figure 90). It can be observed different behavior of extinction for two fog PSD corresponding to the same meteorological visibility. We can remark theat the sensitivity depends on the wavelength (visible range at 0,55 μm and thermal infrared range at 12 μm).

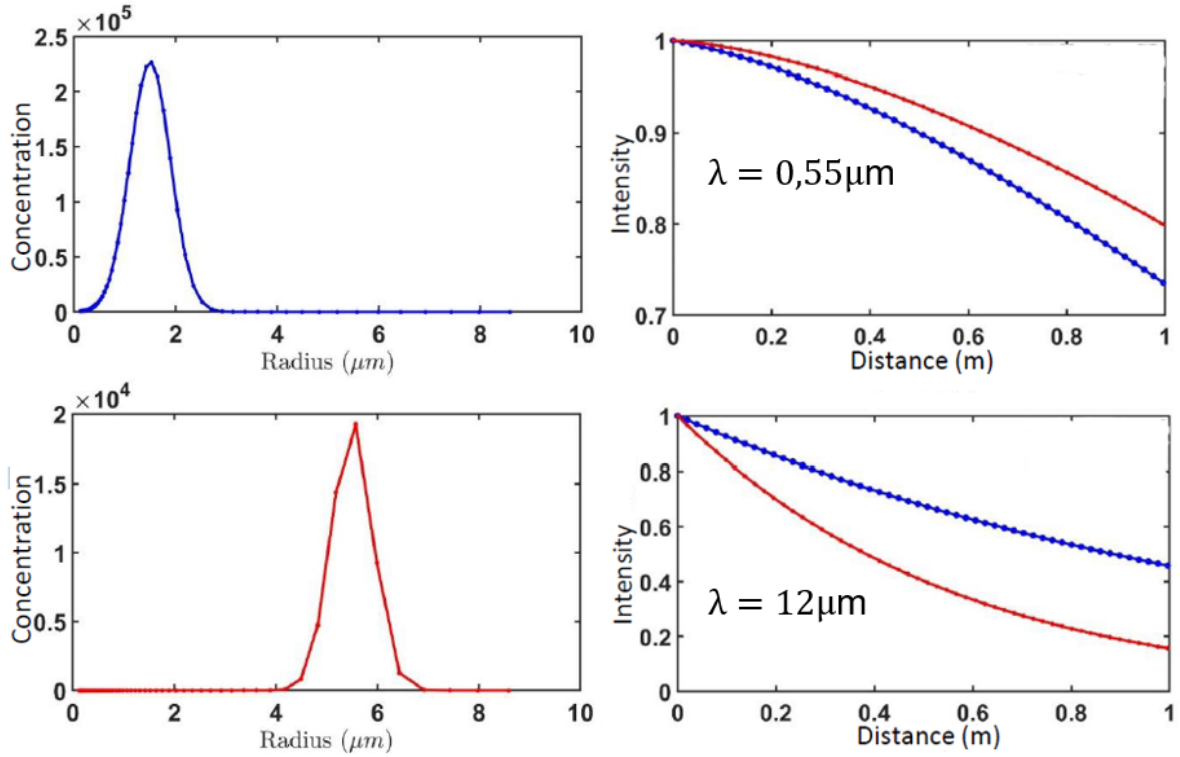

Figure 90: Simulated intensity w.r.t. the distance of a lambertian source for a fog with normalized visibility 0,75 m with small droplets (blue PSD on the left) and bigger droplets (red PSD on the left) at wavelength 0,55μm (top right) and 12μm (bottom right).

It is therefore necessary to take these effects into account in the simulations, and to have controlled meteorological environments to validate the models. To this end, CEREMA's PAVIN BP platform (see Figures 91 and 92) can be used to produce artificial fog and rain whose the characteristics can be measured and controlled (fog droplet size, meteorological visibility, rainfall rate) [239].

4.5.3 Simulation of adverse weather conditions

To facilitate the process of testing and validating the sensors in adverse weather conditions, the most effective method is to produce large realistic synthetic datasets with a wide range of adverse weather conditions and different levels of intensity. Testing the sensors in real conditions is expensive, hard to plan as the occurrence of such weather conditions is no easy to predict precisely and is not reproducible. It is not the most effective way for this task. There are two other ways to simulate adverse weather conditions, either using indoor facilities capable of producing realistic artificial rain, fog, or snow, but the task is long, tedious and can be dangerous

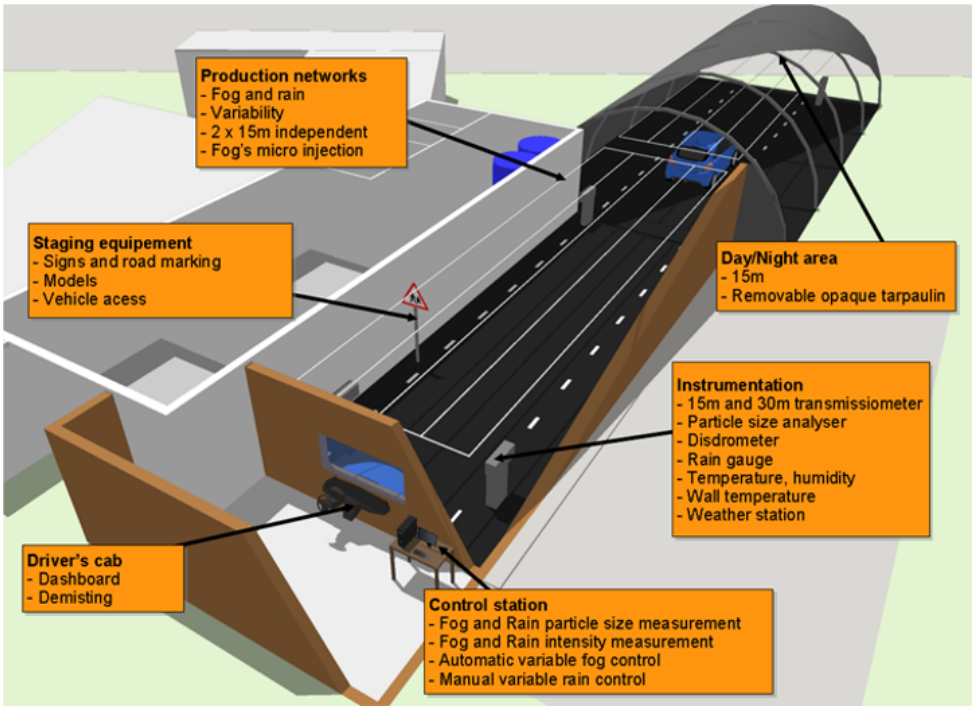

Figure 91: The Cerema PAVIN BP platform (source: Cerema)

Figure 92: Fog producing in the Cerema PAVIN BP platform (source: Cerema)

(i.e. pedestrian detection) and too complex (i.e. dynamic scenes); or by developing numerical simulation models which offer the possibility to produce an infinite number of different weather scenarios. In both cases, the methods allow reproducible and controlled simulation conditions.

The main objective is to train ADAS algorithms with large datasets of simulated adverse weathers before considering doing trials in real road conditions. Virtual adverse weather conditions data remain the most effective way to get reproducibility, low cost and diversity in the scenarios. Moreover, Johnson-Roberson et al. (2017) [240] explained that "by training machine learning algorithms on a rich virtual world, [...] real objects in real scenes can be learned and classified using synthetic data". Some numerical simulation methods are time consuming, such as Monte Carlo method, but the focus of this review of simulation methods is on noise models dedicated to reproducing the effects of adverse weather for sensors such as camera, LiDAR and RADAR. These noise models are easier to use and faster.

4.5.4 Adverse weather noise models

Simulating adverse weather conditions consist in reproducing the effect of hydrometeors and particles present along the line of sight in the medium between the sensor and a target, using physics or statistical principles. This section summarizes several examples of adverse weather conditions simulation for camera images, LiDAR and RADAR signals.

∗ **Dedicated to camera images :**

The effect of adverse weather conditions on camera images has been the subject of several studies in the literature in recent years. These studies concern the addition or removal of degraded weather effects in photo processing softwares (Photoshop), as well as for video games scenarios (Grand Theft Auto V), or even cities webcams. Hence the methods to remove or to add the effects of fog [241], rain ([242], [243]) or snow on images are well documented and could be applied to images from autonomous vehicles cameras. Nevertheless, some methods are time consuming hence not adapted to ADAS. In addition, studies in night conditions are quite rare and yet important for the problem of adverse weather conditions for autonomous vehicles.

In He et al. [241] a method based on the statistics of outdoor haze-free images is used for the dehazing of a single image using a dark channel prior. It was found that the darkest pixels of those images (excluding pixels from the sky) have a very low intensity, close to zero, in at least one of the RGB colour channel. By taking the minimum intensity values of each RGB channel of the pixels of the image, it is possible to create a kind of filter called dark channel prior. In the presence of haze, the intensity of the colour channel increase due to contribution of airlight (i.e. atmospheric hydrometeors and particles), which gives an estimation of the airlight transmission. This method has been a reference for several defogging/dehazing studies [244].

In order to train defogging algorithms or for semantic foggy scene better understanding, it is sometimes necessary to create a dataset made of foggy images aligned with corresponding none foggy images ([245], [246]).

One of the easiest way to simulate various adverse weather conditions is to start from clear weather data and to add the effects of fog, rain, snow or any other adverse weather conditions on those data.

- **Simulating fog**

  The presence of fog or mist induce a loss of contrast in the images of visible cameras. The most popular method to simulate fog is to use the visibility attenuation theory of

Koschmieder defined a century ago [235]. This theory makes it possible to determine the luminance of a black object on a sky background by an attenuation of the visibility due to the extinction of the medium between the object and the observer. This attenuation of visibility is also considered as that of an atmospheric veil or airlight.

The transmittance of a pixel at position (x, y) in the scene is a relation between the distance $d_{x,y}$ from a target to the observer and the extinction coefficient β_{ext} of the medium (in m^{-1}):

$$t_{x,y} = exp(-\beta_{ext} d_{x,y}) \tag{36}$$

Based on the attenuation law of Beer Lambert, the object luminance $L_{x,y}$ of a pixel (x, y) at a distance $d_{x,y}$ with an intrinsic luminance $L_{0;x,y}$ and L_s, the luminance of the airlight, is given by the relation :

$$L_{x,y} = L_{0;x,y} exp(-\beta_{ext} \cdot d_{x,y}) + L_s(1 - exp(-\beta_{ext} \cdot d_{x,y})) \tag{37}$$

According to Koschmieder, the visibility V (in m) is related to the extinction coefficient β_{ext}, if we consider that the minimum contrast identifiable by an observer is 0.02 (i.e. 2%). That value has been adjusted later on to 0.05 (i.e. 5%) [247].

$$V = \frac{-ln(0.05)}{\beta_{ext}} \tag{38}$$

The depth $d_{x,y}$ from the observer to the target is used to get the right estimation of the transmission map which makes it an important parameter for an accurate simulation of adverse weather on camera images. The extinction coefficient β_{ext} is considered as spatially constant when the atmosphere is homogeneous.

To be valid, this theory respects some criteria :

- Horizontal vision, otherwise the extinction coefficient cannot be considered as constant along the sight path.
- Daytime illumination conditions, without any other light source.
- The ground is flat, the atmosphere is static, uniform and none absorbing.
- The scattering of particles is independent of each other.
- The target is small compared to its distance from the camera.

Most studies consider that the camera response as linear, hence the luminance measured intensity $L_{x,y}$ can be replaced by the measured intensity of the camera. Zhang et al. (2017) modified the Koschmieder law by taking into account the angles of the objects to the horizon to adjust the depth value of the elements of the images [245].

Zhang et al. (2021) [217] made a review of some fog simulation studies using Koschmieder theory and pointed out that some of the fundamental hypotheses are neglected in the used datasets. The main negligences relate to the depth map estimation and the images of the datasets without bright sky introducing an error on the background luminance value. Both negligence have an impact on the estimation of visibility with too much or too little

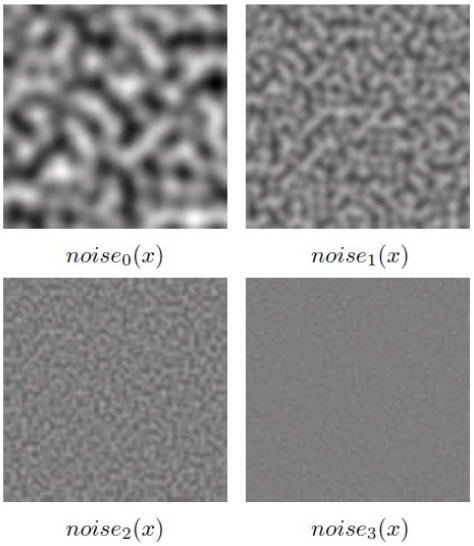

Figure 93: Perlin noise examples with different amplitude and frequency [26].

attenuation of the contrast of the image, giving incorrect fog representation and therefore an unsuitable response from the sensor.

Guo et al. (2014) [248] used a Markov random field to estimate the transmission map. Sakaridis et al. (2018) [246] added synthetic fog to real images with clear weather from Cityscapes dataset. To overcome the problem of estimating the horizon sky luminance in the images, they used the median values of the 0.1% of pixels with the largest dark channel values (i.e. intensity values close to zero in the RGB channels).

For images taken in real world conditions, it is possible to get the depth information using for example stereo imaging or LiDAR scan. The virtual vehicle environment simulators such as CARLA (CAR Learning to Act), GTAV (Grand Theft Auto V) can also give that information. SYNTHIA, Sim4CV, Cityscapes and KITTI datasets also provide the depth maps of the images with more or less accuracy depending on the use of depth sensors or not.

Several method make the assumption that the fog is spatially homogeneous making the simulation easier. Nevertheless, natural fog is heterogeneous and therefore more complex. It is constituted of many layers of non-homogeneous fog exposed to air and wind turbulence. To simulate heterogeneous fog the most popular method in the literature consist in applying a Perlin noise to the synthetic fog images ([26], [245]). Perlin noise gives a heterogeneous fog density distribution texture that depends on noise amplitude and frequency. By associating different Perlin noise (See [26], [245]) following a turbulence equation (39) , with the extinction coefficient β_{ext} of the fog, it is possible to obtain a heterogeneous fog density distribution. Figure 93 illustrates different examples of Perlin noises [**?**] with various frequencies and amplitudes giving different textures (i.e. level of heterogeneous fog density).

$$turbulence(x) = \sum_{i=0}^{N-1} \frac{noise_i(x)}{2^i} \tag{39}$$

Giroud and Biri (2010) [249] modeled fog using a B-Spline function basis to construct the extinction function and then used wavelets decomposition to generate different resolutions. Their method makes is possible to produce an animated fog which is not possible when using Perlin noise.

Simulating fog for camera images will be different for day-time or night-time scenarios. For night-time fog conditions simulation, the light sources illuminating the scene are the vehicle headlights, urban lightings and other secondary sources producing glow effects around sources and shadows. It is then necessary to take into account the contribution of each light source to each element of surface. The presence of a halo around a light source and its size depend on the granulometry of the fog. Gallen et al. (2010) [250] propose a method that has been tested and validated through Monte Carlo-based simulation with PROF (Photometrical Rendering Of Fog). Using several light sources with different distances, they estimated the forward scattering due to fog droplets for nighttime conditions, for advective or radiative fog. Their interest was on the characterization of heavy to dense fog at night. The type of fog is classified based on its forward scattering properties knowing the visibility and the relative error of the source intensity. The relative error is computed as a tabulated function depending on the transmittance and the granulometry of the fog. When the forward scattering is not well estimated, the estimation of the atmospheric extinction is biased and the visibility can be overestimated, especially for fog with big droplets.

Sun et al. (2005) [251] use the OpenGL fog model on indoor simulated images to blend the fog color with the object color, based on the distance from the observer to the object. Their method takes into account the effects like glows around light sources, typical from secondary light sources in the presence of fog and are able to reproduce the volumetric effect of fog.

- **Simulating rain**

  Simulating rain for camera measurements means being able to reproduce the rain steaks characteristic of the camera exposure time and the speed of falling raindrops. It is related to the drop size and the type and intensity of the rain, i.e. drizzle, widespread rain, and thunderstorm. Rain can also affect the camera sensor by covering the camera lens. Rain drops, as well as fog droplets, are often considered as spherical which makes them easier to simulate by only knowing their radius.

  Hasirlioglu (2020) [223] simulated rain fall with the ray tracing technique. Using this technique different intersection matrices, regrouping parameters such as drop diameter, closest hit or drop identifier, are created at a time step corresponding to camera exposure time. When a drop identifier is present in several successive intersection matrices, a rain streak will be represented.

  Then, the colour of the virtual rain $C_{vir,(i,j)}$ is simulated by merging the background colour $C_{bkgd,(i,j)}$ with a partially opaque bright colour, such as white or off-white, corresponding to the rain colour $C_{rain,(i,j)}$ following the equation :

$$C_{vir,(i,j)} = \alpha_{cam,(i,j)} \cdot C_{rain} + (1 - \alpha_{cam,(i,j)}) \cdot C_{bkgd,(i,j)} \tag{40}$$

With $\alpha_{cam,(i,j)} = (1 - \mu_B) \cdot C_{cam,(i,j)}$, the opacity factor, a relation between the mean brightness of the reference image μ_B and the scaling factor from the camera noise filter $C_{cam,(i,j)}$ created with the intersection matrices. It is also possible to consider the rain colour as the mean colour of the ground truth image.

- **Simulating snow**

  To simulate snow on camera images, it is necessary to know the position and size of snow flakes. The method is similar to the one dedicated to rain. Elements such as wind or vehicle speed produce a motion blur effect that must be reproduced. Von Bernuth et al. 2019 [252], use an equation with a relation between snow fall rate and snow mass concentration, for dense snow fall or regular snow fall, based on Koh and Lacombe [253]. They use an exponential relation to defined the snow flake size distribution in relation with the snow precipitation. Flakes are considered as flat crystals and defined as quads for light snow. For dense snow, flakes are considered as thick aggregated and are represented as three pairwise perpendicular quads. The blur effect is created by dividing each frame into inter-frames based, on the exposure time of the camera and snow motion vector.

∗ **Dedicated to LiDAR sensors :**

- **Simulating fog**

  Hahner et al. (2021) [254] developed a fog simulation method applicable to any real clear weather LiDAR point clouds dataset with a bistatic beam configuration. From the LiDAR equation [255], the fog is simulated by "modifying the part of the impulse response that pertains to the optical channel (i.e. the atmosphere)" [254]. The received power$P_R(R)$, which depends on the range R, is expressed as a time-wise convolution between the transmitted signal power $P_T(t)$ and the impulse response H of the optical system, with C_A a system constant independent of time and range, c the speed of light.

$$P_R(R) = C_A \int_0^{\frac{2R}{c}} P_T(t) H(R - \frac{ct}{2}) dt \tag{41}$$

  With the transmitted pulse P_T depending on P_0, the pulse's peak power and τ_H, the half-power pulse width, and time t:

$$\begin{cases} P_T = P_0 \sin^2(\frac{\pi}{2\tau_H} t) & \text{if } 0 \leq t \leq 2\tau_H \\ 0 & \text{otherwise} \end{cases} \tag{42}$$

  The algorithm of Hahner et al. (2021) [254] only needs four inputs parameters and a clear weather point cloud with known intensity, which makes it easy to use.

An object is considered as a hard target whose impulse response is a Dirac delta function, while fog is considered as a soft target whose impulse response is a Heaviside function. The received power in fog conditions is decomposed as the addition of the received power from the hard target and the received power from the soft target.

Hasirlioglu (2020) [223] uses the relative sensor power LiDAR equation :

$$P_R = \frac{1}{R^2}\beta_{back} \cdot exp(-2\int_0^R \gamma_{ext} dr) \tag{43}$$

He explicites the expression of the backscattering coefficient of fog droplets based on the DSD of the fog and the reflected volume of the laser beam:

$$\beta_{back} = \frac{\pi}{4}\int_0^\infty N(D) Q_b D^2 \tag{44}$$

where $N(D)$ is the number of drops of diameter D, and Q_b the backscatter efficiency. Under homogeneous assumption, the backscattering coefficient is constant. The extinction coefficient γ_{ext} is calculated using the relation with visibility already mentioned in the previous paragraph.

- **Simulating rain**

Goodin et al. (2019) [256] developed a mathematical model to evaluate the effect of rain on the performance of LiDAR. The model is based on the LiDAR equation for a target at a distance z from the sensor :

$$P_r(z) = E_l \frac{c\rho(z)A_r}{2R^2}\tau_T\tau_R exp(-2\int_0^d \alpha(z\,')dz\,') \tag{45}$$

With P_r the power received by the LiDAR sensor in W, E_l the laser pulse energy in J, c the speed of light in $m \cdot s^{-1}$, $\rho(z)$ the backscattering coefficient of the target, $\alpha_{z\,'}$ the scattering coefficient of the rain along the path of the target, A_r the effective receiver area in m^2 and, τ_T and τ_R the transmitter and receiver efficiencies.
The extinction coefficient α can be estimated based on a power law depending of the rainfall rate R [257] :

$$\alpha = aR^b \tag{46}$$

With a and b, constant values which depend on the DSD and the droplet velocity. Then, after simplifications Goodin et al. (2019) [256] present the relative sensor power returned by LiDAR expressed as a function of rainfall rate R, distance z and the backscattering coefficient ρ :

$$P_n(z) = \frac{\rho}{z^2} exp(-2aR^b z) \tag{47}$$

Chan et al. (2020) [229] present the results of a LiDAR model combining two noise sources perturbating the LiDAR sensor applied on a LiDAR scan from a MATLAB

dataset of a vehicle driving along a road. The first noise model is the one developed by Goodin et al. (2019) [256] and the second noise model reproduces the effect of partial occlusion of the lens blocking the emission of some LiDAR laser beams. To take into account the noise introduced by the rain to the range measurement, the modified range z' can be determined as follow, for a maximum variance of 2%:

$$z' = z + N(0, 2az(1 - e^{-R})^2 \tag{48}$$

The conclusion of their study highlighted that for both noise sources modelization, the shapes of the objects around the AV in the point cloud are really perturbated by rain drops. The article by Byeon et al. 2020 [258] focuses on modeling the effect of rain on LiDAR performances taking into account the scattering effect of rain particles and the region-dependent distribution of raindrops using precipitation data from three different locations in the world.

- **Simulating snow**
  Hahner at al. (2022) [259] present a study on the simulation of the effects of snow fall on LiDAR measurements to train 3D object detection models. As for camera images, simulating snow fall is similar to simulation rainfall. The snow falls are sampled in the 2D space and the effect of the geometry of the LiDAR beam is applied. They also take into account the wetness of the road produced by the snow fall on the LiDAR point clouds. To do so they use geometric optics and represent the wetness of the road as a thin layer of water. In their study snow flakes are modeled as spherical particles and optical geometry is used. The received power in snowfall is a superposition of multiple echoes associated to snow particles or any other object in the laser beam.

∗ **Dedicated to RADAR sensors :**

RADAR sensors are mainly impacted in case of heavy rain fall and, as already mentioned in the previous section, the effects of rain on RADAR are twofold: attenuation and backscatter. Zang et al. (2019) [222] and Hasirlioglu (2020) [223] propose mathematical models to represent those effects of rain on RADAR signal :

- The attenuation effect, based on the RADAR equation :

$$P_r = \frac{P_t G^2 \lambda^2 \sigma_T}{(4\pi)^3 r^4} V^4 exp(-0.2\gamma r) \tag{49}$$

- The backscatter effect :

$$(\frac{S_t}{S_B}) = \frac{8\sigma_t}{\tau c \theta_{BW}^2 \pi R_t^2 \sigma_i} \tag{50}$$

Using the parameters and variables presented in table 1.

Variables	Names
P_r	Signal Power
P_t	Tranmission Power
G	Antenna gain
f	Radar frequency
T	Pulse duration
θ_{BW}	Antenna Beamwidth
P_r	Signal Power
σ_t	Radar Cross section of target
F_N	Receiver Noise Figure
B	Receiver Filter Bandwidth
T_0	Thermal Temperature
S_t	Power intensity of target signal
S_b	Power intensity of backscatter signal
c	Speed of light
V	Multipath coefficient
γ	Rain attenuation coefficient
σ_i	Rain backscatter coefficient
λ	Radar Wavelength
r	Distance between radar and target

Table 1: Parameters and variables of radar equation

Snow and mist can also affect RADAR signal with attenuation and backscattering, it can be modeled following the same equation than rain but with attenuation and backscatter coefficients corresponding to rain [222].

4.6 Real and virtual augmented and enhanced Data

Different methods and approches have been developed recently in order to mix real and virtual data/scene or to improve the simulation rendering from information and models coming from real data. It is possible to classify these methods in 3 categories:

- **Enhanced Reality**: Like Intel method, this category of processing consists to used models built from dataset with real data in order to take into account specific modifiers and features in real images. From this model, the virtual image could be modified and improved to give photo-realistic synthetic image.
- **Augmented reality**: This category consists to add artefacts and objects to a real or virtual data stream (like Baidu and NVIDIA methods), or to add raw simulated data to the data stream of a real sensor (like INRIA's approach)
- **Enriched Realities**: Consist to add artefacts (real/virtual) to populate a scene (like Baidu and NVIDIA methods)

4.6.1 INTEL: Post processing based AI from photo realistic rendering

Very recently, Richter and al [260] from INTEL present an approach to enhancing the realism of synthetic images. The images are enhanced by a convolutional network that leverages intermediate representations produced by conventional rendering pipelines. The network is trained via a novel adversarial objective, which provides strong supervision at multiple perceptual levels. They analyze scene layout distributions in commonly used data-sets and find that they differ in important ways. They hypothesize that this is one of the causes of strong artifacts that can be observed in the results of many prior methods. To address this they propose a new strategy for sampling image patches during training. They also introduce multiple architectural improvements in the deep network modules used for photo-realism enhancement. The results obtained report substantial gains in stability and realism in comparison to recent image-to-image translation methods and a variety of other baselines. In the example given in [260] and presented in the figure 94, an image generated by a graphical engine (i.e. image from modern computer game (GTA V) is modified by applying the convolutional network. the convolutional network modifies the initial image and enhance it to mimic the style of a data-set with real data (Cityscapes in this example). Nevertheless this enhancements are semantically consistent and non-trivial. For instance, the method adds gloss to cars (1st row of the figure), reforests parched hills to mimic German climate (2nd row of the figure), and paves roads with smoother asphalt (3rd row of the figure). Insets magnify marked regions. The question will be how is realistic the improved images, not from the human perception point of view but more from a camera point of view.

Figure 94: Convolutional networks to enhance the photo-realism of rendered images. Left: frames from a modern computer game (GTA V). Right: same frames enhanced by the INTEL approach to mimic the style of Cityscapes

4.6.2 NVIDIA Omniverse, a way to generate Digital Twin and augmented reality

Recently, NVIDIA has develop an efficient and powerful series of library and simulation softwares allowing to generate from their equipped vehicle an accurate digital twin with a dense mesh. (https://www.youtube.com/watch?v=PWcNlRI00jo&t=2500s)

Figure 95: NVIDIA: Generation of Digital Twin from embedded sensors - Left: real scene, Right: Digital Twin

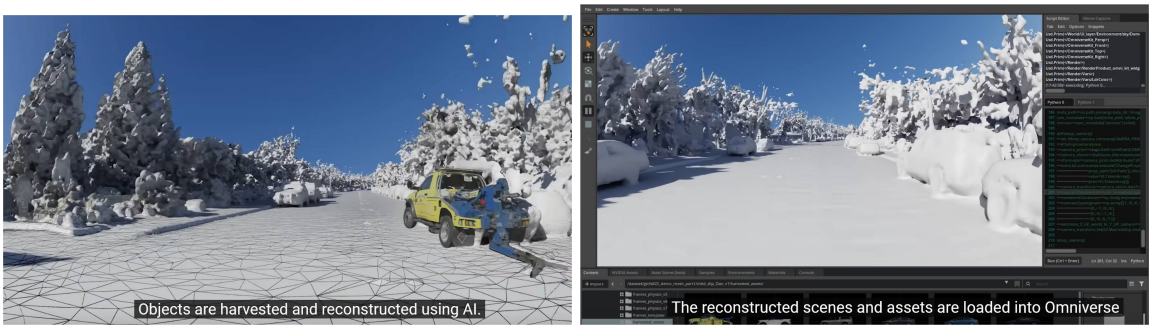

Figure 96: NVIDIA: Generation of Digital Twin from embedded sensors with mesh generation and objects identification and reconstructing using an AI engine

Figure 97: NVIDIA: Add dynamic objects using NVIDIA DRIVE Map. The tow vehicle has been reconstructed and moved. Virtual ball, child, and vehicle have been added to the initial scene

In this digital twin, NVIDIA can identify the vehicles and replace them by virtual ones by using a Neural Reconstruction Engine in DRIVE Sim in order to build specific scenario (see figure 98). Moreover, by using the digital twin in the NVIDIA DRIVE Sim, it is possible to put new virtual dynamic objects in the simulation environment and use these virtual data in order to enhance the real data and then provide the replay of a dataset with real data and apply an augmented reality (see 99).

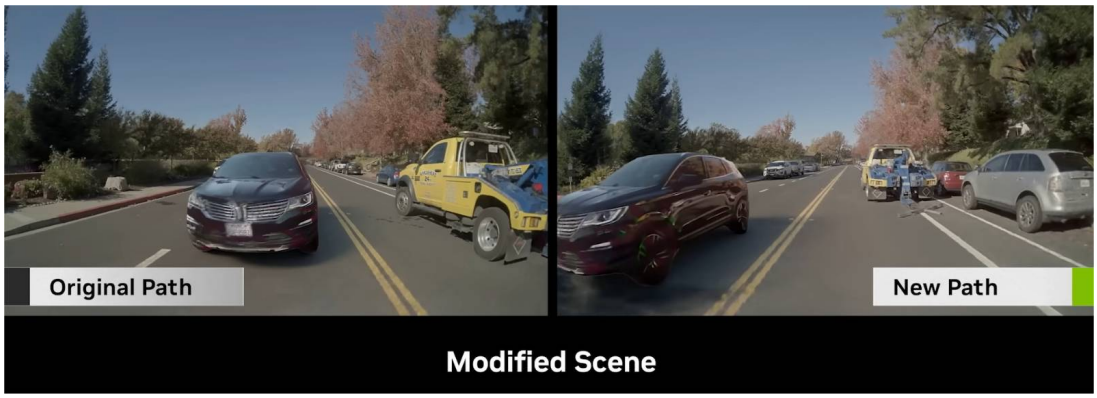

Figure 98: NVIDIA: Omniverse, DRIVE Sim, and digital twin for augmented reality. Left: original path with virtual obstacle, Right: new path modifying the ego-vehicle moving

Figure 99: NVIDIA: Omniverse, DRIVE Sim, and digital twin for augmented reality (virtual moving pedestrians, bus, and cars) in a real scenario

This method of augmented reality is efficient but with some strong constraints. The augmented scene and scenario are constraint by the initial scene in term of illumination, contrast, and existing shadows. Nevertheless, it could be possible to add some weather conditions as fog, rain, and falling snow but with the modification of the sky and the ambient light value.

4.6.3 BAIDU: AADS, Augmented autonomous driving simulation in a real scene

In 2019, Baidu presented an interesting way to generate augmented reality (see figure 100 for the functional architecture). This work has been made in order to develop and to validate autonomous driving (AD) technologies. They started from the existing state-of-the-art approaches for simulation using game engines or high-fidelity computer graphics (CG) models to create driving scenarios. They highlighted that creating CG models and vehicle movements remain mainly a manual tasks that can be costly and time consuming. In 2019, Baidu team thought CG images still lacked the richness and authenticity of real-world images, and using CG images for training leads to degraded performance. From 2019, a lot of improvements have been done in this domain. Nevertheless, in their works, Baidu presents an interesting way for augmented autonomous driving simulation (AADS). Their goal is to generate augmented real-world pictures with simulated traffic flows to create photo-realistic simulation images and renderings. More specifically, they used LiDAR and cameras to scan street scenes. From the acquired trajectory data, they generated plausible traffic flows for cars and pedestrians and composed them into the background. The composite images could be resynthesized with different viewpoints and sensor models (camera or LiDAR). The resulting images seem to be photo-realistic (see figure 101), fully annotated, and ready for training and testing of AD systems from perception to planning. Compared with the traditional approaches in 2019, their method offers scalability for complexity and diversity management and realism with the capability to combine the virtual environment with the richness of the real world.

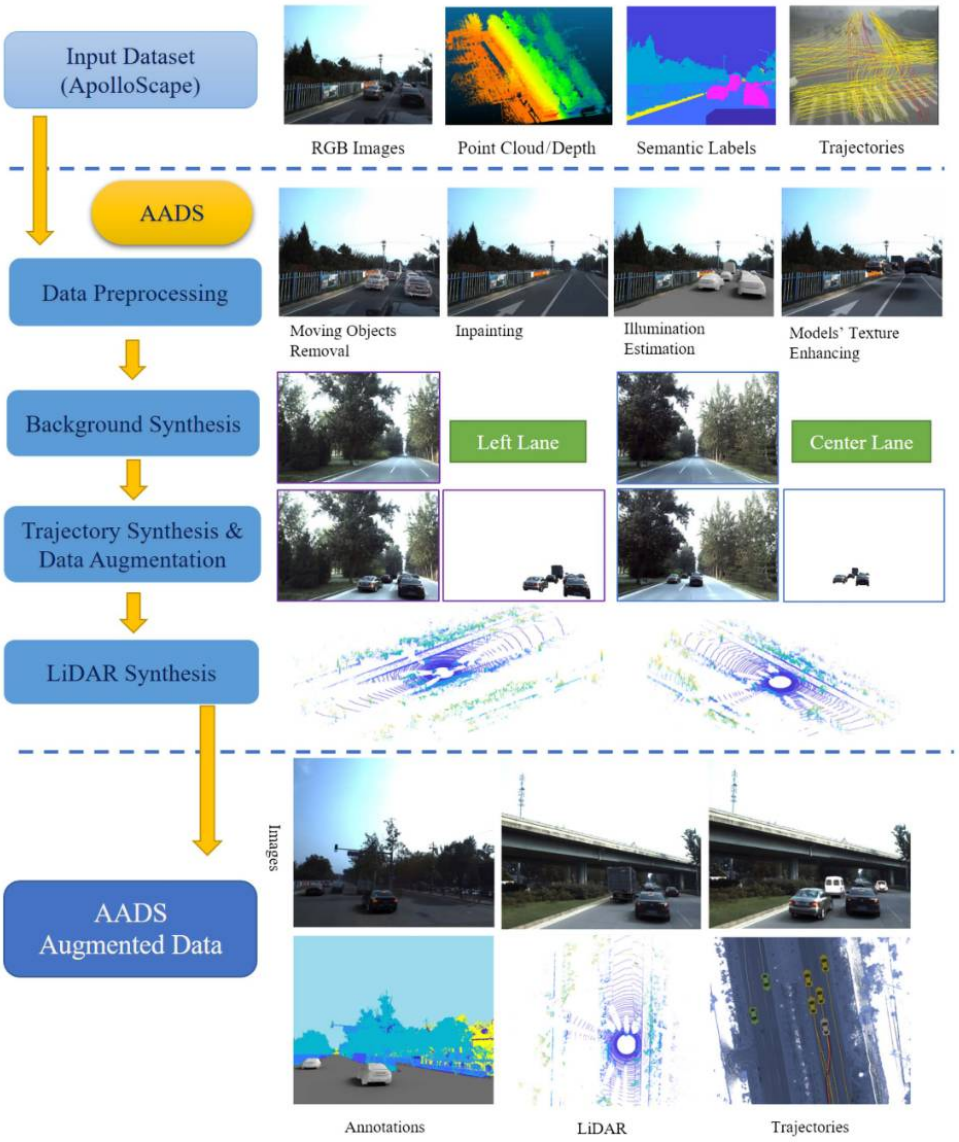

Figure 100: The inputs, processing pipeline, and outputs of our AADS system. Top: The input dataset. Middle: The pipeline of AADS is shown between the dashed lines and contains data preprocessing, novel background synthesis, trajectory synthesis, moving objects' augmentation, and LiDAR simulation. Bottom: The outputs from the AADS system, which include synthesized RGB images, a LiDAR point cloud, and trajectories with ground truth annotations ([27])

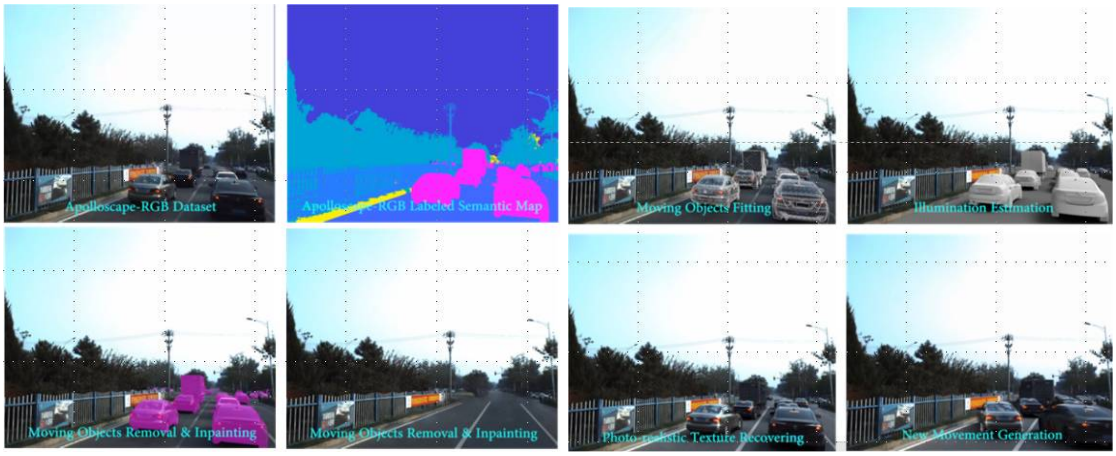

Figure 101: AADS, the capability of add virtual objects in real data in order to generate a large set of scenarios for evaluation and validation of AV

4.6.4 Real-time augmentation of LiDAR sensor data

Even though simulation environments can play a fundamental role in testing and validating advanced automotive algorithms to guarantee their safety and quality, they still present some limitations cannot be the unique answer. In order to overcome these limitations, INRIA recently proposed a new augmented reality framework that is intended to be a bridge between Vehicle-in-the-Loop and real-world testing [28].

In the last years, several works made a significant improvement to go beyond ViL and combine a virtual and real test environment. As these methods augment real test scenes with virtual elements, we call them augmented reality. AR offers safe and efficient testing with virtual elements but also rich, dense and realistic environments. Several approaches introduce AR at object level. They represent the virtual elements by their position, speed, status and main characteristics. There is no realistic sensor emulation. Object level AR does not require heavy computation and allows light implementations. The simple representation of AR elements enables to share them on communications between vehicles and with infrastructure. Thanks to this, object level AR can be synchronised with several vehicles under test. It can also be centralised and computed in the infrastructure. In [261], Chen et al. present a unified fusion data format to represent the AR scene at object level. Also, an implementation of object level AR has been realised at the Mcity facility [262], it relies on the V2X communication protocol to introduce and share virtual objects. Another implementation has been developed for the ZalaZONE proving ground, it introduced the concept of Scenario-in-the-Loop [263]. SciL offers a mixed reality framework where virtual objects are introduced on a communication protocol based on 5G. Also some sensors are emulated on the vehicle under test with a simple range sensor model. However perception is a critical part of advanced automotive software and it can not be challenged by object level AR. A realistic sensor emulation is needed. AR has to be introduced directly in the sensor detection, in the messages as they come out of the sensor drivers, and only then perception can be tested in AR. Moreover scenes with partial perception, i.e. when something occludes part of the scene, cannot be recreated with object level AR but can be accurately generated with sensor level AR. Also, object level AR cannot integrate virtual background elements such as for example infrastructure, trees or walls because they cannot be represented by objects in the main communication protocols. In the meanwhile, any foreground or background element can be real or virtual with sensor level AR. But sensor level AR requires rather heavy

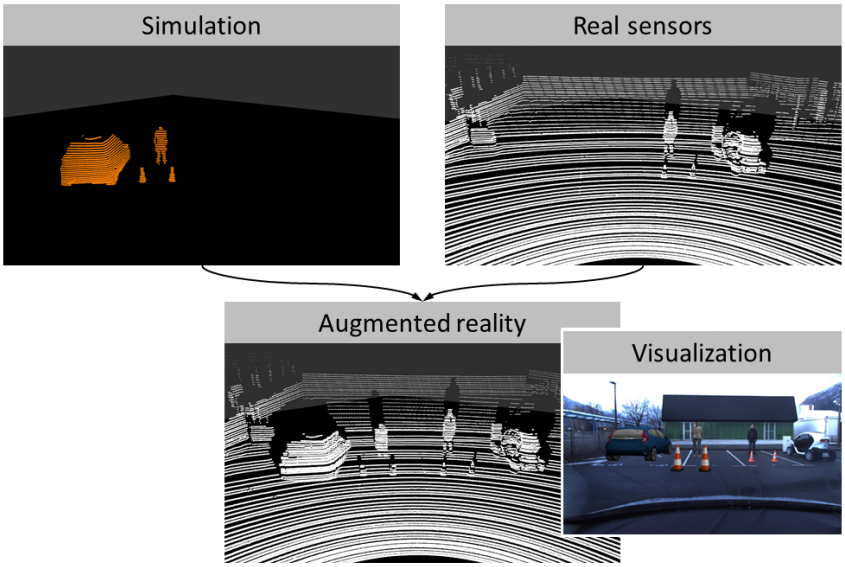

Figure 102: Example of LiDAR point cloud augmentation. The introduced virtual point cloud and initial sensor point cloud are shown on the top. The technique, deployed on a vehicle, generate the fused point cloud and the visualization of the augmented scene on the bottom.

computations to be executed in real time, this is a hard constraint on implementation.

The key aspect of the method proposed by INRIA is the design of a merge function allowing a real-time augmentation of LiDAR data with virtual elements (see Fig. 102 and Fig. 103 for illustrative examples). This solution may open new possibilities for testing: a testing site can very easily be populated with many and diverse virtual elements in order to create more complex test scenarios. Virtual pedestrians or cars are easier to operate and offer richer and more active behaviours (e.g. reacting to the ego-vehicle's motion). Furthermore, all elements of the test scenario that may induce a collision risk can be replaced by their virtual counterpart to secure the tests in the early stages of development or to test the system in critical situations. Virtual scenarios are also repeatable and this is a key feature to reproduce experiments. The AR[8] testing implementation accurately represents the virtual scenes and guarantees a consistent fusion with the real world. So AR tests produce meaningful results that can be used to infer the behaviour of the vehicle in the real world. Finally, as any element can be either real or virtual, AR testing offers a smooth transition from simulation to actual testing. For these reasons, the proposed AR framework can be a fundamental testing solution for the validation of advanced automotive software. In particular, the main novel contributions that this work presented are:

- the design of a new framework enabling AR directly on sensor data;
- the data fusion methodology that allows the real-time augmentation of LiDAR sensor data.

Even if this AR system does much more than ViL, it has the same architecture and several modules in common with most ViL systems. Reusing the ViL topology which is described in [264], the AR system consists of the four following modules:

- a virtual environment which contains a twin of the experimental vehicle;
- a synchronization module which updates the position and state of the virtual twin;

[8] AR: augmented reality

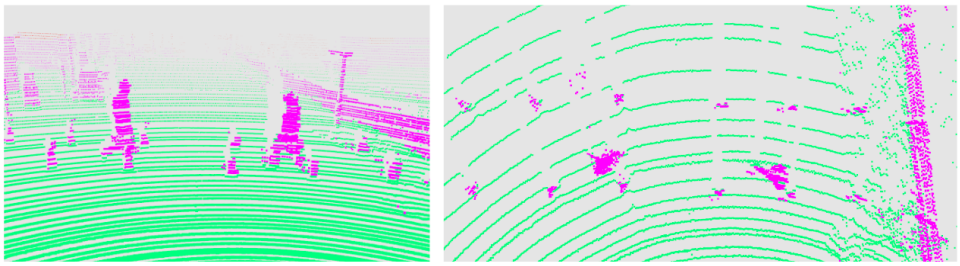

Figure 103: Example of a point cloud obtained with Augmented Reality, seen from the vehicle prospective and from above. The scene consists of an actual pedestrian and 6 actual construction cones on the left plus a virtual pedestrian and the same set of cones on the right. Pedestrians are approximately 5m away from the vehicle. The points are displayed in green for those of the ground and magenta for the other points.

- a sensor emulation which generates outputs from the virtual sensors and integrates them in the actual sensors' outputs;
- a visualization which helps testers to understand the AR scene.

Fig. 104 shows a schematic representation of the software framework. The periodic messages of the sensors of the real vehicle give rhythm to the virtual world. So all modules must run in real time, their execution duration must be short compared to the period of the sensors. This is a heavy constraint on the design and implementation of the final solution. For more technical details on the different modules see [28].

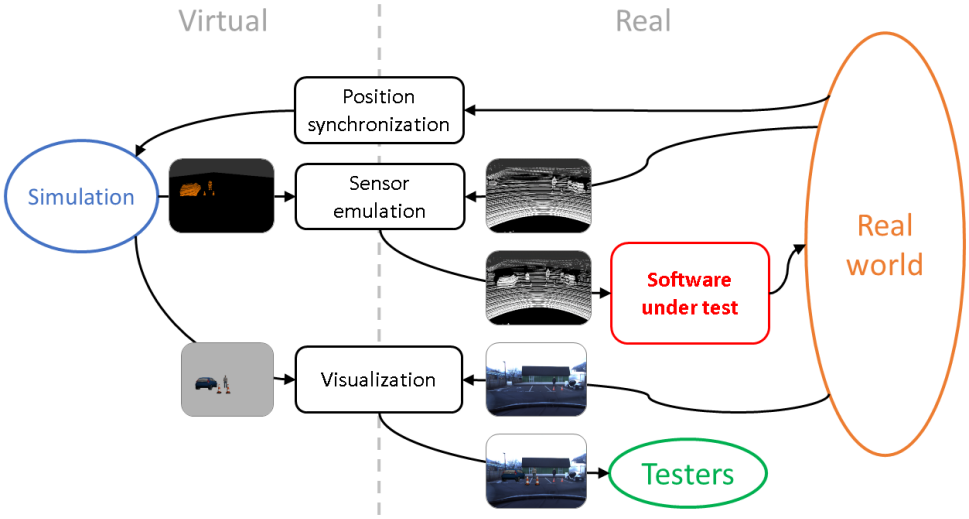

Figure 104: Structure of the Augmented Reality framework presented in [28].

This framework for real-time augmentation on LiDAR data will be used and further tested in the PRISSMA project, in connection with both WP2 and WP3. In particular, it will play a central role in the POC organised in collaboration between INRIA and Transpolis.

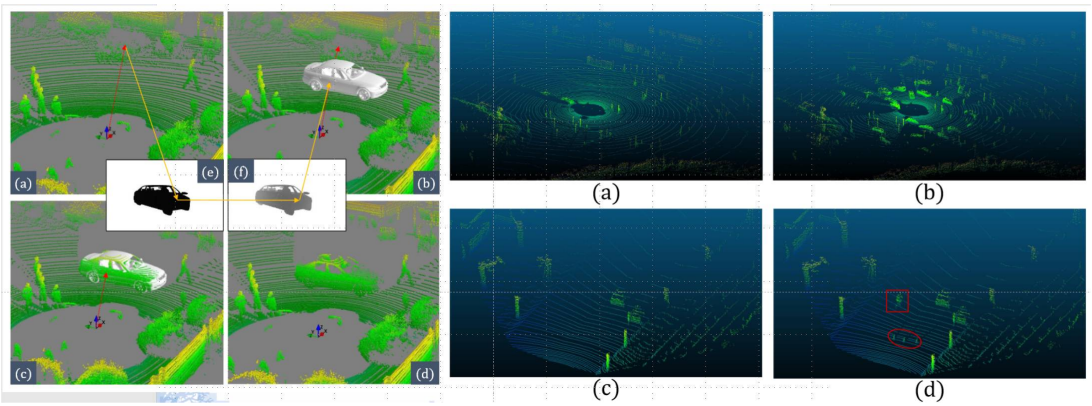

Figure 105: LiDAR-Aug: A General Rendering-based Augmentation Framework for 3D Object Detection presented in [29].

The following reference [30] because the authors propose Data Augmentation of Automotive LIDAR Point Clouds by in degraded condition with adverse weather conditions.

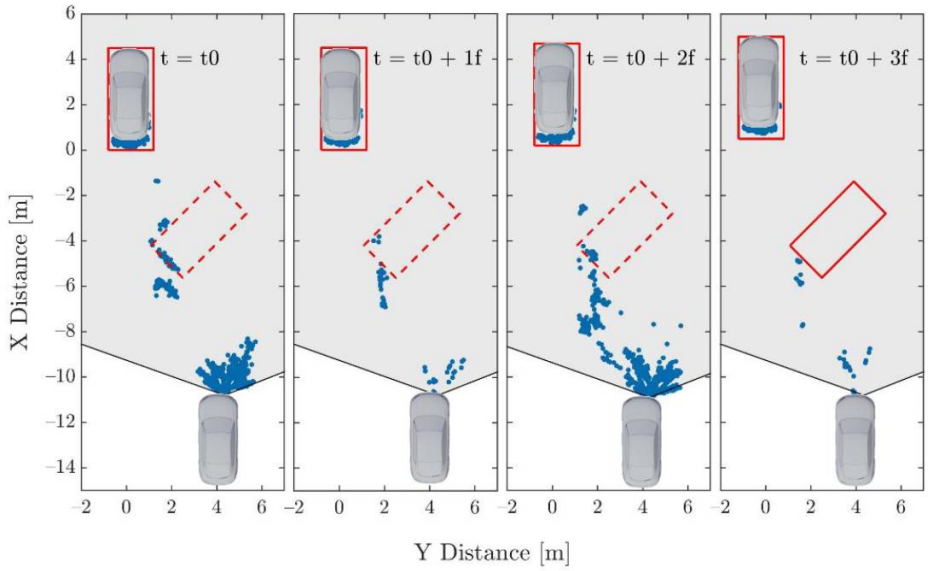

Figure 106: Data Augmentation of Automotive LIDAR Point Clouds under Adverse Weather Situations presented in [30].

4.7 Data sets and meta-material data bases

4.7.1 Data sets from real data

In order to evaluate the performances of the algorithms, some datasets are available with the associated ground truth. Among these datasets, we can quote the large scale ATG4D object detection dataset, and the small-scale KITTI object detection benchmark [8]. The ATG4D dataset contains 5,000 sequences for training and 500 for validation. The sequences from the train set are sampled at 10 Hz, while the validation set are sampled at 0.5 Hz. The entire dataset contains 1.2 million sweeps for training, and 5,969 sweeps for validation. All sweeps are captured using a Velodyne 64E LiDAR. The KITTI 3D object detection benchmark ([265]), [266]) consists of 7481 training images and 7518 test images as well as the corresponding point

clouds (captured by a Velodyne 64E LiDAR), comprising a total of 80.256 labeled objects. For the training set, object annotations are provided within the front 90° field of view of the LiDAR up to approximately 70 meters. For evaluation, KITTI computes precision-recall curves. To rank the methods we compute average precision. KITTI requires that all methods use the same parameter set for all test pairs. The development kit provided by KITTI contains details about the data format as well as MATLAB / C++ utility functions for reading and writing the label files. KITTI proposed to evaluate 3D object detection performances using the PASCAL criteria also used for 2D object detection. Far objects are thus filtered based on their bounding box height in the image plane. Only objects also appearing on the image plane are labeled. KITTI notes that the evaluation does not take care of ignoring detections that are not visible on the image plane — these detections might give rise to false positives. In order to validate car detection, KITTI evaluation stage requires a 3D bounding box overlap of 70%, while for pedestrians and cyclists we require a 3D bounding box overlap of 50%. Difficulties are defined as follows:

- **Easy**: Min. bounding box height: 40 Px, Max. occlusion level: Fully visible, Max. truncation: 15 %
- **Moderate**: Min. bounding box height: 25 Px, Max. occlusion level: Partly occluded, Max. truncation: 30 %
- **Hard**: Min. bounding box height: 25 Px, Max. occlusion level: Difficult to see, Max. truncation: 50 %

On October 2019, KITTI's team has followed the suggestions of the Mapillary team in their paper "Disentangling Monocular 3D Object Detection" [267] and use 40 recall positions instead of the 11 recall positions proposed in the original Pascal VOC benchmark. This results in a more fair comparison of the results. In [31], the authors give an overview of main driving data-set allowing the evaluation of the obstacle detection with different types of sensors.

Dataset	Image	LiDAR	2D annotation	3D annotation	ego signals	Naturalistic	POV	Multi trip	al weathers	day and night
Astyx Dataset HiRes2019										
Cityscapes	■		■				Vehicle			
Berkeley DeepDrive	■		■				Vehicle		■	■
Mapillary	■		■				Vehicle		■	■
Oxford RobotCar	■	■					Vehicle	■	■	■
KITTI	■	■	■	■			Vehicle			
H3D	■	■		■			Vehicle			
ApolloScape	■	■	■	■			Vehicle			
nuScenes	■	■	■	■			Vehicle		■	■
Udacity	■	■	■	■			Vehicle			
DDD17	■		■		■		Vehicle		■	■
Comma2k19	■				■		Vehicle			■
Landmarks										
Landmarks-v2	■									
Level 5	■	■	■	■			vehicle			
LiVi-Set	■	■			■		Vehicle			
NU-drive	■				■	semi	Vehicle	■		
Open Images V5	■	■	■							
Pandaset	■	■	■	■			Vehicle			
SHRP2	■				■	■	Vehicle			
100-car	■				■	■	Vehicle		■	■
euroFOT	■				■	■	Vehicle			
TorontoCity	■	■	■	■			Aerial, Panorama, Vehicle			
KAIST multi-spectral	■	■	■				Vehicle			■
Waymo Open Dataset							Vehicle	■	■	■

Figure 107: The main Driving Data-Sets [31]

About the road attributes detection and identification, CULane [268] dataset is widely used. This data set contains many different challenging driving scenarios, like crowed, shadow, night, dazzle light and so on. For light conditions style transfer, 3200 images in suitable light conditions and 3,200 images in low-light conditions have been selected. These images are divided according to a ratio of 3:1, which are respectively used as the training set and test set.

This data-sets could be interesting for virtual sensors and simulation platform in a process of sensor modelling. For instance, [269] used a large scale data-sets containing large amounts of sensor measurements in order to assess sensor performance and sensor modelling. This process allowed to find and extract dominant sensor effects and to configure these effects in a sensor model.

The nuScenes dataset (Caesar et al., 2019 [270]) is the first publicly available large-scale data-set that provides data from the entire sensor suite of an autonomous vehicle, i.e., camera, LiDAR, and radar. The full nuScenes data-set was released in March 2019 and consists of 1000 driving scenes in Boston and Singapore. Each scene is 20 seconds long and manually selected to comprise diverse driving manoeuvres, traffic situations, and unexpected behaviours. The data-set includes 1.4 M camera images, 390k LiDAR sweeps, 1.4 M radar sweeps, and 1.4 M object bounding boxes in 40k key-frames.

In [33], the authors provide an interesting survey of Scene Driven Autonomous Data-sets and Tool-sets. They start to propose a clustering of the Knowledge Map based Test Influencing Factors Analysis (see figure xx). For each one of this cluster (key concepts and influencing factors) they provide remarks, references, and relative data-sets (see figure 111).

Most deep multi-modal perception methods are based on supervised learning. Therefore, multi-modal datasets with labelled ground-truth are required for training such deep neural networks. In In [32], authors summarise several real-world datasets published since 2013, regarding sensor setups, recording conditions, dataset size and labels (see figure 108 and figure 109). The mentioned datasets are mainly KAIST Multispectral, KITTI, Apolloscape and nuScene dataset. The Eurocity dataset [88] focuses on vulnerable road-users (mostly pedestrian). The BLV3D dataset provides unique labelling for interaction and intention.

Name	Sensing Modalities	Year (published)	Labelled (benchmark)	Recording area	Size	Categories / Remarks	Link
Astyx HiRes2019 [94]	Radar, Visual camera, 3D LiDAR	2019	3D bounding boxes	n.a.	500 frames (5000 annotated objects)	Car, Bus, Cyclist, Motorcyclist, Person, Trailer, Truck	https://www.astyx.com/development/astyx-hires2019-dataset.html
A2D2 [87]	Visual cameras (6); 3D LiDAR (5); Bus data	2019	2D/3D bounding boxes, 2D/3D instance segmentation	Gaimersheim, Ingolstadt, Munich	40k frames (semantics), 12k frames (3D objects), 390k frames unlabeled	Car, Bicycle, Pedestrian, Truck, Small vehicles, Traffic signal, Utility vehicle, Sidebars, Speed bumper, Curbstone, Solid line, Irrelevant signs, Road blocks, Tractor, Non-drivable street, Zebra crossing, Obstacles / trash, Poles, RD restricted area, Animals, Grid structure, Signal corpus, Drivable cobblestone, Electronic traffic, Slow drive area, Nature object, Parking area, Sidewalk, Ego car, Painted driv. instr., Traffic guide obj., Dashed line, RD normal street, Sky, Buildings, Blurred area, Rain dirt	https://www.audi-electronics-venture.de/aev/web/en/driving-dataset.html
A*3D Dataset [86]	Visual cameras (2); 3D LiDAR	2019	3D bounding boxes	Singapore	39k frames, 230k objects	Car, Van, Bus, Truck, Pedestrians, Cyclists, and Motorcyclists; Afternoon and night, wet and dry	https://github.com/I2RDL2/ASTAR-3D
EuroCity Persons [88]	Visual camera; Announced: stereo, LiDAR, GNSS and intertial sensors	2019	2D bounding boxes	12 countries in Europe, 27 cities	47k frames, 258k objects	Pedestrian, Rider, Bicycle, Motorbike, Scooter, Tricycle, Wheelchair, Buggy, Co-Rider; Highly diverse: 4 seasons, day and night, wet and dry	https://eurocity-dataset.tudelft.nl/eval/overview/home
Oxford RobotCar [74], [85]	2016: Visual cameras (fisheye & stereo), 2D & 3D LiDAR, GNSS, and inertial sensors; 2019: Radar, 3D Lidar (2), 2D LiDAR (2), visual cameras (6), GNSS and inertial sensors	2016, 2019	no	Oxford	2016: 11,070,651 frames (stereo), 3,226,183 frames (3D LiDAR); 2019: 240k scans (Radar), 2.4M frames (LiDAR)	Long-term autonomous driving. Various weather conditions, including heavy rain, night, direct sunlight and snow.	http://robotcar-dataset.robots.ox.ac.uk/downloads/, http://ori.ox.ac.uk/datasets/radar-robotcar-dataset
Waymo Open Dataset [84]	3D LiDAR (5), Visual cameras (5)	2019	3D bounding box, Tracking	n.a.	200k frames, 12M objects (3D LiDAR), 1.2M objects (2D camera)	Vehicles, Pedestrians, Cyclists, Signs	https://waymo.com/open/
Lyft Level 5 AV Dataset 2019 [81]	3D LiDAR (5), Visual cameras (6)	2019	3D bounding box	n.a.	55k frames	Semantic HD map included	https://level5.lyft.com/dataset/
Argoverse [82]	3D LiDAR (2), Visual cameras (9, 2 stereo)	2019	3D bounding box, Tracking, Forecasting	Pittsburgh, Pennsylvania, Miami, Florida	113 scenes, 300k trajectories	Vehicle, Pedestrian, Other Static, Large Vehicle, Bicycle, Bicyclist, Bus, Other Mover, Trailer, Motorcyclist, Moped, Motorcycle, Stroller, Emergency Vehicle, Animal, Wheelchair, School Bus; Semantic HD maps (2) included	https://www.argoverse.org/data.html
PandaSet [83]	3D LiDAR (2), Visual cameras (6), GNSS and inertial sensors	2019	3D bounding box	San Francisco, El Camino Real	Announced: 60k frames (camera), 20k frames (LiDAR), 125 scenes	28 classes, 37 semantic segmentation labels; Solid state LiDAR	https://scale.com/open-datasets/pandaset

Figure 108: Datasets Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving - part 1 ([32])

Name	Sensing Modalities	Year (published)	Labelled (benchmark)	Recording area	Size	Categories / Remarks	Link
nuScenes dataset [89]	Visual cameras (6), 3D LiDAR, and Radars (5)	2019	3D bounding box	Boston, Singapore	1000 scenes, 1.4M frames (camera, Radar), 390k frames (3D LiDAR)	25 Object classes, such as Car / Van / SUV, different Trucks, Buses, Persons, Animal, Traffic Cone, Temporary Traffic Barrier, Debris, etc.	https://www.nuscenes.org/download
BLVD [80]	Visual (Stereo) camera, 3D LiDAR	2019	3D bounding box, Tracking, Interaction, Intention	Changshu	120k frames, 249,129 objects	Vehicle, Pedestrian, Rider during day and night	https://github.com/VCCIV/BLVD/
H3D dataset [79]	Visual cameras (3), 3D LiDAR	2019	3D bounding box	San Francisco, Mountain View, Santa Cruz, San Mateo	27,721 frames, 1,071,302 objects	Car, Pedestrian, Cyclist, Truck, Misc, Animals, Motorcyclist, Bus	https://usa.honda-ri.com/hdd/introduction/h3d
ApolloScape [60]	Visual (Stereo) camera, 3D LiDAR, GNSS, and inertial sensors	2018, 2019	2D/3D pixel-level segmentation, lane marking, instance segmentation, depth	Multiple areas in China	143,906 image frames, 89,430 objects	Rover, Sky, Car, Motobicycle, Bicycle, Person, Rider, Truck, Bus, Tricycle, Road, Sidewalk, Traffic Cone, Road Pile, Fence, Traffic Light, Pole, Traffic Sign, Wall, Dustbin, Billboard, Building, Bridge, Tunnel, Overpass, Vegetation	http://apolloscape.auto/scene.html
DBNet Dataset [78]	3D LiDAR, Dashboard visual camera, GNSS	2018	Driving behaviours (Vehicle speed and wheel angles)	Multiple areas in China	Over 10k frames	In total seven datasets with different test scenarios, such as seaside roads, school areas, mountain roads.	http://www.dbehavior.net/
KAIST multispectral dataset [93]	Visual (Stereo) and thermal camera, 3D LiDAR, GNSS, and inertial sensors	2018	2D bounding box, drivable region, image enhancement, depth, and colorization	Seoul	7,512 frames, 308,913 objects	Person, Cyclist, Car during day and night, fine time slots (sunrise, afternoon,...)	http://multispectral.kaist.ac.kr
Multi-spectral Object Detection dataset [91]	Visual and thermal cameras	2017	2D bounding box	University environment in Japan	7,512 frames, 5,833 objects	Bike, Car, Car Stop, Color Cone, Person during day and night	https://www.mi.t.u-tokyo.ac.jp/static/projects/mil_multispectral/
Multi-spectral Semantic Segmentation dataset [92]	Visual and thermal camera	2017	2D pixel-level segmentation	n.a.	1569 frames	Bike, Car, Person, Curve, Guardrail, Color Cone, Bump during day and night	https://www.mi.t.u-tokyo.ac.jp/static/projects/mil_multispectral/
Multi-modal Panoramic 3D Outdoor (MPO) dataset [77]	Visual camera, LiDAR, and GNSS	2016	Place categorization	Fukuoka	650 scans (dense), 34200 scans (sparse)	No dynamic objects	http://robotics.ait.kyushu-u.ac.jp/~kurazume/research-e.php?content=db#d08
KAIST multispectral pedestrian [90]	Visual and thermal camera	2015	2D bounding box	Seoul	95,328 frames, 103,128 objects	Person, People, Cyclist during day and night	https://sites.google.com/site/pedestrianbenchmark/home
KITTI [6], [75]	Visual (Stereo) camera, 3D LiDAR, GNSS, and inertial sensors	2012, 2013, 2015	2D/3D bounding box, visual odometry, road, optical flow, tracking, depth, 2D instance and pixel-level segmentation	Karlsruhe	7481 frames (training) 80.256 objects	Car, Van, Truck, Pedestrian, Person (sitting), Cyclist, Tram, Misc	http://www.cvlibs.net/datasets/kitti/
The Málaga Stereo and Laser Urban dataset [76]	Visual (Stereo) camera, 5× 2D LiDAR (yielding 3D information), GNSS and inertial sensors	2014	no	Málaga	113,082 frames, 5,654.6 s (camera); > 220,000 frames, 5,000 s (LiDARs)	n.a.	https://www.mrpt.org/MalagaUrbanDataset

Figure 109: Datasets Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving - part 2 ([32])

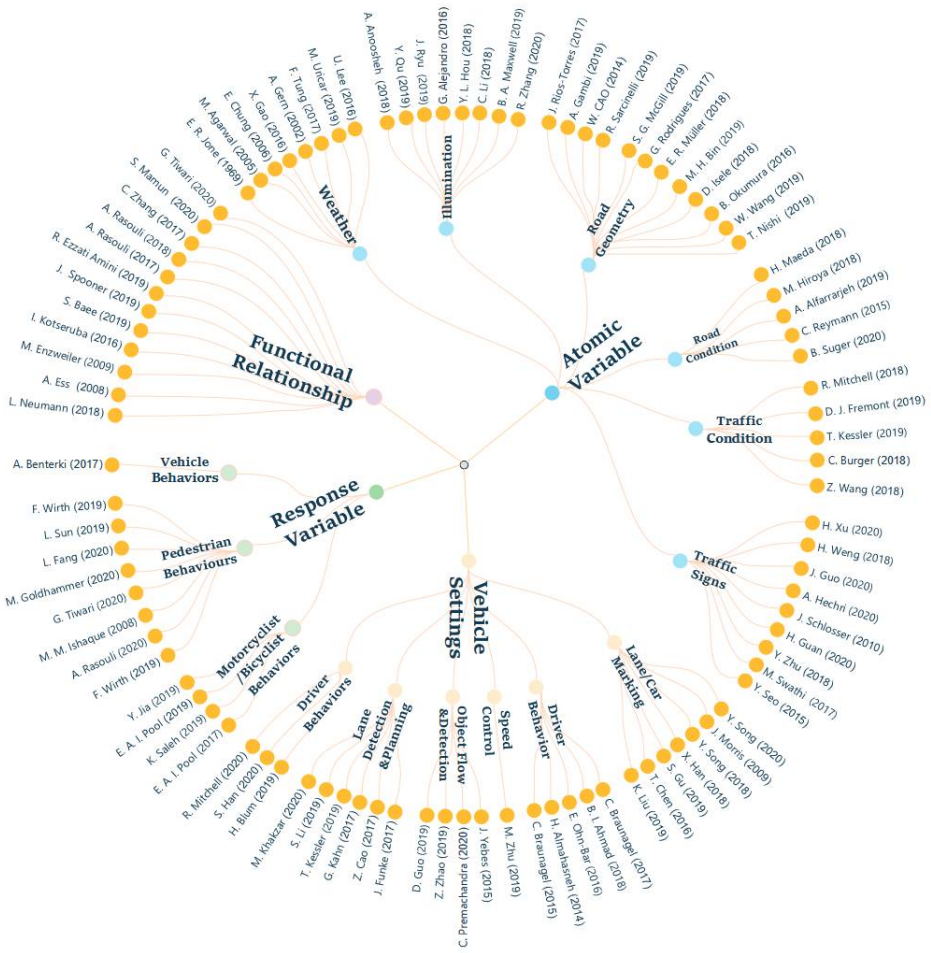

Figure 110: Knowledge Map based Test Influencing Factors Analysis [33]

Factor	Challenging scenes	typical data-sets
Atomic variable		
Weather	Blizzard, Heavy fog, Torrential rain, Overcast clouds	ApolloScape, Dr(eye)ve, FLIR
Illumination	Shadow, Night, Direct sunlight, Alternating light and dark	Daimler Urban, NightOwls, SYNTHIA
Road geometry	Intersections, Roundabouts, Unmapped areas, Country road	Berkeley DeepDrive, UTBM Robotcar, Malaga
Road condition	Bumpy road surface, Unmarked road, Muddy soil, Mountainous area	Road Damage, Waymo, GIDA
Traffic condition	High-traffic, High-speed, Accident-prone intersections	Daimler Urban, Berkeley DeepDrive, KAIST Urban
Traffic signs	Different traffic lights in different regions, Road sign, Road director, Fence	Berkeley DeepDrive, German Traffic sign, LISA Traffic Sign

Factor	Challenging scenes	typical data-sets
Vehicle settings		
Lane/car marking	Broken and missing markings, Irregular lane and road shapes	ApolloScape, CULane, Interaction
Driver behaviors	Distraction, Drowsiness	Dr(eye)ve, JAAD
Speed control	Driving on highways or speed limit section, Sudden braking, Overtaking	Highway Workzones, KAIST Urban, NightOwls
Object flow and detection	Small objects moving at high speed, Flying objects, Suddenly appearing objects	Brain4Cars, DIPLECS, HighD
Lane detection and planning	Irregular Lanes, Crossing multiple lanes, Lanes with traffic rules	CULane, NGSIIM, Argoverse
Semantic segmentation	Difficult to capture or fuzzy mark, Road signs with cultural differences	Karlsruhe Labeled objects, SYNTHIA, LostAndFound

Factor	Challenging scenes	typical data-sets
Response variable		
Vehicle behaviors	Overtaking intention, Sudden movement, Non-compliance role, Walk freely in trod, Whistle, Interactive challenge	Brain4Cars, UAH, Oxford RobotCar
Pedestrian behaviors		Stanford Track, JAAD, TUM City
Motocyclist/bicyclist behaviors		Daimler Pedestrain, Road Damage, ETH Pedestrain

Figure 111: Summary of Autonomous Test Datasets With Scenario and Influencing Factor Concerned [33]

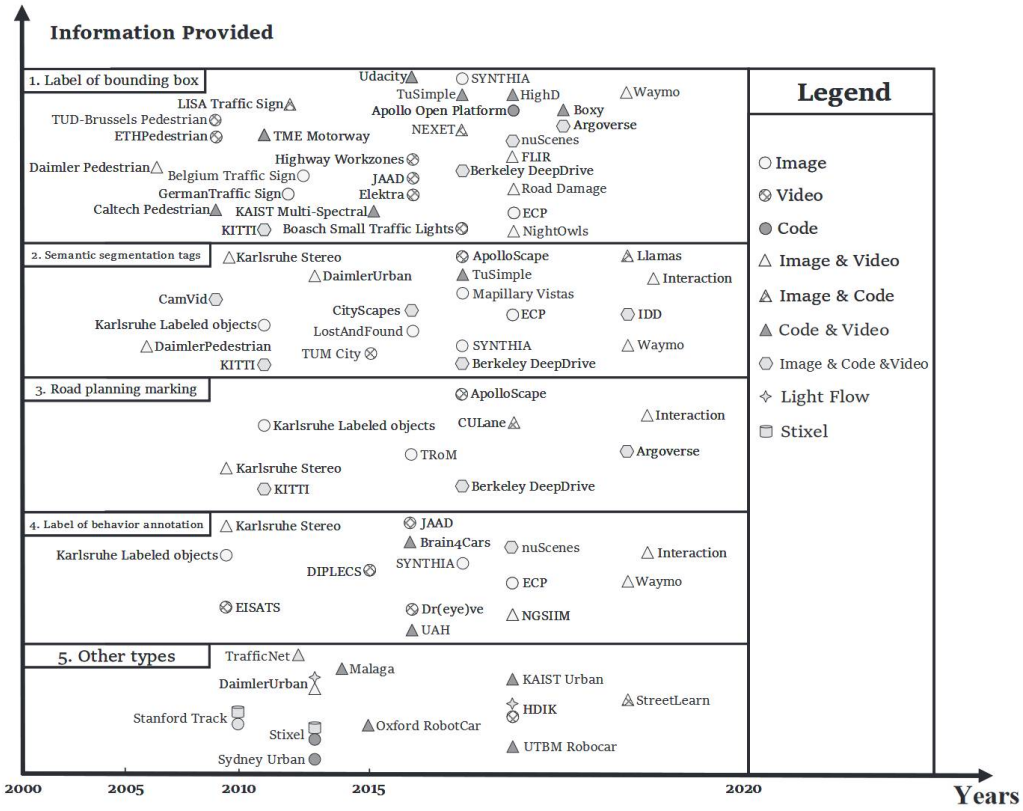

Figure 112: Scenario Driven Autonomous Driving Datasets Survey [33]

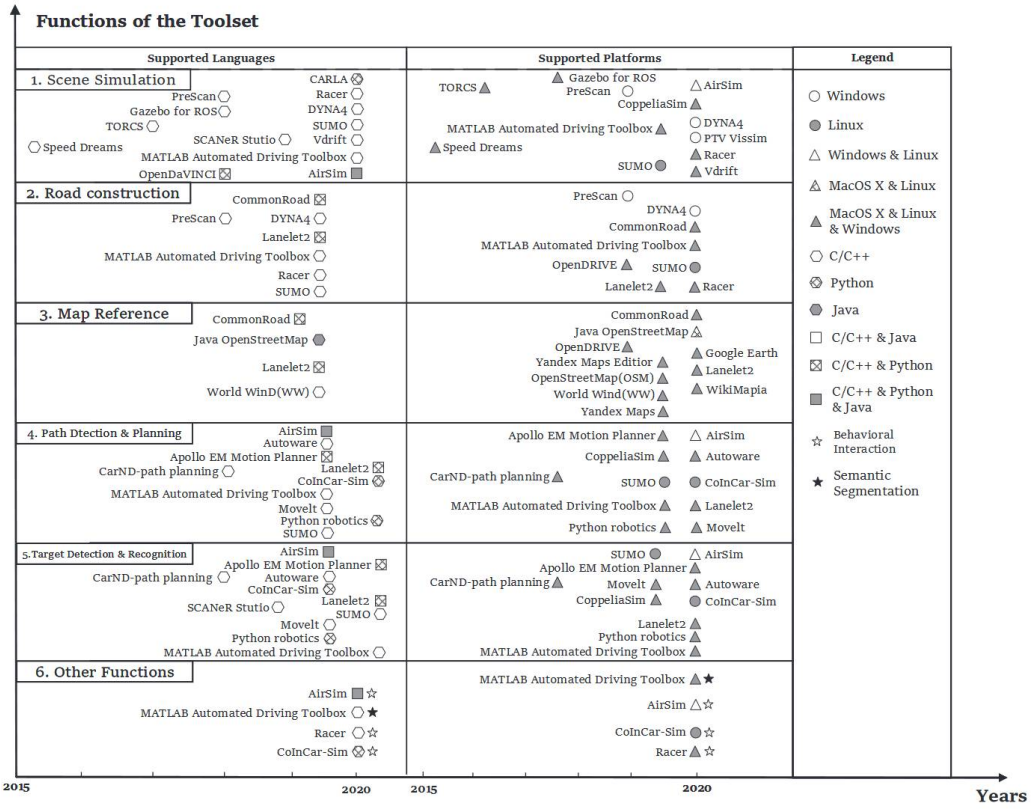

Figure 113: Scenario Driven Autonomous Driving Toolsets Survey [33]

4.7.2 Data sets from virtual data

But these data-set are mainly build from real sensor data. More and more companies propose Synthetic Data-sets for ADAS and Autonomous Driving and mainly for automotive training and validation data.

One way to overcome the limitations of the datasets with real sensor data is by data augmentation via simulation. In fact, a recent work [164] states that the most performance gain for object detection in the KITTI dataset is due to data augmentation, rather than advances in network architectures. Pfeuffer et al. [111] and Kim et al. [110] build augmented training datasets by adding artificial blank areas, illumination change, occlusion, random noises, etc. to the KITTI dataset. The datasets are used to simulate various driving environment changes and sensor degradation. They show that trained with such datasets, the network accuracy and robustness are improved. Some other works aim at developing virtual simulators to generate varying driving conditions, especially some dangerous scenarios where collecting real-world data is very costly or hardly possible. Gaidon et al. [165] build a virtual KITTI dataset by introducing a real to virtual cloning method to the original KITTI dataset, using the Unity Game Engine. Other works [166]–[171] generate virtual datasets purely from game engines, such as GTA-V, without a proxy of real-world datasets. Griffiths and Boehm [172] create a purely virtual LiDAR only dataset. In addition, Dosovitskiy et al. [173] develop an open-source simulator that can simulate multiple sensors in autonomous driving and Hurl et al. [174] release a large scale, virtual, multi-modal dataset with LiDAR data and visual camera. Despite many available virtual datasets, it is an open question to which extend a simulator can represent real-world phenomena. Developing more realistic simulators and finding the optimal way to combine real and virtual data are important open questions.

The main benefits of using simulated driving scenarios and simulated sensor data are:

- **Data Generation:** Driving simulators can generate virtually unlimited data configuration in controlled environment for research and testing purposes. Apart from the main scene images, Simulation stages also allow to export different vehicle configuration data and the data from virtual and realistic sensors attached to the ego vehicle. Furthermore, the data generated can be automatically annotated. This automated annotation stage eliminates the need for the labor-intensive manual annotating process and provide accurate enough ground truth.

- **Varying vehicle, cyclist, pedestrian, and traffic conditions:** The driving simulators include various vehicle and sensor models, pedestrians, and cyclists. The diversity of these road scenes and traffic actors allow training for classification on different shapes, sizes, colours, and behaviours of vehicles, cyclists, pedestrians, and other road users.

- **Dynamic Traffic, Weather, and Lighting Conditions:** The vehicles, sensors, driving simulators provide high fidelity traffic simulation, supporting dynamic changes in traffic density, time of day, lighting, and weather conditions, including rain, snow, fog, dust adverse and degraded conditions. All these aspects and potential disturbances can be generated in a controlled way with repeatability constraints.

- **Rapid Scenario Construction:** Typical road networks can be easily laid out using the in-built tools and are automatically connected for routing and navigation purposes.

It is the case of Cognata company (https://www.cognata.com/datasets/) which provide synthesis data with accurate ground truth. But it is the case of the main popular simulation platforms for AV with a sufficient realistic simulation engine. In these vitual dataset, the following can be mentioned:

- **SYNTHIA** Dataset also provides a collection of synthetic image and annotation. This dataset is one of the best examples of datasets generated using simulated environments for Autonomous Driving research. SYNTHIA consists of photo-realistic frames rendered from a virtual city and comes with precise pixel-level semantic annotations for thirteen classes, i.e., sky, building, road, sidewalk, fence, vegetation, lane-marking, pole, car, traffic signs, pedestrians, cyclists and miscellaneous.
- **Synscapes**: Magnus Wrenninge and Jonas Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. arXiv preprint arXiv:1810.08705, 2018.
- **VIPER**: Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In ICCV, 2017
- **Virtual KITTI 2**: Yohann Cabon, Naila Murray, and Martin Humen-berger. Virtual kitti 2. arXiv preprint arXiv:2001.10773, 2020

4.7.3 Data for interactions between sensors and environment

We have to speak about modelling and data such as BRDF, Antenna diagram, propagation channel, RCS for RADAR, satellites constellation and atmosphere layers modelling.

5 State of simulation environments, platforms and tools

5.1 Simulation Platform Requirements

A simulation platform for AV development needs to be custom-built in order to support the testing based on standards described above. Some of the key requirements include:

- **Easy creation of complex interactions**: The simulation engine should allow developers to put the system under many edge case tests to support SOTIF analysis
- **Requirements traceability**: The simulation platform should support careful traceability on both sides of the development V-cycle to help teams understand high-level coverage and specific failures
- **Supporting unit and integration tests**: Modular testing is important to identify issues and trace quality of subsystems
- **Repeatable results**: The core simulation engine should provide identical results for a well-defined scenario across different compute platforms
- **Configurable fidelity**: Both high-fidelity models such as detailed sensor models and simple mathematical models are required in the platform to support AV teams while minimising computational costs
- **HIL compliant**: Complete system testing including hardware tests is important for final testing and needs to be supported by the simulation platform

5.2 VIL, HIL, MIL, SIL

5.2.1 SPHEREA U-TEST® Hybrid test system state-of-the-art

As a test solution provider for Systems Manufacturers, SPHEREA has delivered test systems in various domains (Aerospace, Railways, and Energy markets) and for various types of usages from early simulation until maintenance test systems.

5.2.1.1 Overview

The first criteria to consider when designing a test system is defining its functional application domain: In which phase of the System Under Test (SUT) lifecycle tests have to be performed? Indeed, three main tests domains can be considered at first level:

- in development phase Tests are targeting Functional checking and qualification,
- in serial production phase, Tests are targeting compliance to validated design checking,
- in operational usage phase, tests are targeting Fault identification and diagnostic.

Those different tests domains were traditionally managed by different teams and as sequential silos. Yet with systems based on hardware and a growing part of functions performed by software, the need to cycle those domains and being able to crosscheck non regression or optimize test campaigns becomes critical. A global management of Test strategy and means over the complete lifecycle of a SUT is the only way to enable efficient upgrade/correction change on deployed SUTs. With AI embedded in autonomous systems, this global Test strategy consistency will become a key driver for successful system deliveries and support.

5.2.1.2 Early design stages: Virtual engineering

In early design phases: the system under test (SUT) is a representative model of a system of interest (SOI). In this case, the test system has to execute the model of the SOI, simulate its environment, and the exchanges between the model and its environment (see section 2.3: Virtual engineering). The performed test if based on full virtual (modeled) test configuration. Some key factors must be accounted:

- Environment and/or the expected SUT may involve different physical fields (mechanic, thermodynamic, power electronic, logical electronics, radio frequency,. . .). Usually modelling and simulation environment have preferred applications fields and may not be valid over all the expected fields. That's why the virtual test is usually associated with integration of heterogeneous simulation engines. The reuse of performing existing simulation models or capabilities enforces the use of standard of execution platforms like FMI or ED247.
- Validity of the simulation: how the results of the simulation is close to the behaviours of the real system? If the simulation is a co-simulation of different kind of simulation models, this aspect can be very complex to evaluate before real system can be deployed for physical tests.
- Is the simulation executed on the same computing node (either real physical node, or logical software node with its own context), or distributed (across multiple physical nodes, or across multiple processes on the same computer)

- If the computing nodes are distributed, how are synchronised the simulations inputs and outputs? Do all these computing nodes have to share a common simulation clock? If so, how are these clocks synchronised between the nodes, and what is the error of synchronisation between these clocks?
- Evolution of the SUT and the test system: when the components of a system are provided by many different suppliers, it's more likely that the several models supplied will have different maturity along the life-cycle of the whole simulation. In this situation, the environment simulated by the test system has to manage these evolutions, and help providing confidence and reuse of the simulations results obtained with the previous versions of the configuration of models. The concept of virtual test bench is helpful for designing and managing the configuration of the whole simulation (composed of the model of the system and the virtual bench).
- In order to ease transitions between the different test stages (Virtual/HIL&SIL/hybrid), consistency on relevant information shall be initiated at this stage :stage: Reuse of existing information: the definition of the system's interfaces at both physical (cable, connector) and logical (data encoding) layer is a long and costly task. The test system must be able to use the available information, whatever how the information's data is stored.

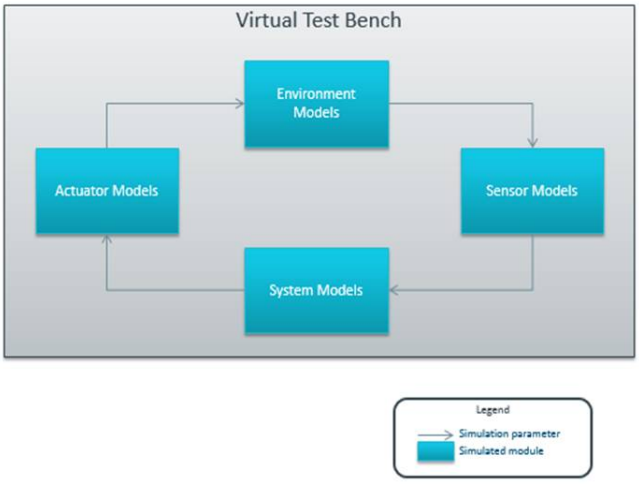

Figure 114: Virtual Bench

In virtual mode, U-TEST® is used to do "Model In the Loop" (MIL) and "Software In the Loop" (SIL). The benefit of MIL is that it can quickly test concepts and models of the system from modelling tools like MatLab Simulink, Scade, AMESim or FMI 2.0. In SIL, the model is now validated under its compiled form.

5.2.1.3 Integration, Verification and Validation: HIL & SIL

Later in the development process of a system of interest, once the real SUT is available and before deploying in its operational environment a test system is required to stimulate the inputs, monitor the outputs and simulate the environment of a system of interest to go through the different test sequences required to integrate, verify or validate the SOI. This kind of test system is named HIL (Hardware in the Loop), or SIL (System in the Loop) depending is the

system under test is complete or not. The second use case of U-TEST system is hardware in the loop.

In a Hardware In the Loop (HIL) configuration, the real equipment in the loop, the bench is used in a way that it can interface with a physical equipment under test. The bench generates signals destined to the equipment, which after processing sends back several data, which are acquired by the bench and then processed in a feedback loop. In order to allow an optimised and efficient IVV campaign of the SUT, one key aspect of the Test System is to allow efficient instrumentation of the complete Test Configuration while performing the test campaign. It implies that time performances of the instrumentation are often more challenging then on the SUT.

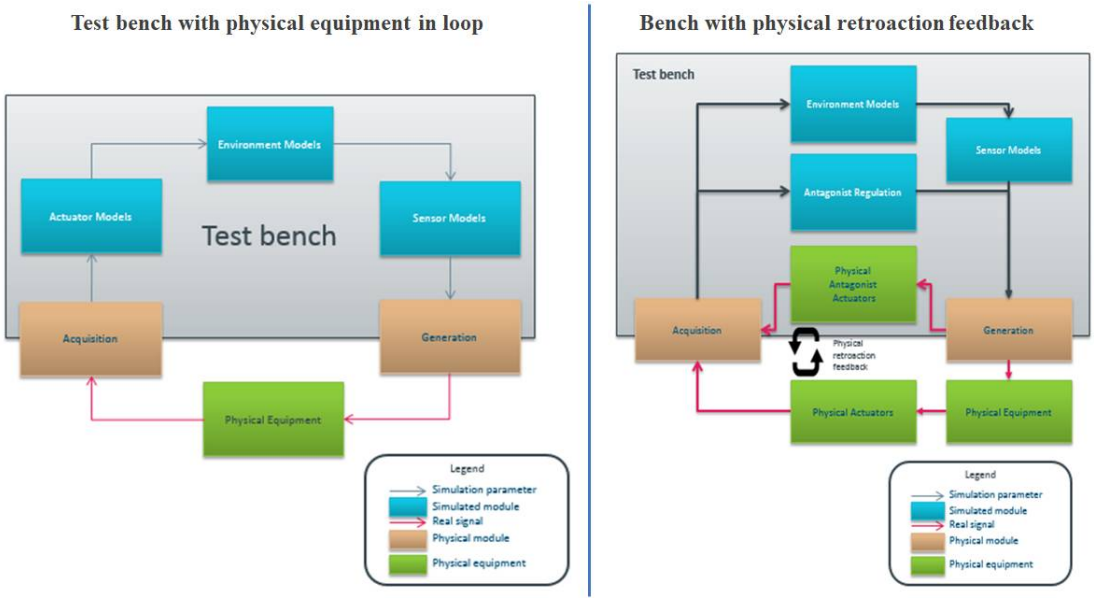

Figure 115: Test bench with physical equipment: left with equipment in loop, right with retro-action feedback

This type of bench is an extension of the HIL bench. Now, the entire system is tested, with the entire configuration of software and mechanical components. Furthermore, in opposition to the hardware in the loop, sensors and actuators are real in this configuration. Their simulation is therefore not possible, they must be stimulated (for sensors) and submitted to contrary-efforts based on the system physical environment model.

In addition to the constraints listed for virtual engineering test system, HIL and SIL test systems brings additional constraints and requirements:

- Real-time loop frequency: this constraints is an extension of the simulation validity listed previously when applied to a real system. Depending on the main timing constraints of the real system of interest, the minimal simulation step of the test system may vary from hundreds of milliseconds to few microseconds. The technology involved greatly depends on this factor.

- Interfacing the test system with the system under test may vary from simple, but often costly, instrumentation resources (analog, digital or bus acquisition/generation electronic devices) to mechanical interfaces specifically developed to adapt the simulated environment with the system under test.

- The allocation of physical instrumentation resources to the simulations channels can be complex and requires dedicated technology to easily route and allocate often thousands of electrical connectors to the instrumentation resource. At logical level, the allocation of simulated channels to their physical counterpart also brings complexity due to the often large amount of combinatorial possibilities.
- Dysfunctional scenario or system resilience check require the test system to be able to inject error both at logical level and physical level, like creating short circuits or open circuits without creating physical damages to the SUT (and the Test Solution).

5.2.1.4 The hybrid test system

As the time-to-markets constraints decreases the delay between concepts and delivery of the systems and the increasing number of components has increased the complexity of the systems, the frontier between pure virtual test systems and pure HIL test systems has more and more became a limit in the transmission of verified definition information between the design offices and IVV departments. Evolution to systems configurations due to software upgrades (corrective or evolution) associated with certification requirements require full cycle test campaign with aggressive schedules and cost management. Test systems is now meant to enable the seamless transition from virtual engineering to hardware in the loop tests, managing the configurations of the virtual systems and virtual testing at the same time as the configuration of the real system of interest and its verification and validation environment. The MPVE cycle illustrates the interleaving of the lifecycle of the simulation models of the system of interest and the system of interest itself. The hybrid test system is the enabling system that enables the transitions between the V cycle of the simulation model and the V cycle of the system of interest.

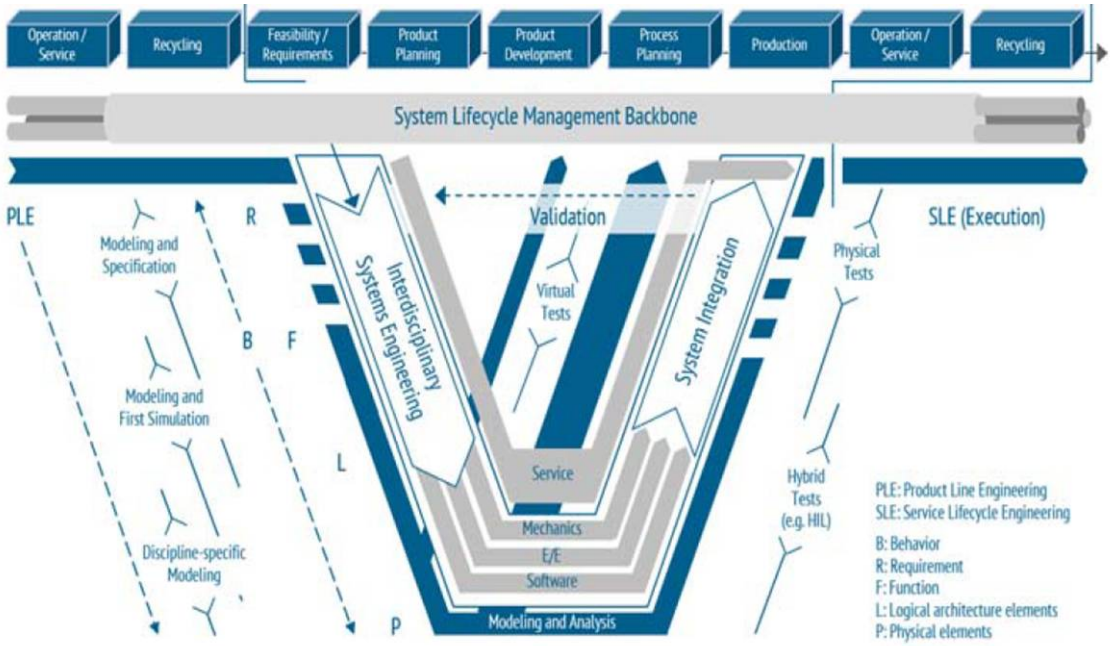

Figure 116: MPVE V cycle

5.2.1.5 U-TEST® test systems

U-TEST® based test systems are able to address the full range of constraints of an hybrid test system. Using open standards for all its interfaces (simulation data exchange, simulation model execution, resources allocations, clock synchronisations, test sequence, test control and command), U-TEST® test suite is an enabler for the development of Hybrid Test Benchs with heterogeneous Models Simulation types (MathWorks, AMESIM, ...) integration with real physical SUT components.

U-TEST® is based on a group of modules and plugins which allow to adapt throughout the system's whole V life-cycle.

During the down-phase of the V life-cycle, U-TEST® is used to validate concepts via fast prototyping and validate models through simulations.

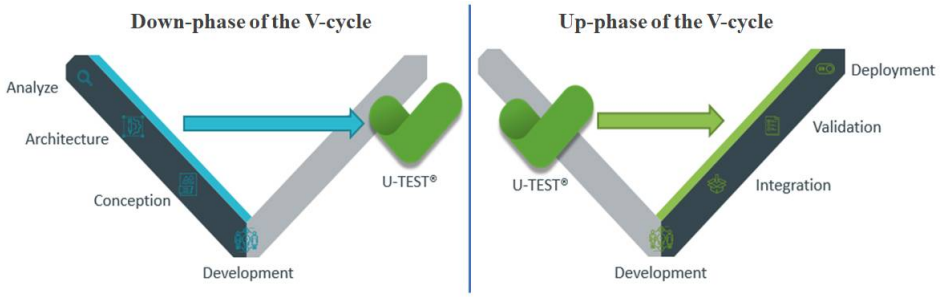

Figure 117: Down-phase and up-phase of the V-cycle

During the up-phase of the V life-cycle, U-TEST® is used during the integration and validation phases of a system before its deployment.

5.2.1.6 U-TEST® workflow

The typical workflow of test campaign is detailed in the figure below:

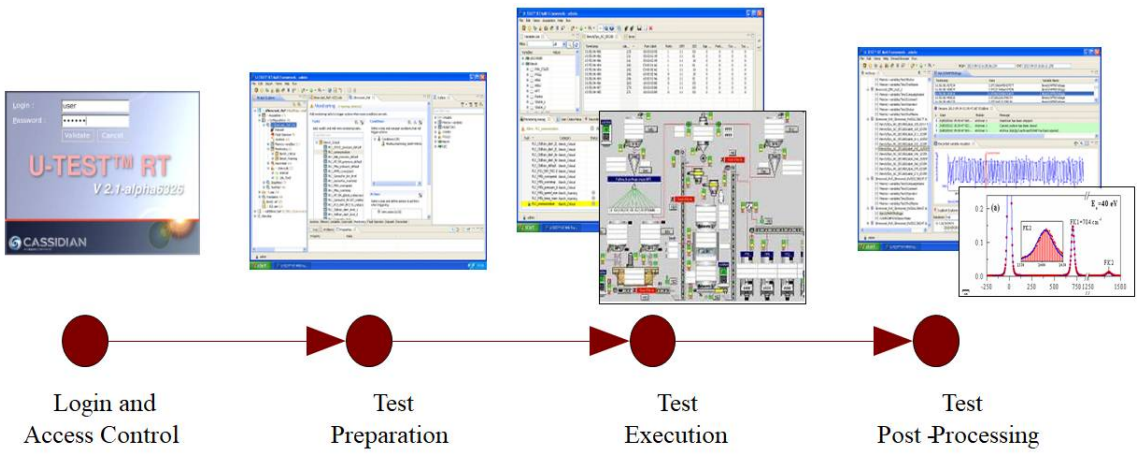

Figure 118: The different stages of utilisation of the U-TEST®

- Login and Access Control

- Test preparation :
  - Import/Create SUT interface configuration
  - Import/Create simulation model
  - Allocate simulation channels to real I/O
  - Import/create replay data
  - Import/Create simulation model HMI (custom synoptics)
- Test Execution:
  - Control & Command the test system and simulations
  - Record data
  - Monitor simulation data
- Test Post-Processing, where the operator performs analysis of the data recorded during the test

5.2.1.7 Synoptics

UI customisation in U-TEST™ enables domain engineers to create dedicated user interfaces for control/command of the whole test environment. A synoptic display is a graphical operator interface that represents the structure and current state of a complex system (a set of devices). It is composed of basic elements so called widgets like labels, meters or bar graphs.

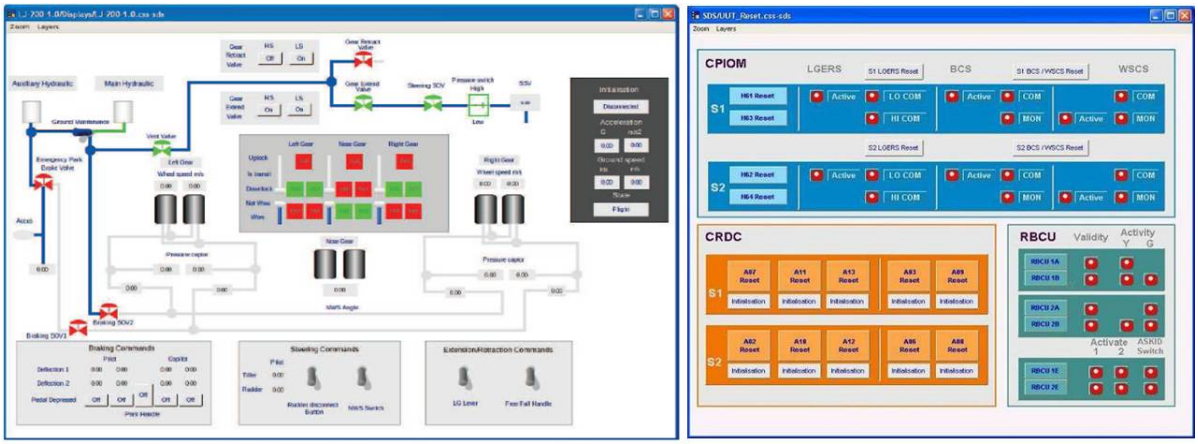

Figure 119: Synoptics of the U-TEST®

5.2.2 ESI-UGE: Pro-SiVIC^{TM}, ImPACT 3D, perSEE, SiVIC MobiCoop

5.2.2.1 Pro-SiVIC: Initial needs and requirements

Since 2002, the Pro-SiVIC platform has been developed in University Gustave Eiffel then by CIVITEC and now ESI group in order to provide an efficient and reliable answer to the prototyping, the test, the evaluation, and the validation of firstly ADAS and now for CAV. For upstream prototyping, it was necessary to have:

- An alternative solution to real experimental means:
  - Prototyping, testing and evaluation of ADAS before actual integration
  - Handling of controlled conditions and environments
  - Definition and exploitation of use cases in boundary and dangerous conditions
  - Design cycle in W
- The possibility of generating and replaying "sensor" databases similar to those obtained in real conditions.
- The possibility to prototype of HIL and SIL applications
- The capability to implement distributed and cooperative applications.
  - Requires a modular, scalable and dynamic software architecture
  - Use of complex and realistic dynamic models
  - Communication bus for communication simulation (NS3)
- An efficient way to prototype new generation and technology of sensors.
  - Requirements specification for non-existing sensors (capability, behaviour, technology ...)
- References and ground truth for ADAS assessment and sensor calibration
- Tools for visual demonstrations involving vehicles, environment and sensors.

In order to provide a answer to this demands, a first architecture with a first set of automotive sensors was developed. This fist version of Pro-SiVIC allowed to guarantee a full loop architecture with an interconnection with RTMaps platform. The fist communication mechanism used optimised FIFO (see figure 120).

5.2.2.2 Efficient communication library for distributed architecture

A couple of year after, in order to prototype and to implement distributed solutions for large scale applications (embedded and infrastructure cooperation) and complex sensors implementation, the DDS library was made. This new way to manage distributed set of synchronised softwares has been used in DIVAS project (HMI in vehicle, fog density assessment from infrastructure, control of Variable Message Sign, communication between car and infrastructure, management of events). (see figure 121)

In FUI eMOTIVE, FUI SINETIC, and ANR ABV projects, the functionalities of Pro-SiVIC were strongly extended in order to prototype CAV. In eMOTIVE, the efforts were focused on the development of new physical realistic sensors (RADAR, GPS, cameras ...), on platforms interconnection (i.e SCANeR, AMESIM, RTMaps ...) with Pro-SiVIC, and the proposal of procedure and protocols for simulation tools and sensor models evaluation and validation. This last topic is very close of PRISSMA objectives.

From all these works, a more generic and inter-operable simulation platform has been developed in University Gustave Eiffel. This architecture is mainly focus on the implementation of an efficient multiple computers, multiple softwares, and multiple Operating Systems architecture. (see figure 124)

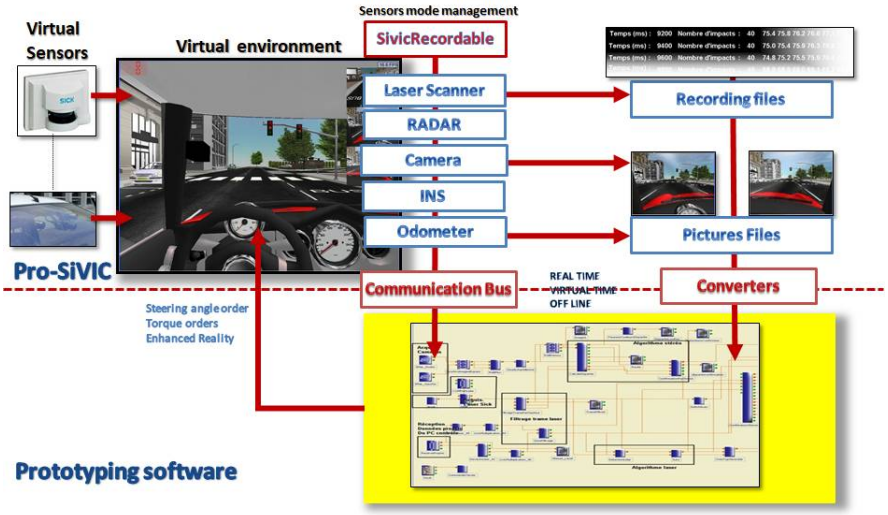

Figure 120: First generation of Pro-SiVIC platform for prototypage, test, evaluation of ADAS (source: Univ-Eiffel)

5.2.2.3 perSEE platform and Have-it project: from virtual to real prototyping

The perSEE platform (see figure 125) has been developed in order to prototype, to test, to evaluate, and to validate embedded real applications, modules, and functions with real and virtual data with a generic way. This development has been done in eFUTURE European project and the PerSEE box has been used in the Tata Motors prototype in the Gaydon test tracks for new generation of active ADAS with a "green aspect" to mixed advantages of automated driving and eco-driving. The goal is to process virtual and/or real data without difference for hardware and software ADAS architecture.The PerSEE platform was made from previous works done in the European FP7 Have-It project. In this European project, an implementation of a highly automated driving system was made on automotive Electronic Control Units (ECUs). It integrates perception components, which use the combination of high-level sensors to map the environment, a co-pilot, which finds an optimal trajectory in this environment, and a control component, which guides the vehicle on this trajectory. The cooperation between human and automation is managed by the driving Mode Selection and arbitration Unit (MSU) and Human-Machine Interface (HMI) components. The co-pilot and control components have been implemented on AUTOSAR-based ECUs, the other components in the RTMaps environment on a standard PC. It has been first been tested on the simulation tool pro-SiVIC and was then transferred to a physical vehicle on test track. It was the first real use of the W design cycle.(see figure 126).

5.2.2.4 SiVIC MobiCoop: Generic and interoperable architecture for CAV prototyping, test, evaluation, and validation

The FUI SINETIC project has for objective to assess if existing simulation platforms and models (communication, vehicles, sensors, propagation channel ...) was good enough to build a full and efficient simulation environment for the design of cooperative intelligent transport systems (C-ITS) involving automated vehicles. In this project, two ways was proposed to address this topic: The system level and the component level.
The objective of the system level was to simulate all the components of the system (vehicle, infrastructure, management centre, etc.) and its realities (roads, traffic conditions, risk of acci-

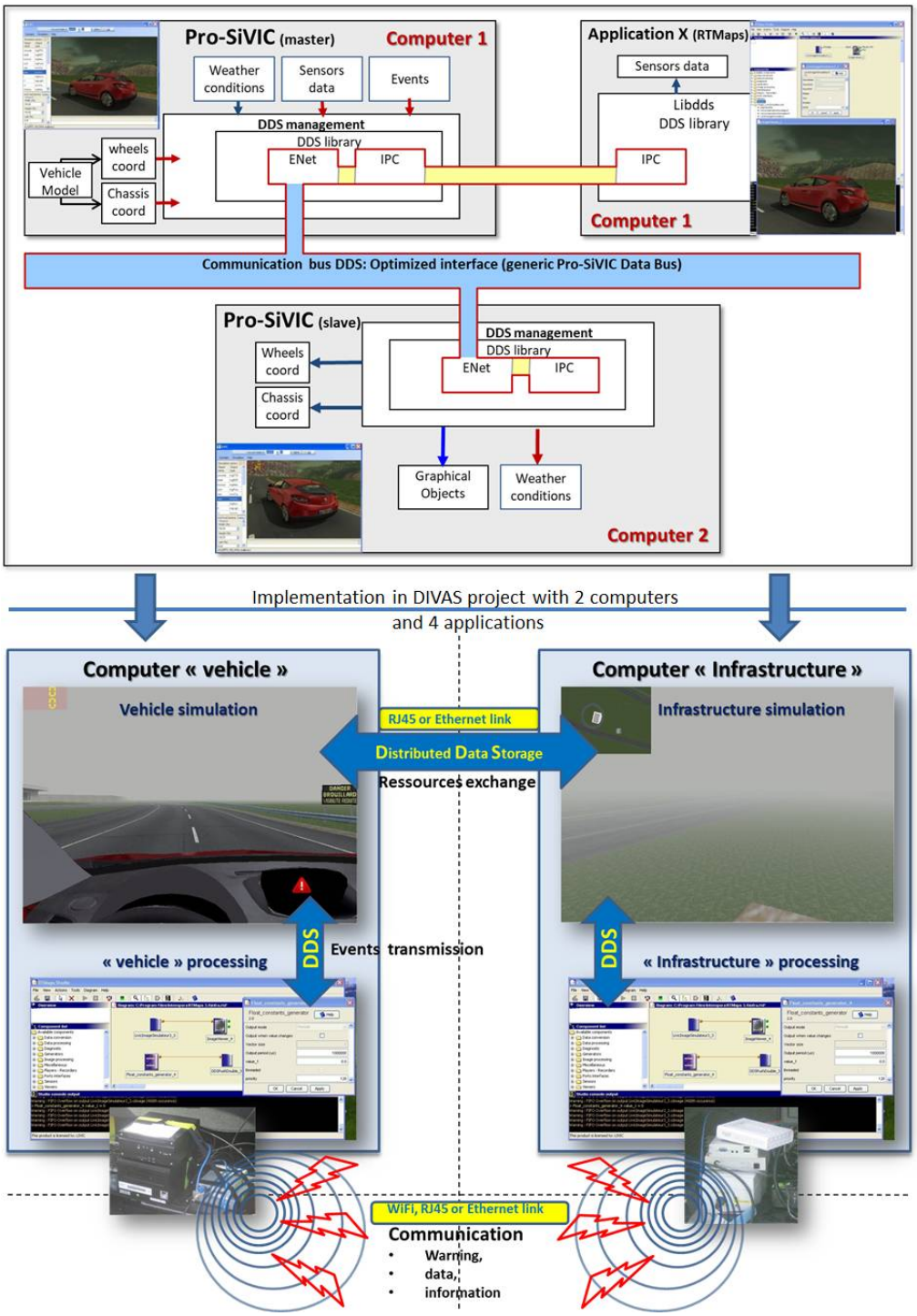

Figure 121: DDS implementation for distributed simulation on several softwares and several computers (source: Univ-Eiffel)

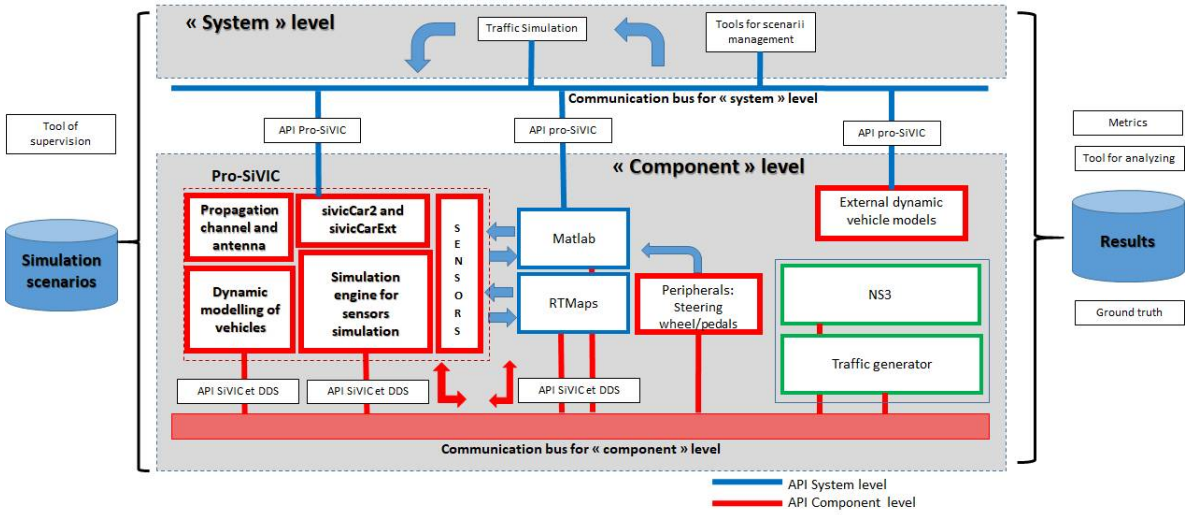

Figure 122: Multi-platform distributed simulation architecture proposed in eMOTIVE and SINETIC projects (source: Univ-Eiffel)

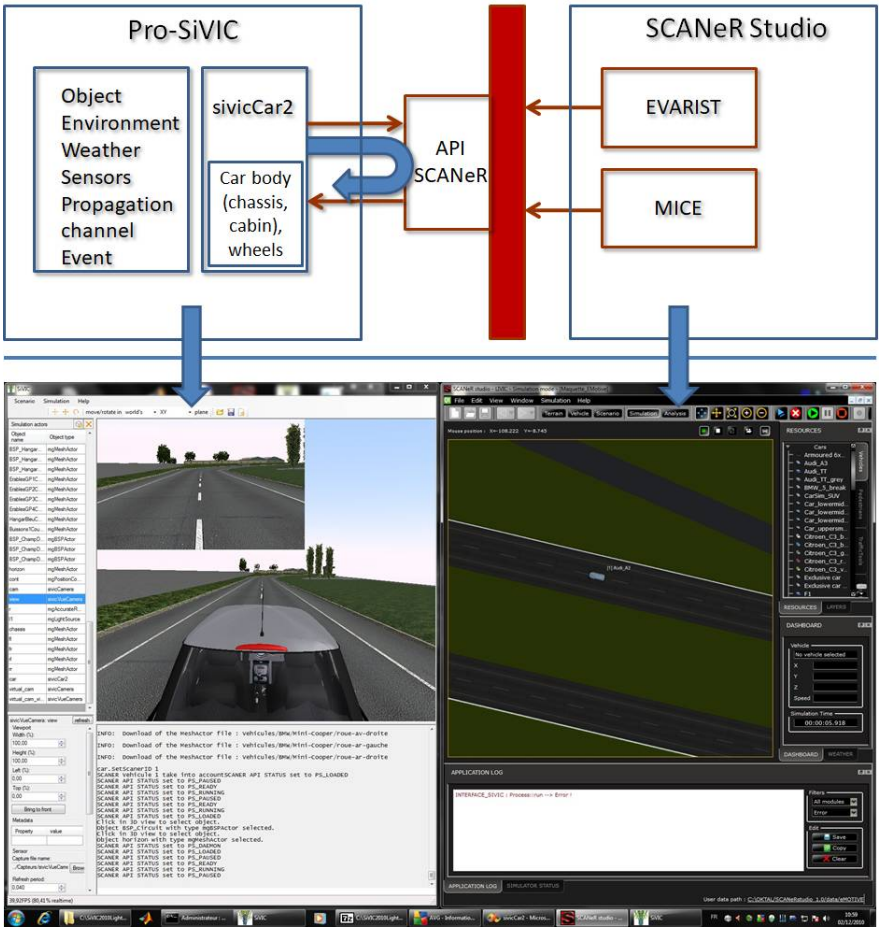

Figure 123: Real time interconnection of Pro-SiVIC (dynamic car modelling) and SCANeR Studio (traffic generation) for a distributed simulation architecture in eMOTIVE project (source: Univ-Eiffel and AVS)

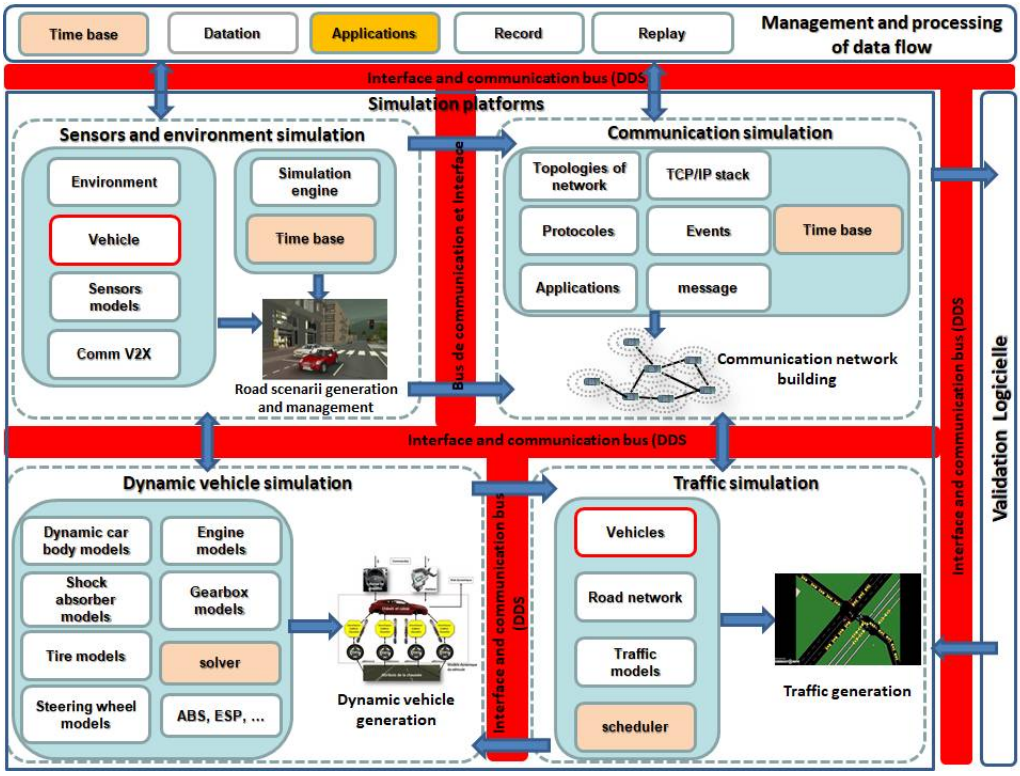

Figure 124: Virtual distributed simulation architecture for CAV prototyping, test, and evaluation (source: Univ-Eiffel)

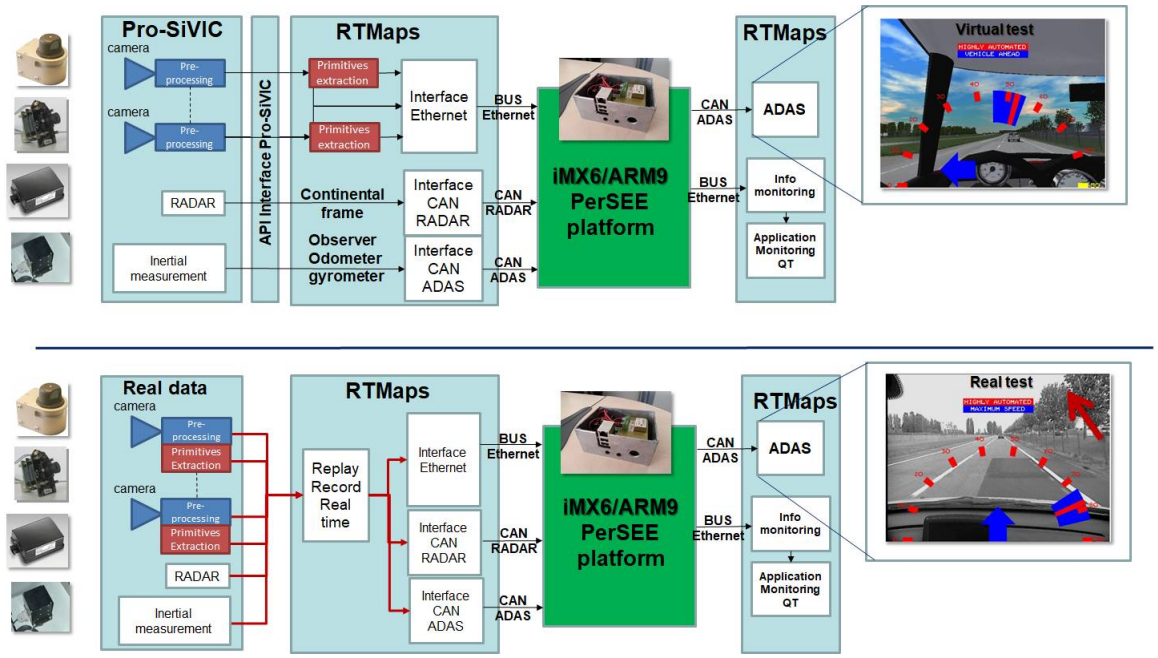

Figure 125: PerSEE: Generic and adaptive platform for evaluation and validation of embedded systems [34] (source: Univ-Eiffel)

Figure 126: Have-It: Highly automated driving on highways: system implementation on PC and automotive ECUs [35] (source: Univ-Eiffel)

dent, etc.) with a simple and low level of modelling. The objective a this level was to validate usage scenarios taking place over a large scale geographic area involving a full city and more than a couple of hundred equipped vehicles.In this level the POC was developed with already existing iTETRIS (NS3, SUMO, and iCS scheduler) and SCANeR driving simulator platforms. The component level was made to accurately model the characteristics and behaviours of the various components (vehicles, sensors, communication systems, etc.) with physico realistic way. This level can be seen as a very accurate zoom of a specific area of the system level. In this level, the area is limited to a quarter and to less than 30 equipped vehicles. The implementation of such a modelling requires the expression of finer characteristics for the C-ITS:

- modelling of the environment in which the vehicles operate;
- modelling of vehicular means of communication;
- modelling of the ”intelligence” either embedded in vehicles or integrated into the road infrastructure.

In the component level (see figure 124, three softwares were selected for the development of the ”component” simulator (second POC):

- Pro-SiVIC, a simulation software for sensors, environment and vehicle dynamics;
- NS-3, a network simulator containing functions for simulating vehicular communications;
- RTMaps, a real-time prototyping software for on-board systems allowing the processing and management of data for intelligent vehicles.

The component level platform is called SiVIC-MobiCoop. This project has proposed 2 POCs with 3 specific use cases (cooperative emergency braking, cooperative platooning, Automated driving and global risk assessment) and a set of KPI, metrics and procedures in order to quantify the quality of the C-ITS applications.
The last 2 years, SiVIC-MobiCoop has been strongly improved with the last version of NS3, the development of a new more efficient communication library (DDsL), and the last version of RTMaps. This simulation architecture is able to generate cooperative frame of perception data similar to the real one. In this way, it is possible to merge both real and virtual communication messages. (see figure 127)

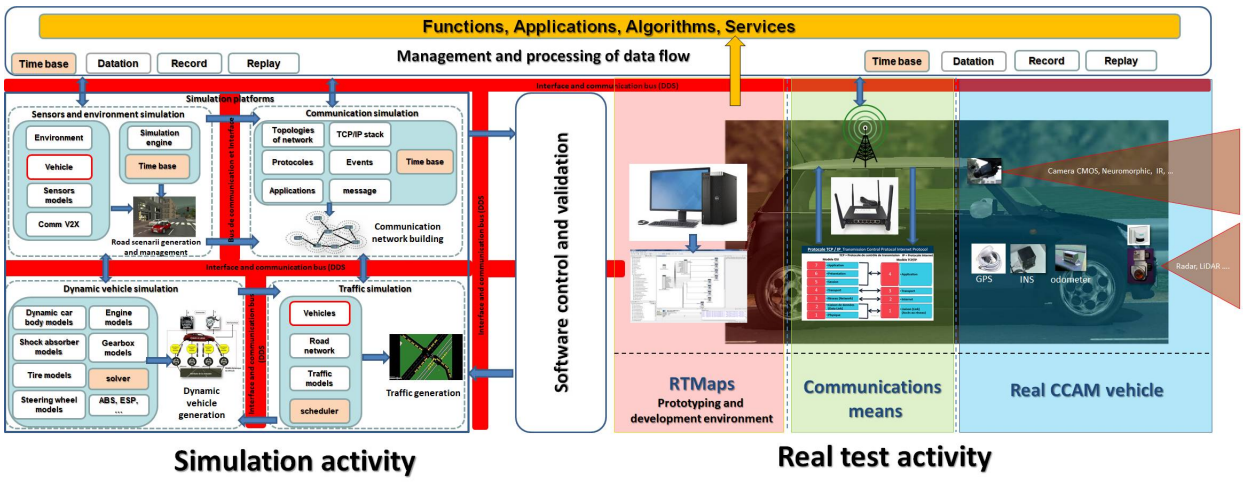

Figure 127: Real and virtual facilities architecture for CAV prototyping, test, and evaluation (source: Univ-Eiffel)

5.2.3 Vehicle in The Loop application

According to Euro NCAP, Vehicle-in-The-Loop is intended to be included in future test protocols. Therefore, some OEMs, Tiers-1, Simulation software providers have launched studies and Proof of concept to explore Vehicle in the loop concept.

5.2.3.1 Porsche - IPG Automotive CarMaker®

A communication video [271] on IPG automotive website (date: August 2020) states that they have supported Porsche in enabling Vehicle in The Loop testing. According to the video, this activity has been done for Verification & Validation of Active Lane Keeping System. As a consequence, Object perception seems not to be included in this Concept, only Road Marking Perception seems to be done.

5.2.3.2 Renault - Prototype Vehicle Equipment

Another project has been led by Renault [272], again powered by IPG Automotive. The use case are quite similar to the Porsche one. The few differences are that the function under test is Adaptive Cruise Control.

Figure 128: IPG CarMaker®and Porsche screenshot

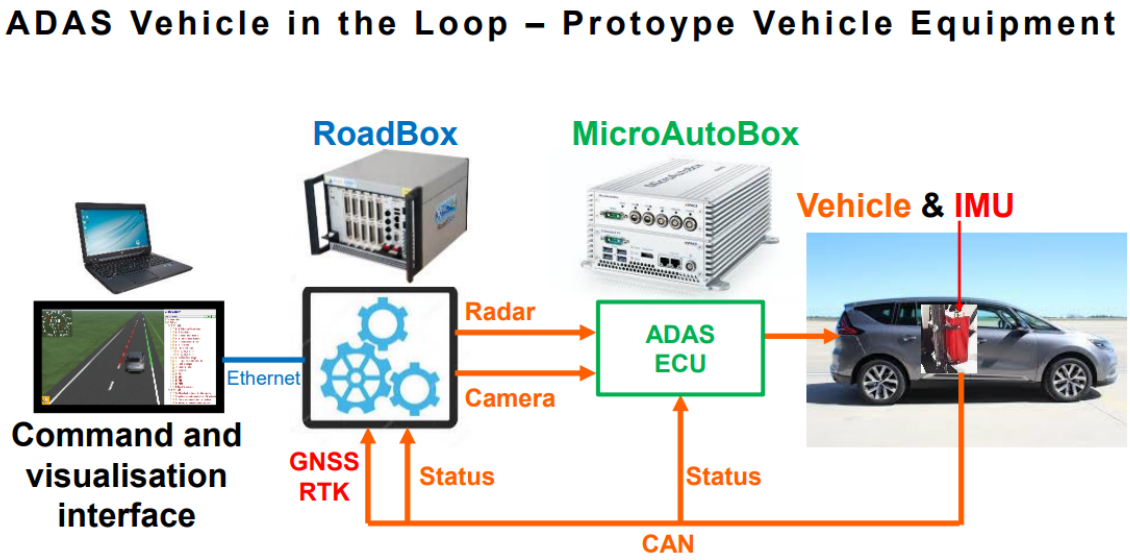

Figure 129: Renault architecture proposal for Vehicle In The Loop

5.2.3.3 IDIADA CAVRide platform for Vehicle in The Loop

CARLA Simulation software has also been used for Vehicle in The loop testing. The function under test is AEB and for this particular test, VRU Child crossing the road at 4 kph. This project was run by IDIADA [273].

Simulation Software	Function Under Test	Comments
IPG CarMaker	Lane Keeping System	OEM or Tier-1 : Porsche Lane Perception emulated No raw data injection
IPG CarMaker	Adaptive Cruise Control	OEM or Tier-1 : Renault Lane and Objection perception emulated No raw data injection
CARLA	Autonomous Emergency Braking	OEM or Tier-1 : IDIADA Object perception emulated No raw data injection

Figure 130: IDIADA showcasing their ViL Concept

As a synthesis, we can find here, a recap table with ViL activities and the Simulation Software used :

5.2.3.4 ESI-Univ-Eiffel's platform for HiL and ViL

In the frame work of several project, several HiL and ViL architectures have been developed and proposed by University Gustave Eiffel and ESI. The first one was made with a road making painting robot (see figure 131). The interconnection of both Pro-SiVIC and the painting Robot allowed to prototype a new accurate control algorithm for slow speed robot. The robot replaced the images coming from its embedded cameras by the raw images generated by Pro-SiVIC in order to detect and to follow the road marking in the virtual environment. The full loop was made in this platform.

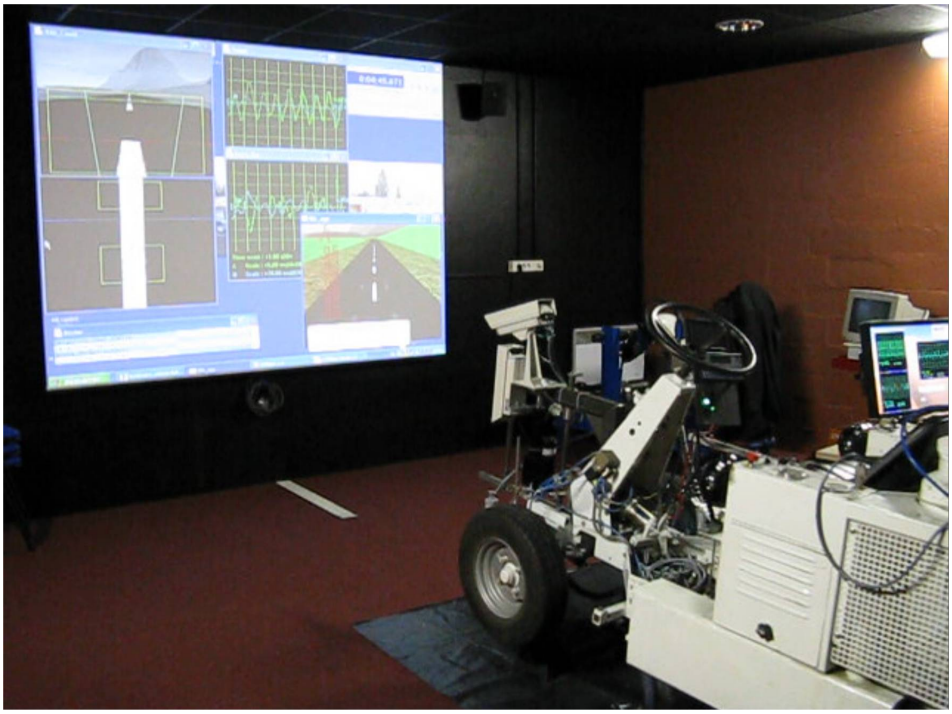

Figure 131: University Gustave Eiffel: Interconnection of real painting robots and Pro-SiVIC (source: Univ. Eiffel)

In the framework of eMotive project, an interconnection of Pro-SiVIC embedded in a real automated prototype (307 Peugeot). The goal was to validate sensor models implemented in Pro-SiVIC in real conditions on the Satory test tracks. In the embedded instance of Pro-SIVIC, several functionalities was used: a GPS model, a camera model, the vehicle dynamic, the digital twin of the Satory test track. In same time and in real time we collected and used RTK GPS, Trimble GPS, and camera data. The use of RTK GPS data allowing in real time to control the Pro-SiVIC camera in order to be positioned at the same location. In same same we collected simulated GPS data with real Trimble data in order to validate the simulated model (see figure 142). Finally a third platform was made with the same Prototype in order to obtain a full virtual immersion of the driver in the real vehicle (with an OCULUS helmet). The objective of this platform was to allow, for a driver, to drive in a virtual environment (highway, urban, rural areas) but with a real filling of the vehicle dynamics. This platform was developed in 2 project (PARTAGE and SINETIC). First results have been presented in [274] with the interconnection of SCANeR and RTMaps. In this platform, a scenario generator provide the virtual environment rendered in a RV helmet (Nvisor MH60). A RTK GPS provides the positioning and orientations of the real vehicle and a projection was made in order to control the virtual position in real time. In this platform, the real driver with the RV helmet drove the real car and evolved in a virtual urban area. In this virtual area, the goal was to react to a virtual pedestrian crossing the road behind a bus. The driver had to react to this sudden event. If not, an emergency braking system reacted to this event. The continuity of this work was done with Pro-SiVIC and RTMaps in the FUI SINETIC project with a new RV helmet (OCULUS). All this previous platform will be merge in the Univ-Eiffel's ImPACT 3D platform.

Simulation Software	Function Under Test	Comments
Pro-SiVIC and RTMaps	Road marking detection Road marking following	Univ-Eiffel Real road marking detection raw data injection (image from virtual camera)
Pro-SiVIC and RTMaps	GPS and ego-localisation	Univ-eiffel real Localisation algorithm and camera control raw data generation (GPS)
SCANeR and RTMaps	immersive driving emergency braking with VRU (driver or ADAS)	Univ. Eiffel and AVSimulation real Localisation data camera control real emergency braking, obstacle detection emulated headtracker system
Pro-SiVIC and RTMaps	immersive driving	Univ. Eiffel and ESI real Localisation data camera control headtracker system

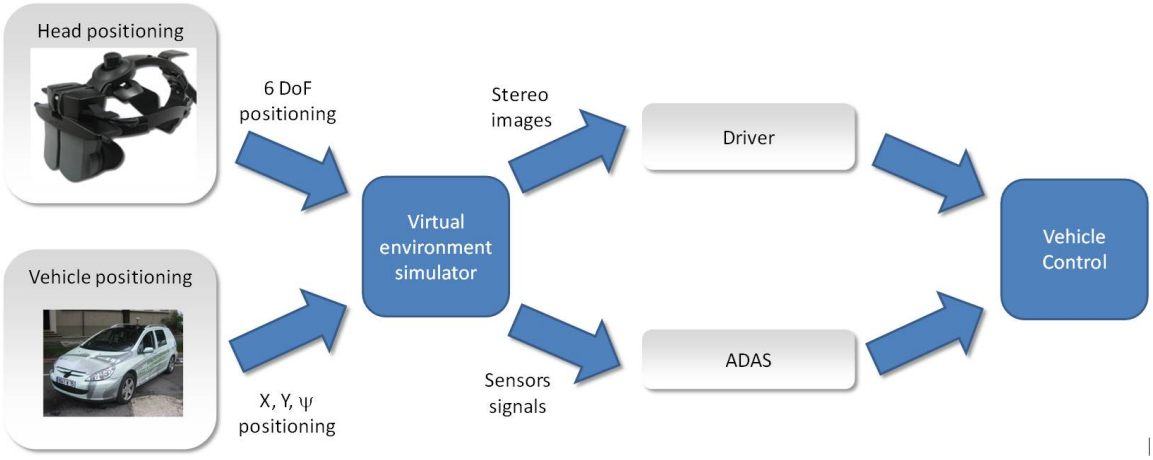

Figure 132: Architecture of the VR ADAS test system

Figure 133: Mapping of the virtual environment on Satory test track

5.3 Sensors and information sources

Autonomous transportation systems seek to automate partially or fully tasks typically performed by a human operator. These tasks are numerous and they range from understanding a vehicle's surroundings, interacting in a smart way with other actors in the scene and executing a sequence of maneuvers to navigate safely across the environment.

Scene understating, in particular, is an area of great interest to researchers and engineers working on smart mobility systems. Data coming from different kinds of sensors are the input to perception and localization algorithms that aim at detecting and classifying objects in the ego-vehicle environment. Giving the great importance of high-quality sensory data in the development of safe and performing ADAS and AD systems, sensors models having been a major R&D subject and many organisations have worked on developing such models.

As shown in the Figure 134, AVSimulation embeds in its simulation platform, SCANeR™ studio, three different levels of sensor models that are optimised for specific needs and use-cases. This modelling is based on the following interpretation of a real-world smart sensor: a device that processes signal propagating in the real world and outputs a object list to the ADAS system. It can be understood as a sequence of different stages; in the **preprocessing** one, the sensor detects a physics property (e.g. luminosity) in the environment, performs a set of initial operations and outputs raw data (e.g. a gray-scale image) to the second module, **detection /signal processing**, which produces a list of relevant objects in the scene (targets list), that is finally fed into the **object tracking** module, responsible for tracking a given object over time.

In simulation, the three levels of sensors (L1, L2 and L3) interact with the virtual environment (i.e. extract information from it) through one or more layers (semantic layer, 3D world model and physics layer). Conceived primarily for automotive applications, SCANeR™ studio uses the open format RoadXML©[275] to describe a terrain's semantic layer.

The semantic layer contains a high-level description of the environment in terms of its topology (element's location and connections with the rest of the network) and logical information (element's signification). The second layer corresponds to the 3D world model, that contains a terrain's geometric description (meshes, triangles and textures), as well as the full environment 3D representation.

The level 1 corresponds to an ideal representation of the real-world sensor. It uses the 3D simulated world and its semantic layer to deliver the full list of objects detected to the ADAS/AD system. A L1 sensor applies geometrical detection alongside with the semantic layer to compute the object list. Users may add a statistical noise model to introduce generic disturbances to the model. The main advantage of such level of modelling is the efficiency (real time implementation) and simplicity. But, since it does not include the underlying physics phenomena into the model, it lacks realism when compared to the other two levels of sensor modelling: level 2 and 3.

The level 3 sensor model corresponds to the most realistic one, and it incorporates into the computation information about the physics layer of the environment. This layer describes the 3D World model with physics based properties (material light reflection, EM waves propagation, temperature sensibility etc). The aim of this model is to deliver super-realistic raw data that can feed the real sensor processing. It does that by simulating the electromagnetic waves propagation, using a ray-tracing technique of solving the asymptotic formulation of Maxwell's equations. The EM radiation emitted by the sensor model into the 3D virtual world will potentially hit millions of surfaces (called "EM contributors") and interact with them in four different ways, according to the physical optics: reflection, diffraction, transmission and scattering. Part

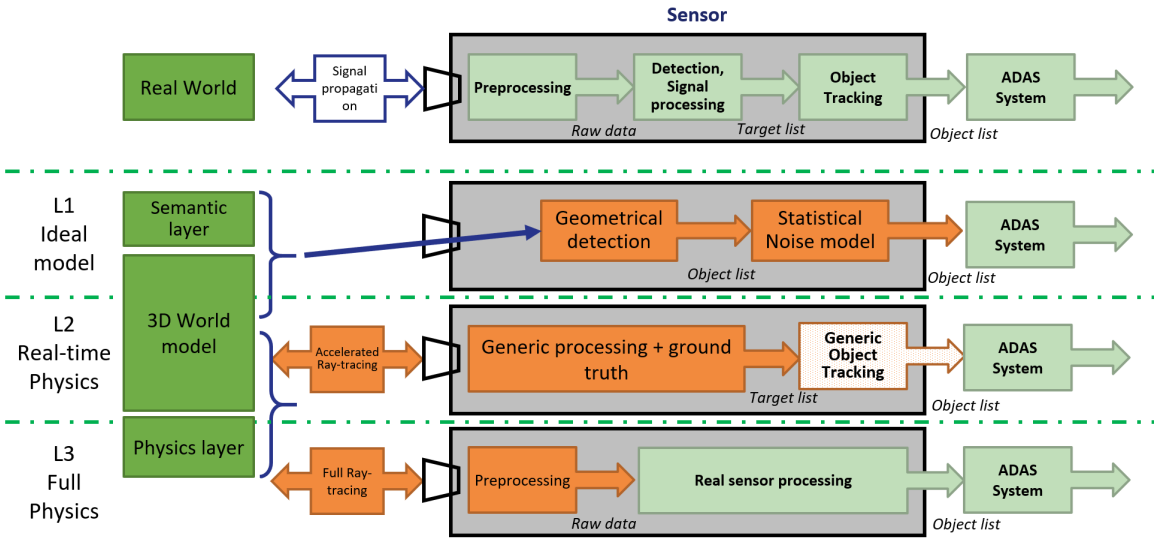

Figure 134: Sensor modelling levels present on SCANeR^{TM} studio.

of the emitted signal is them detected by the sensor, that goes into a preprocessing stage that finally outputs realistic raw data.

Since a large amount of EM contributors are computed at each simulation step, the L3 sensor model is not prone to run in real time. In order to keep the physics based aspect of it and yet deliver a real time implementation (under certain conditions) the Level 2 sensor model has been introduced. It adopts a slightly simplified ray-tracing technique, optimized for decreasing the execution time while keeping a realistic physics-based behaviour. At this level of modelling, a generic processing is used (alongside with ground truth provided by the 3D layer) to generate a target list.

5.3.1 Real time implementation and use

5.3.1.1 SCANeR^{TM} studio sensors package

Several level-1 sensor models are available within SCANeR^{TM} studio simulation environment, namely Radar, Camera, LiDAR, GPS, E-HORIZON and Light Sensor.

The **camera** model can have three types of outputs: the raw image of the scene, a list of detected targets in the sensor's field-of-view and information about the road markings. This list of output options should cover the needs of teams working on perception algorithms (interested in the raw image) as well as teams working on sensor fusion or path planning, interested on target lists usually generated by a smart sensor, e.g. [276] that detects and tracks vehicles on the road ahead providing range, relative speed and lane position data.

The **LiDAR** (LIght Detection And Ranging) is a 3D environmental mapping device using laser remote sensing. The device sends laser beams into several directions and the time taken to receive the reflection emitted by the target object makes it possible to assess the distance to the latter. Therefore, it is possible to establish a point cloud of the environment. Beyond the geometry, additional information can be obtained through the intensities of the points obtained. The intensity depending on the physical characteristic of laser beams and receiving surfaces. In SCANeR^{TM}, the intensity is modelled by the bidirectional reflectance distribution function of the materials used to describe the SCANeR objects. These make it possible to represent in particular the reflectance, and the nature of the surfaces. The image 135 shows a comparison

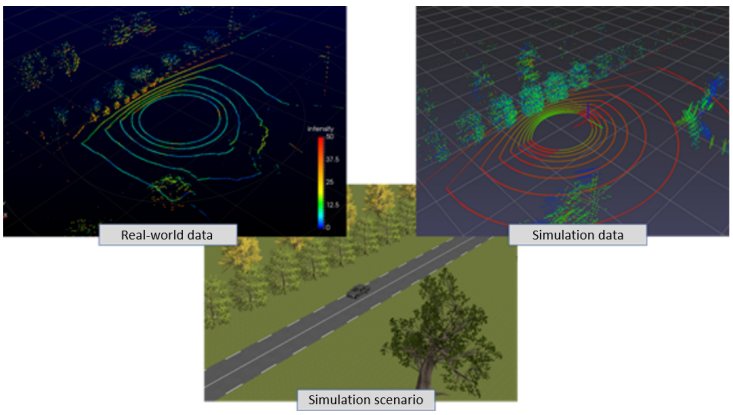

Figure 135: Lidar L1 on SCANeR

between data from a Velodyne VLP 16 Lidar and data collected from simulation.

The GPS sensor model outputs the position of a vehicle in WGS84 coordinates (latitude, longitude, altitude) using RoadXML project information. It outputs data that may be fed into the E-HORIZON sensor model, whose goal is to describe the road geometry with its related attributes based on a vehicle's position and a digital map (ADAS horizon). The purpose of the light sensor is to compute the illuminance received bu the sensor from light sources located into its field-of-view.

5.3.1.2 Sensor-realistic simulation with Ansys AVxcelerate

With the 21R3 release of AVxcelerate, Ansys published an API for physics-based sensor and headlamp simulation, compatible with any driving (or flying) simulator from the market. Connectors with AVSimulation SCANeR^{TM} and IPG CarMaker are provided out of the box.

For sensor-realism, Ansys uses an accurate description of the environment, and provides a library of materials describing di-electric properties for the radar and measured optical properties for the camera and lidar simulations. The optical library describes the reflectance (brdf: bi-directional reflectance distribution function) for every incident / reflection angle of light, for the visible range and up to 1600nm for the infrared. These measurements come from proprietary optical measurement devices (figure 136).

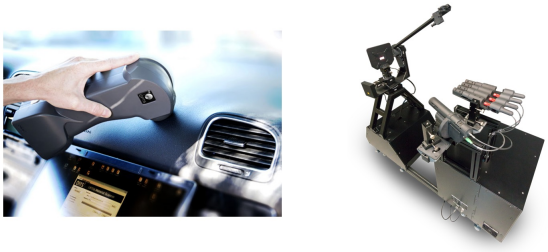

Figure 136: Optical Measurement Devices for camera and LiDAR simulations (source: Ansys)

All sensor models aim to mimic the real sensor behaviour in real-time (all simulations are running on the GPU). Outputs are always provided in the International System of Units:

- The **camera** model uses the accurate light and spectral material description mentioned previously. For a front camera, usually behind a windshield, the camera model will take

into account the windshield distortion, the camera lens (lens distortion and transmission, lens coating, aperture size, focal length) and the camera imager (resolution, size, exposure, noise, dynamic range, bit depth, gain, framerate, ...). Put together, the lens and imager models simulate the conversion of photons into electrons inside the camera. This camera model can both be used for Software in the Loop and Hardware in the Loop with an injection box. Fisheye cameras can be simulated. A typical use case in the industry would be first to use Ansys SPEOS for packaging studies (to find the best location of each camera and visualise their field of views), and second to use Ansys AVxcelerate for real-time simulation in open or closed loop simulations.

- The **thermal camera**: through a technical collaboration, FLIR Systems have integrated a fully physics-based thermal sensor into Ansys AVxcelerate to model, test, and validate thermal cameras designs within an a realistic virtual world.

- Like the camera, the flashing and rotating **LiDAR** models use the same accurate light and material description to simulate the raw waveform output. The user can describe the shape, peak and wavelength of the light pulses generated by the emittor, and a receiver model can be defined with parameters describing its resolution, distortion and sensitivity. The ability to simulate retro-reflective materials contribute greatly to the accuracy of the simulation, as we find many retro-reflective materials on the road (traffic signs, cats-eyes, ...).

- Whilst real-time physics-based **radar** simulation has always been a challenge for the community, Ansys introduced in February 2021 the first real-time simulation of MIMO (Multiple-Inputs Multiple-Outputs) radars, characterised by a high number of transmitters and receivers. A single-channel radar operating in a realistically complex traffic environment can even go up to 160 fps ([277]), so it is also possible to simulate several radars in real-time. The radar model simulates micro-doppler effect created by the vehicle wheels, walking / running pedestians and cyclists. In terms of output, the user can access either a range-doppler map (figure 137) or the raw radar output (I/Q data), from which an angle of arrival and a point cloud can be generated for each target. False radar detections regularly seen on the road due to metallic objects can be simulated with the Ansys real-time radar.

5.3.1.3 Physical and realistic modeling in Pro-SiVIC

In case of microscopic and nanoscopic simulation of the sensor behaviour, it is necessary to develop physical and realistic sensor models. This type of modelling need to implement different layers: wave modelling, electronic modules modelling, antenna modelling, and propagation channel modelling. In the figure 138, a presentation of the different classes of disturbances impacting the data and the operating of a sensor is given. The set of propagation channel's disturbances is not exhaustive but give a good enough overview of what we need to consider in order to obtain accurate enough sensor model.

University Gustave Eiffel and ESI group have applied this type of modelling for a great number of sensors: RADAR, LiDAR, cameras, GPS, INS, Odometer, consumption, ... In Pro-SiVIC several type of automotive sensors with different types of modelling are provided:

- **The LiDAR:** different level of LiDAR are available. The more complex version is based of specific raytracing. A less complex model uses an adapted version of the depth map.

Figure 137: Radar simulation from Ansys, here in co-simulation with SCANeR^{TM} (source: Ansys)

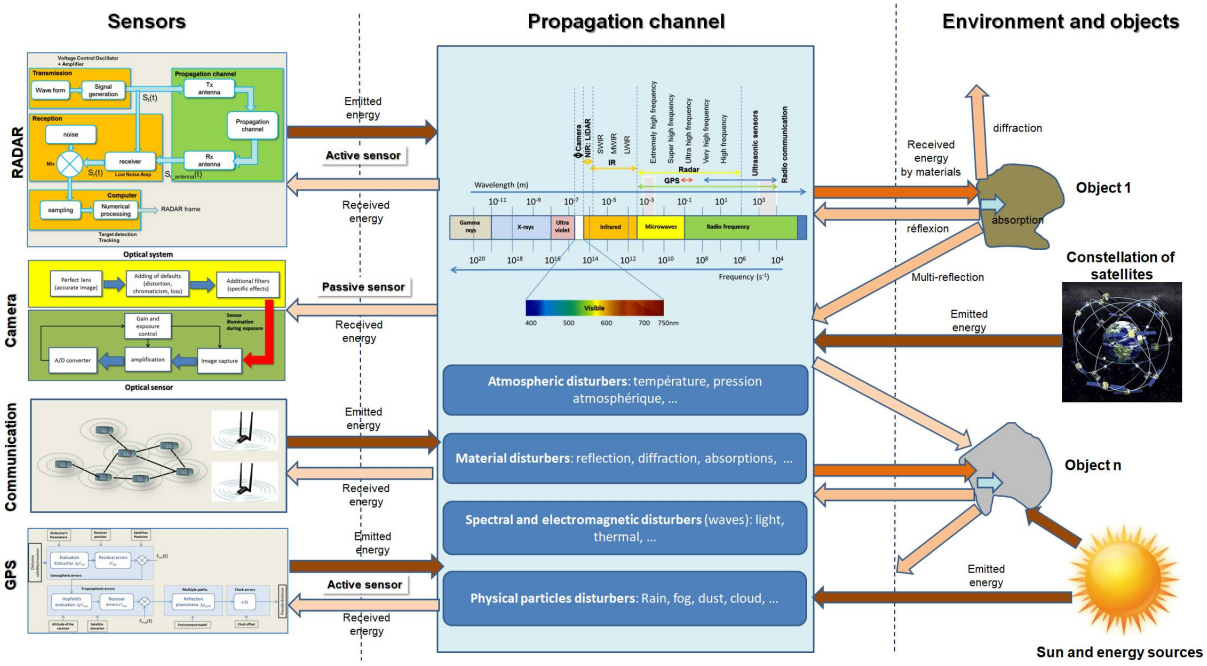

Figure 138: The different types of disturbances in the propagation channel (source: Univ Eiffel)

Moreover a set of filters allows to take into account a great part of the existing disturbances (dust, rain drop, fog ...). The obtain results are available in [278] and are shown in the figure 139. This sensor model considers automotive LiDAR disturbances generated by rain drop, fog, and dust clouds caused by natural phenomena, mechanical or man-made processes like a travelling vehicle and has been validated in comparison with real LiDAR.

- **Cameras:** Pro-SiVIC provide a great set of camera models involving CMOS cameras, omnidirectional cameras (cyclop), and fisheye cameras with the same disturbances than in real life (light adverse conditions, nigh conditions, rain and fog configuration). This sensor has been validation with DXO bench and with real cameras [183]. This model also takes into account HDR texture and the different features of a real camera (Noise, Fog, Optical deformation, Glare, Rain and rain drop impact and effects, Depth of Field, Self exposure, Auto focus ...). Some screenshot are provided in figures 86 and 141.
- **RADAR:** several level of RADAR are available. The more realistic and complex model is based on GPU processing and a Illumination of the road scene by a electromagnetic wave. This modelling considers the electronic part, the wave modulation/propagation/processing, the antenna, and the propagation channel in order to rebuild a signal very close to the real one with multiple reflexions and doopler effect. Figure 140 shows some data coming from this sensor.
- **GPS:** The GPS model is based on an optimised raytracing mechanism with the modelling of the different atmospheric layers. More over, a multiple reflection mechanism is available and interact with the environment and the existing building. Some tests have been made on the Satory test track with natural GPS, RTK GPS and the simulated GPS embedded in a real car on the real test track. The result is shown in the figure 142.

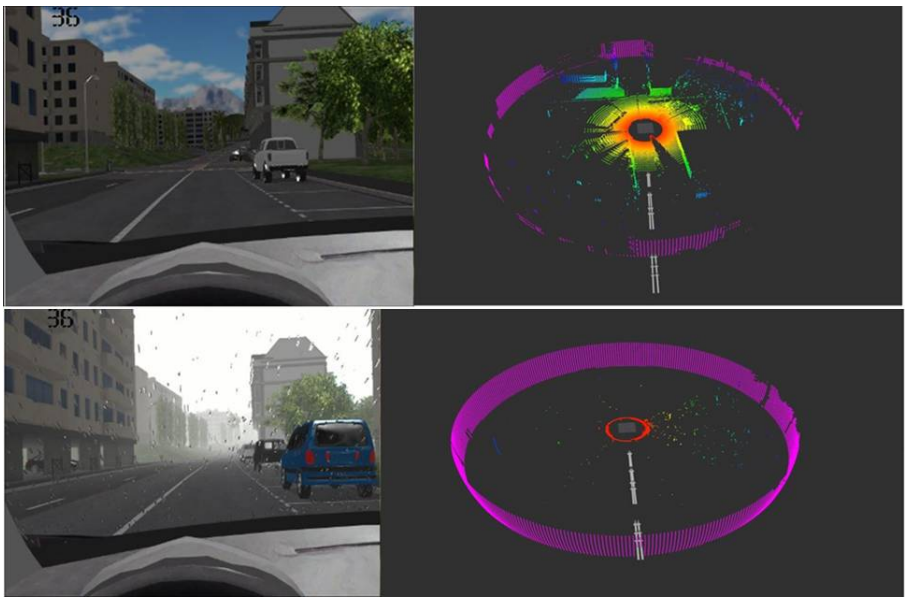

Figure 139: LiDAR modelling with realistic result (source: ESI group)

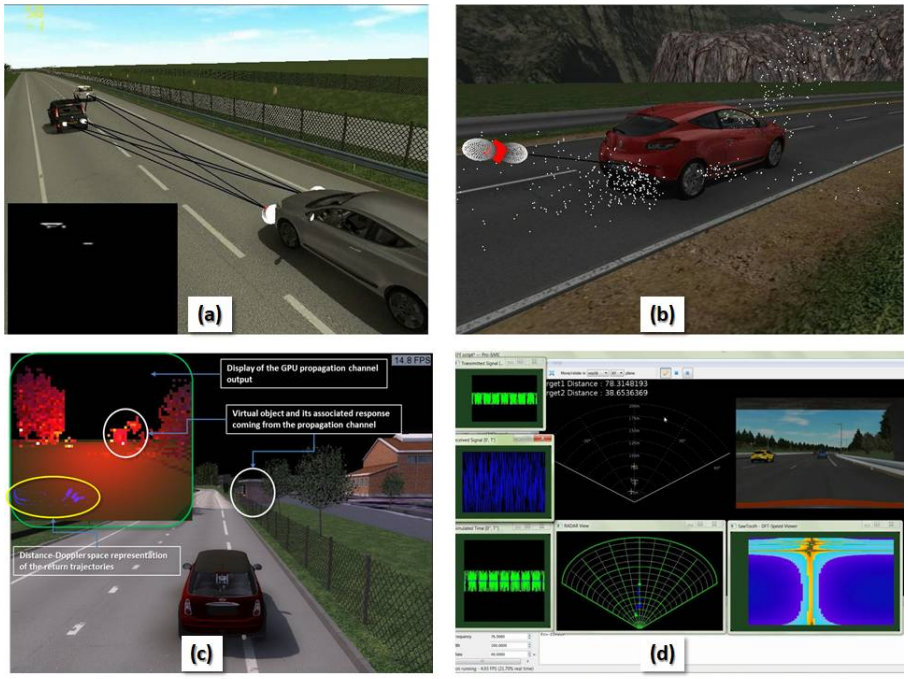

Figure 140: RADAR modelling with the different levels. a: low level with raytracing and ”lobe” model, b: use of a real RCS, b and d: physical based RADAR for realistic model with propagation channel, antenna, and wave generation and processing (source: Univ. Eiffel and ESI group)

Figure 141: Camera by night with headligth (source: ESI group and Univ. Eiffel)

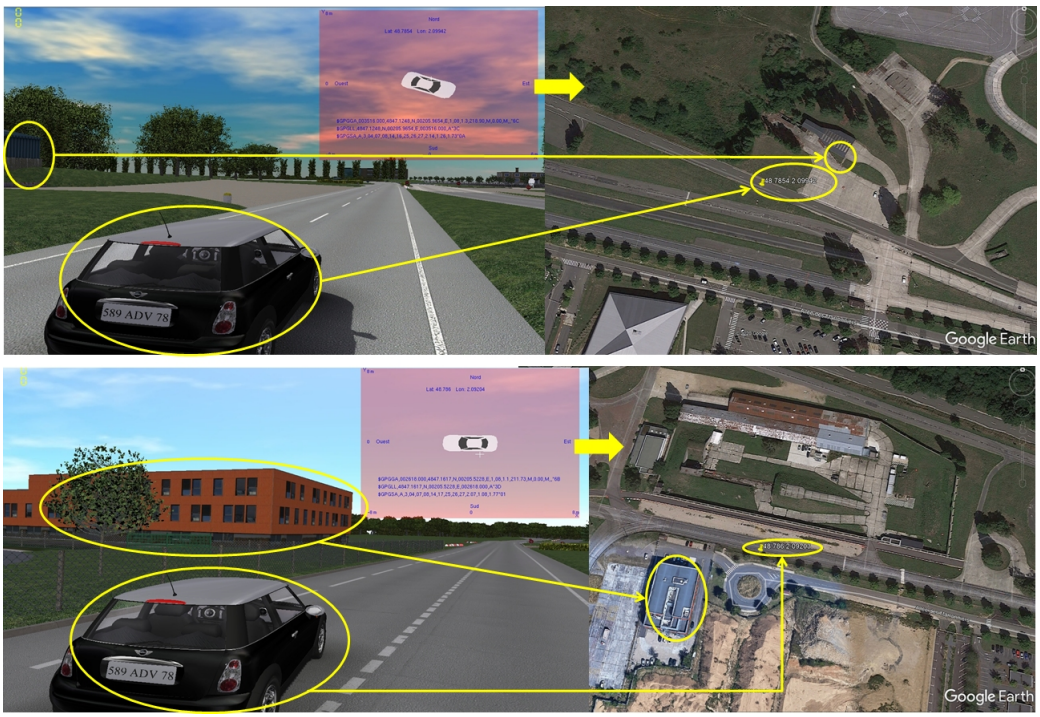

Figure 142: Simulated GPS result on the Satory test track in comparison with real GPS. The simulated NMEA frame positioning is projected in a GoogleEarth map (source: Uni. Eiffel)

5.3.2 Off line or post processing

5.3.2.1 Physics-based Radar model

Thanks to the collaboration with "Oktal-SE Synthetic Environment", a French company of the SOGECLAIR group, SCANeR^{TM} studio embeds physics-based sensors. Such models use a fully ray-tracing engine able to resolve asymptotic formulation of Maxwell equations taking into account physical optics and geometrical optics to simulate the wave propagation. Figure 143 shows more details of the physics-based simulation environment applied to Radar waves, tuned for time efficiency (L2) or improved realism (L3). For the latter case, we calculate the EM wave propagation with ray-tracing, that produces EM fields at the entry of the sensor. These fields can be recorded and used to include the processing (EM field) and compute tools (such as providing range-Doppler maps). In the other hand, the real-time physics model uses an accelerated ray-tracing, followed by a processing stage that delivers Radar data in the form of polar plots.

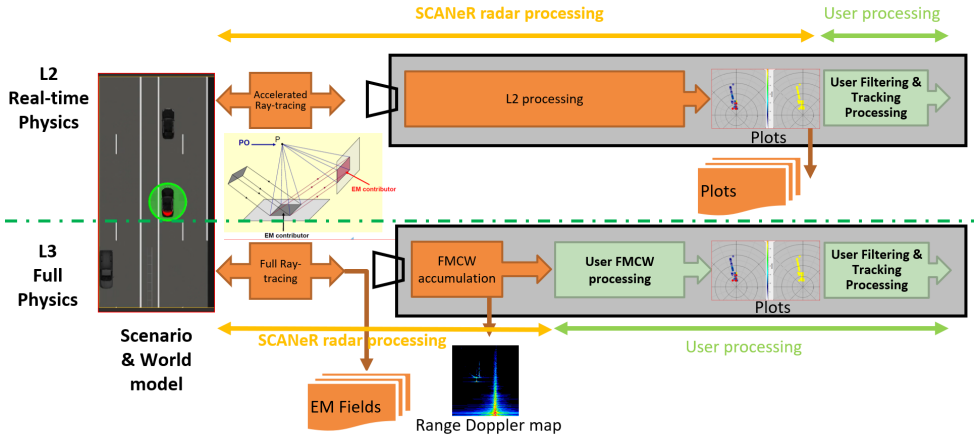

Figure 143: Physics-based Radar

Figure 144 shows an example of an urban scene and the range-Doppler representation output by the Radar model **Level 3**. On this scene, the physics properties of vehicles, metallic barriers, building and road are modelled. Colours on the range-Doppler map are linked to intensity of the EM field received by the Radar antenna. In this map, the vertical line on right-hand side corresponds to static elements in the scene (whose relative speed to the ego-vehicle is proportional to its forward speed) and the remaining signal is linked to rays that hit the target vehicle directly or indirectly (coupling effects of waves bouncing between objects are taken into account).

Figure 144: Simulation scenario and physics-based Radar output data (range-Doppler representation).

5.4 Traffic simulation and microscopic modelling

This section focuses on the models for simulating Automated Vehicles (AV) and human (HV) in traffic simulators, and presents a literature review of the lane-changing models and approaches, as well as the main limitations they have today. A concept for modelling Automated vehicles driving involving both longitudinal and lateral trajectory control is introduced.

The inputs of the microscopic models for the conventional traffic simulation are usually the idealised key measurements of vehicle movements (e.g., speed, difference in speed, time headway, spacing, reaction time), and the outputs are the accelerations for continuous-time car following models and speeds for discrete-time car following models (Treiber and Kesting, 2013 [279]; Laval et al., 2014 [280]; He et al., 2015 [281]). Such simplifications help to capture the major features of vehicle dynamics and traffic flow dynamics. In contrast, for AV driving strategies, it is necessary to investigate each component within the sensing-recognition-

decision-action loop of automated driving. Although the outputs can be simplified as the acceleration/steering values that are similar to microscopic models for HV (Human-driven Vehicle) traffic flow, the inputs are usually raw data from various sensors (e.g., laser point cloud data, image/video data). Such raw data often contains richer information but is much more complex than the one perceived by human drivers. The usual way to simulate Autonomous Vehicles with microscopic traffic model [282] consists in adapting the parameters of HV to reproduce Vehicles driven by an autonomous machine (e.g. reduction of the reaction time to 0.5s and implementation of smoother deceleration). Recently, some intermediate models were proposed to enrich the inputs of the microscopic models for traffic flow so that the obtained analysing results could be shared by both HV traffic flow studies and AV design. The inputs of these intermediate models are kept simple but not too simple to keep an appropriate balance between model complexity and model accuracy. For example, Wang et al. [283] proposed an input presentation method that takes the traffic snapshots during the last few sampling times. In this intermediate models each traffic snapshot is a two-dimensional occupancy grid that reflects the traffic situation around the subject vehicle. Gueriau et al. [284] build a car-following model for AV that generates a "ghost" leader from the simplified information and data collected by the AV at every time-step.

In these HV models, it is necessary to take into account a set of factors relevant to the microscopic models. Nevertheless, some of the presented factors are applicable to human but are not applicable for well-developed AVs (Haiyang Yu 2021 [285]). The factor only usable for UV models are:

- Socio-economic characteristics (e.g., age, gender, income, education, family structure).
- Imperfect driving: For the same condition, drivers may behave differently at different times.
- Driving skills (mainly relative to age and training).
- Distraction level.
- Reaction time of the system.
- Estimation errors: Spacing and speeds can only be estimated with limited accuracy.
- Context sensitivity: Traffic situation may affect driving style.
- Desired speed. For HV, this factor is clearly dependant on skill and Aggressiveness of the driver
- Desired spacing. For HV, this factor is clearly dependant on skill and Aggressiveness of the driver.
- Desired time headway.

About AV, the main factor to consider are the following:

- Reaction time. Much shorter than HVs
- Estimation errors: Spacing and speeds can only be estimated with limited accuracy. Much smaller than HVs or non applicable

- Aggressiveness or risk-taking propensity. Depending on types of AVs
- Perception threshold: Drivers cannot perceive small changes in stimuli. Depending on types of AVs
- Temporal anticipation: Drivers can predict traffic situations for the next few seconds. Depending on types of AVs
- Spatial anticipation: Drivers consider the immediate leading and further vehicles ahead. Depending on types of AVs
- Driving needs. Applicable to AV
- Context sensitivity: Traffic situation may affect driving style.
- Desired speed. AV must respect traffic rule.
- Desired spacing. In AV, this factor must be respected due to risk constraint.
- Desired time headway.

Modelling of longitudinal and lateral movements for mixed traffic suggest to consider the car following paradigm in several ways: First, the vehicle (and the driver) may react differently to their leader [282] depending on the combination of the types of the two vehicles (their own and the leader). Second, the lack of lane discipline in the traffic stream causes drivers to react not only to their leader but also to other vehicles on the other road lanes (lateral aspect). Furthermore, the lack of lane discipline and the variability in vehicle widths result in situations in which vehicles (and by extension drivers) do not strictly follow a leader. For example, vehicles may follow their leader only partially in a staggered way. They may follow two leaders at the same time or be squeezing between two leaders. Finally, in road sections with unseparated bidirectional flow, vehicles may also respond to oncoming traffic sharing the same roadway. It is for this reason it is relevant to address this section with 2 sub parts (longitudinal and lateral models)

5.4.1 Longitudinal models: Car-following models

Longitudinal Movement Models generally describe how a following vehicle reacts to a lead vehicle in the same lane. A large number of car following models have been proposed in the context of homogeneous traffic. These may be classified based on behavioural assumptions. In [286] the Car-following models based on utilised model logic and assumptions are shared in 3 main groups:

- **the top-down group**, starting from the macroscopic scale to achieve microscopic modelling of drivers' behaviour: The main contributor of this group is **the Newell's mode [287]:** The following vehicle is supposed to adopt the same trajectory in space and time as the one adopted by its leader with a delay τ and a gap δ. The spacing defined by (τ, δ) is full dependent on the macroscopic features of the traffic.
- **the bottom-up group**, modelling the microscopic behaviour of the driver, then checking the compliance of the model with macroscopic traffic trends. This group can be refined by:

  - **Gazis-Herman-Rothery (GHR) mode:** The following vehicles' acceleration is proportional to the subjected vehicle (own vehicle), the speed difference between the follower and the leader, and the space headway.
  - **Safety-distance mode:** the follower always keeps a safe distance to the vehicle ahead. Among the existing model, the most famous is probably the Gipps Model. The Gipps model is successful in switching between free flow and following situations.
  - **Psychophysical car-following mode (Wiedemann):** These models use threshold values." Drivers react to vehicles when the set threshold for relative velocity (speed difference between a follower and a lead vehicle) or spacing is reached.

- **the IA-based group**, where the car-following model directly results from the application of **Artificial Intelligence process**: This category of models involves the use of theory like Artificial Potential Field ([288]), or elastic band theory. Some more recent approaches are based on neural network, deep learning, or Reinforcement Learning [289] methods with post learning or real time learning of driving manoeuvres.

Car following models have also been extended to more general acceleration models that also consider free flow situations in which the subject driver does not closely follow a leader.

In traffic simulation, a well known longitudinal model is called Intelligent-Driver Model (IDM) ([290]). This modelling takes into account accelerations and braking deceleration of the drivers. The IDM is a microscopic traffic flow model where each vehicle is seen as an active "particle" in the simulation. Such model characterizes the traffic state at any given time by the positions and speeds of all simulated vehicles. In case of multi-lane traffic, the lane index complements the state description. More specifically, the IDM is a car-following model, so for longitudinal modelling. In such models, the decision of any "*driver*" to accelerate or to brake depends only on its own speed, and on the position and speed of the "leading vehicle" immediately ahead. Lane-changing decisions, however, depend on all neighboring vehicles. The model structure of the IDM can be described as follows:

- **The influencing factors (model input)**: consist on the own speed v, the bumper-to-bumper gap s to the leading vehicle, and the relative speed (speed difference) Delta v of the two vehicles (positive when approaching).
- **The model output**: concern the acceleration dv/dt chosen by the driver for this situation.
- **The model parameters**: describe the driving style, whether the simulated "*driver*" drives slow or fast, careful or reckless, and so on.

In [285], a set of representative car-following models of AV-involved traffic are presented. Considering the factors presented previously, a set of assumptions on car-following models for AV-involved traffic are enumerated:

- AVs aim for optimal performance
- Driving of AVs is complex and may adopt different motion control laws from time to time. However, most existing car-following models only reflect one type of control laws.
- Everything that cannot be well explained by the model is treated as noise/disturbance for the parsimoniousness of the model.

In this study, it appears that the psychophysical-physiological models that directly depict drivers' actions based on the visual angle subtended by the leading vehicle are generally unsuitable for being applied to AVs, as with onboard sensors AVs can accurately measure its gap toward the leading vehicles. Similarly most conventional stimulus-response car-following models (e.g., the GHR model) are not adopted for AV-involved traffic, as the human drivers' reaction described by those models significantly differ from that of AVs.In recent studies, researchers adopted and modified a set of car-following models better designed to describe AV-involved traffic. Among these models more adapted to AV driving modelling, we can mentions desired measure models, safety distance models, optimal velocity models, and ACC/CACC controller models.

- **Desired measure models**: Desired measure models usually assume that vehicles aim to reach both desired speed and desired headway simultaneously. For example, Intelligent Driver Model (IDM) is one of the most popular models that use desired measures ([279]).
- **Safety distance or collision avoidance models**: Different from desired measure models, safety distance models focused on maintaining sufficient spacing to the leading vehicle rather than the relative speed. As a result, safety distance models usually reserve a relatively large gap for AVs compared to other types of car-following models for AVs. [291] proposed a new concept of Responsibility-Sensitive Safety (RSS) to derive the collision avoidance condition for AVs.
- **Optimal velocity models**: The first optimal velocity (OV) model was proposed in Bando et al. [292] to reveal the link between microscopic driving behaviours and macroscopic traffic flow measures. The OV models assume that each vehicle has an OV that is dependent on the gap from the leading vehicle. The original OV model in Bando et al. [292] characterised the acceleration of subject i as the scaling difference between the actual velocity $vi(t)$ and the OV that is written as follows. The new OV models dedicated for AVs still hold the assumption about the sensitivity coefficient but usually determine the OV based on the information of gaps among a few consecutive vehicles with the aid of V2X communication. By appropriately choosing a car-following model function, it is possible to fully employ the additional position and speed information of other vehicles to improve local traffic flow stability and smooth shock waves. For [293] the Intelligent Driver Model (IDM) also belongs to this category of Optimal Velocity Models.
- **ACC/CACC controller models**: To address the string stability of traffic flow, a variety of controllers for AVs were proposed by the researchers in vehicle studies to keep AVs running at an ideal speed regarding their leading vehicles in a vehicular platoon ([294], [295]). Field tests show that such new models can guarantee the string stability of AV platoons.

5.4.2 Lane changing models

Dynamic models involve first the longitudinal models. There are many literature review done on those models which are well-known approaches. Toledo, 2007 [296] and Olstam and Tapani, 2004 [297] are valuable surveys on car-following models among them, and Saifuzzaman et al., 2014 [37] describes the human factors to model car-following. But longitudinal manoeuvres are not sufficient and it gets relevant to extend the reachable domain to the lateral manoeuvres in a second time. Indeed, in given situations like critical situations with crash, the lateral manoeuvres offer solutions. This paper focuses on the lateral models (lane-changing) which are

more challenging. In a third time, the longitudinal and lateral models must be coupled to make a full dynamic model.

This literature review is based on the references of three important papers surveying the existing lane-changing models. The oldest Moridpour et al., 2010 [298] provides a critical review of the lane-changing models with a classification of the decision making approaches such as described in Fig 2. Those are divided in two branches: the driving assistance models and the driving decision models.

A few years later, Rahman et al.[36] use and complete this review by proposing a new classification of the lane-changing models including the most recent studies, shown in Figure **??**. The paper opposes the microscopic models to the macroscopic models and the hybrid models (mix of microscopic and macroscopic models). It focuses on the lane changing microscopic models, which are divided into four general categories:

- Rule-based models
- Discrete choice-based models
- Artificial Intelligence models
- Incentive-based models

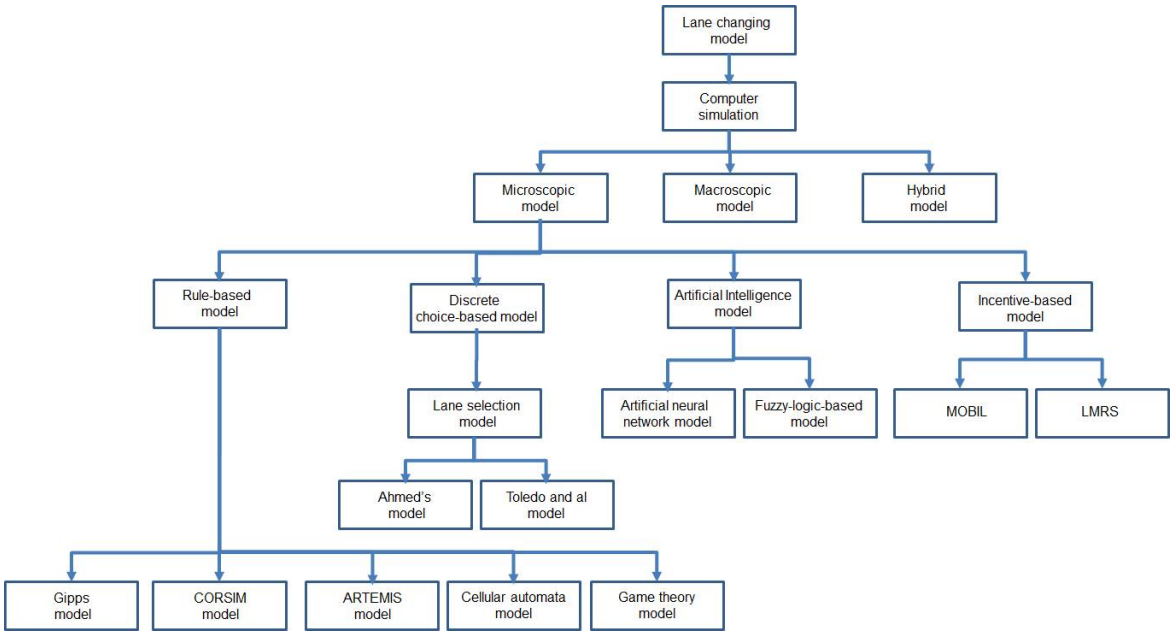

Figure 145: Classification of lane-changing models [Rahman et al., 2013][36]

Both surveys present the existing limits of lane-changing models which are:

- Lack or absence of dynamic description for the lateral trajectory of the lane-changing manoeuvre,
- Necessity to get a large amount of data on actual lane changing trajectory to calibrate the models' parameters and to validate them.

Later, the literature review of Zheng, 2014 [299], on the recent developments and research needs in modelling lane changing completes the work of the two aforementioned surveys.It draws a line between two categories of lane changing models: those which describe the lane-changing decision making process, and those which quantify the impact of lane changing behaviours on the road driving environment. In this study, the research need to develop a lane-changing model capturing decision-making and impact on the surrounding traffic rises. In addition, there is the necessity to develop a model providing lane-changing trajectory accurately close with low margin of error to the actual trajectories. Evaluating the impact on the mixed traffic being our goal, it appears an improved lane-changing model is required for a multi-criteria evaluation. But, we have to keep in mind, that usually there are two steps in planning lane-changing trajectories for AVs. The first step is to consider what kind of lane-changing trajectories are allowable, and then select a suitable family of curves (shape) for planning lane-changing trajectories.

5.4.2.1 Lane changing models for human behaviours modelling

The referenced lane-changing models listed in the aforementioned studies are detailed and regrouped in the following categories. Most recent papers published after Zheng, 2014 [299] are also added to the existing list of models:

- Psychological and cognitive: Espié et al., 1994, Champion et al., 2001, Champion et al., 2002; Bornard, 2012 [300]; Bellet et al., 2018 [301];
- Rule-based: CORSIM: Halati et al., 1997; Wei et al., 2000; MOBIL: Treiber and Helbing 2002; ARTEMIS: Hidas and Behbahanizadeg, 1999; Hidas, 2002; 2005; Jesting et al., 2007;
- Cellular automaton: Nagatni., 1993, 1994; Nagel et al., 1998; Rickert et al., 1996; Wagner et al., 1997; Wolfram, 1983; Fukui and Ishibashi, 1996 (stochastic TCA); Nagel and Schrekenberg, 1992 (TCA); Nagel and Paczuski, 1995; Takayasu and Takayasu, 1993 (slow to start TCA); Borlovic et al., 1998; Borlovic, 2003 (Velocity dependant randomization TCA)
- Fuzzy logic based: Ma, 2004; Mendel, 1995; McDonald et al., 1997; Brackstone and McDonald, 2000; Wu et al., 2003; Das and Bowles, 1999 (AASIM); Das et al., 1999; Moridpour et al, 2012
- Game Theory: Kita, 1999, Kita, 1993; Pei and Xu, 2006
- LC Incentive: Laval and Daganzo, 2006; Laval and Leclerq, 2008; Duret et al., 2011; Zheng et al., 2013;
- Probabilistic: Worrall et al., 1970; Pentland and Liu., 1999; MITSIM: Yang and Koutsopoulos 1996; Zhang et al., 1998; Amhed, 1996, 1999; Sheu and Ritchie, 2001; Singh and Li, 2012; Toledo and Katz, 2009 (Hidden Markov Model); Wang et al., 2019 [302]
- Data-driven deep learning: Li et al., 2015 [303]; Xie et al., 2019 [304]
- LC implementation process: Enke 1979; Nelson 1989;Chovan et al 1994, Shamir, 2004; Yao et al., 2013 (data driven LC Implementation); Ding et al., 2013 (BP Neural Network model)

5.4.2.2 Lane changing models for automated vehicles modelling

There are very few lane-changing models which can be applied on purpose to automated vehicles. Indeed, the conception of lane-changing models are based on human behaviours to capture their decision making process, and therefore does not fit a non-human decision maker. However,some models can be applied indistinctly to human and nonhuman driving behaviours:

- **Fuzzy interference system based model**: Balal et al.,2016 [305]
- **Lane-changing implementation process**: Enke 1979; Nelson 1989; Chovan et al., 1994, Shamir, 2004; Yao et al., 2013; Ding et al., 2013

While those models are said to be fit for Automated Vehicles modelling, they actually present no specificities to this kind of driving for they are human-related. In the prospect of our research, we selected the model developed by Vanholme et al. [306] which fits best for Automated Vehicles with both integrated car-following and lane-changing models with in addition a comprehensive implementation process.

5.4.2.3 Literature review: a short synthesis

The three surveys of Moridpour et al., 2010 [298]; Rahman et al., 2013 [36] and Zheng, 2014 [299] provide interesting classification on the currently existing lane-changing models,mainly to capture human-driving behaviours. In addition of those, a few more papers published after the later complete the review. The classification of lane-changing models presents various strengths and weaknesses. The most important limitation of those models for our research needs is an inaccurate description of the lateral trajectories during the lane-changing process. In addition, the decision making process is rather discrete and irreversible when it should be continuous and reversible. All three surveys accord themselves on the need of large set of data to calibrate the parameters and validate the models based on closeness to actual lane-changing trajectories.

An important remark concern the limit of usual tracking models such as IDM or Gipps. The main drawback is probably their "collision-free" component ($TTC \geq threshold$) which prevents an impact in terms of risk from being measured.

An interesting synthesis about advantages, weaknesses, and human factor consideration have been proposed in [37] and a synthesis is given in the figure 146.

A synthesis also is provided but for driving models involving human factors (see figure 147).

5.4.3 Micro-Simulation calibration and validation

Micro-Simulation calibration processes aim at finding the parameters of the simulation that will minimize the difference between observed data and modelled data. This process, mathematically introduced in [307] can be assimilated to an optimisation process. It necessitates heavy computation capacities and only certifies the validity of the simulation for the specific use case of the observed data. The complete calibration process, apart from the simulator itself, requires at least 4 main elements:

- The **parameters** that will be modified to obtain the best possible performance (i.e. which minimises the discrepancies between an observed database and a modelled one). In [308], Maheshwary et al. have shown that this calibration is generally performed with a number of parameters varying between 2 and 15 depending on the objective of the simulation.

Model category	Model name	Strenghts	Weaknesses	Humans factor's aspects
GHR model and extensions	**Linear CF model**	Simplest model. The stability of the model is proved. Several functional form ofkis found. however the authors used λ=a as a constant.	The model is too simple to describe actual traffic phenomena as the later models do.	Driver reaction time
	Non linear GHR model	Simple and well-established model. Most studied model. Model parameters can be easily estimated from either vehicle trajectory data or macroscopic data (using speed-density relationship). Many estimations of the parameters are available.	Use of identical reaction time for all drivers does not capture inter-driver heterogeneity. Human ability to perceive small changes in driving conditions are overestimated. Model parameters do not consider behavioral differences between acceleration and deceleration. The model is highly sensitive to velocity difference. When velocity difference is zero, any value of spacing is acceptable.	Driver reaction time
	Lee	Introduces memory function in the linear CF model. The memory function makes the driver decisions consistent with the past driving profile, rather than creating instantaneous accelerations. Removes unrealistic peaks in acceleration profile of the driver.	The implementation of the model in traffic simulation is considerably more complex due to the need of maintaining an array of past conditions for each vehicle.	Driver reaction time. Memory function to consider past driving experience
	Ahmed	The model considers acceleration/deceleration asymmetry. Two separate models are proposed for free flow and CF, separated by a headway threshold. Traffic density is considered within 100 m in front of the driver. Driver heterogeneity is considered by distributing the reaction time over the driver population.	Transition from free flow to CF and vice versa are not smooth. Human ability to perceive small changes in driving conditions are overestimated.	Distribution of Driver reaction time to consider driver heterogeneity. Acceleration/Deceleration asymmetry. Driving condition in terms of traffic density
	Herman and Rothery	The model considers multi-vehicle interactions where driver follows more than one preceding vehicle.	Linear CF model is used as a base model which has already been criticized for its simplicity.	Driver reaction time. Multi-vehicle interaction
Desired measures models	**Helly's model**	The desired space headway creates a safety buffer which prevents collision. The desired space headway is dependent on speed and acceleration of the subject vehicle.	Direct dependency on following distance might create unrealistically high acceleration/deceleration. Model parameters for desired space headway are difficult to estimate as it is unobservable in usual traffic data. Desired space headway is not driver dependent	Driver reaction time Desired space headway
	Koshi et al. (1992)and Xing (1995)	Direct dependency on following distance is solved. The model has four stages: standard driving, acceleration from standing queue, effect of gradient and free flow acceleration.	While the physical condition of the road in terms of gradient is considered in this model, horizontal curvature effect is neglected. No estimation effort is found for this model.	Driver reaction time Desired space headway Desired speed
	IDM (Treiber et al., 2000) IDMM (Treiber and Helbing, 2003) HDM (Treiber et al., 2006)	This model considers both the desired speed and the desired space headway. The desired space headway depends on speed, speed difference, minimum spacing, maximum acceleration, comfortable deceleration and desired time headway. The model considers vehicle capacity. Several attempts are found for calibration of the model. The model is a combination of free-flow and CF model. The transition between free-flow and CF model is smooth.	Reaction time is ignored in IDM. Two extensions of IDM: IDMM and HDM are available. IDMM considers driver's adaptation capability with surrounding environment to improve desired time headway calculation. In HDM four extensions to IDM is proposed: finite reaction times, estimation errors, spatial anticipation, and temporal anticipation.	**IDM and IDMM** Desired space headway Desired speed Comfortable deceleration Desired time headway **HDM** Driver reaction time TTC estimation error Multi-vehicle interaction Relative distance error
Safety distance models	**Kometani and Sasaki**	The model seeks to specify a safe following distance. The model parameters are estimated. The parameter d prevents the model from collision.	Does not describe stimulus–response type function as most of the other models.	Driver reaction time
	Newell	It is a non-linear model and follows stimulus–response type function. The model use a safety distance which prevents collision Has separate model for free-flow and CF	Direct dependency on density might result in unrealistic accelerations or decelerations.	Driver reaction time
	Gipps	The transition between free-flow and CF is smooth unless the preceding vehicle brakes harder than anticipated, preceding vehicle changes the lane or a new vehicle enters in front of the subject vehicle from adjacent lane. The model offers distribution of behavioral parameters to offer driver heterogeneity. (Vehicle size + safety distance) follows a normal distribution.	Although many behavioral parameters used, the model does not consider driver errors. How the parameter values are estimated/selected is not explained	Driver reaction time Desired acceleration Desired deceleration Estimation of preceding vehicle's deceleration Desired speed
Optimal velocity model	**OV model (Bando et al., 1995) (Bando et al. (1998)**	The optimal velocity (OV) depends on the distance from the preceding vehicle. The model was estimated byHelbing and Tilch (1998).	Driver reaction time is ignored in this version but later introduced inBando et al. (1998). The model produces unrealistic acceleration and decelerations. The model is unrealistically sensitive to reaction time	Driver reaction time
	GF model (Helbing and Tilch (1998)	Heaviside function (H) works at negative velocity difference and solves the problem of unrealistic deceleration that occurs in OV model. Model parameters are estimated from real data.	The model still produces unrealistic accelerations. Driver reaction time is ignored	Non Adapted
	FVD model (Jiang et al., 2001)	Velocity difference is included explicitly to overcome unrealistic accelerations/decelerations The model uses different responses in acceleration and deceleration as an improvement over FVD model.	Having a single parameter for both acceleration and deceleration might lead to an unrealistic situation where the subject vehicle brakes insufficiently, even if the distance to the preceding vehicle is extremely short. Driver reaction time is ignored No estimation of the model parameters are found, neither the model is applied on real data	Non Adapted
	AFVD model (Gong et al., 2008)		AFVD model takes longer time then FVD model to become stable. Numerical simulation is done but the model is not yet applied on real data and the parameters are not estimated.	Non Adapted
	Lenz and al	The model considers multi-vehicle interactions which increases the model's stability.	Driver reaction time is ignored Numerical stability analysis is performed, but the model parameters are not estimated from real data	Non Adapted
	Davis (2003)	The OV function is extended to consider the change in velocity difference as well as headway. Unrealistic sensitivity to reaction time is solved.	Although the OV function is measured at time (t-τn), the velocity is measured at timet, which needs a behavioral justification.	Driver reaction time
Newell's sipmplified CF model	**Newell (2002)**	It is a parsimonious model (only two parameters are required). The model can express macroscopic flow theory very well. The model has been extended to capture traffic oscillation.	The model is purely based on traffic flow theory. No driver behavior parameters are available.	Non Adapted
Cellular Automata (CA) models	**Nagel and Schreckenberg. Klauss and al.**	Randomness in speed is implemented to accommodate deceleration noise The discrete version byNagel and Schreckenberg (1992) was difficult to calibrate with real data, hence the continuous version is proposed by Krauss et al. (1996)as shown in the equation presented in this table.	Unrealistic deceleration is observed at high densities. Driver reaction time is ignored. Although the discreteness of the model does not correspond directly to any property of real traffic, the model shows nontrivial and realistic behavior of traffic flow at the macroscopic level.	Non Adapted
	S-K model (Krauss and Wagner)	Unrealistic deceleration problem of Krauss et al.'s (1996) model is solved by replacing S_{gap} with V_{safe}. V_{safe} is calculated based on maximum allowable deceleration. S-K model outputs more realistic traffic characteristics at the macroscopic level than Krauss et al.'s (1996).	Driver reaction time is ignored	Non Adapted

Figure 146: Comparison of the Car Following methods with strengths and weaknesses ([37])

Model category	Model name	Strenghts	Weaknesses	Humans factors included
Use of perceptual thresholds	Wiedemann. Fritzsche	Perceptual thresholds selects minimum value of the stimulus a driver can perceive and will react to. The thresholds are expressed as a function of speed difference and relative spacing. They are different for acceleration and deceleration decisions. The thresholds divides the driving plane to several decision zones, such as 'no reaction zone' (free-flow), 'closing in', 'danger' (must decelerate), and 'car-following'.	The thresholds are simply obtained from the human factors literature. The equations for different thresh-olds are undisclosed.	Perceptual thresholds
Driving by visual angle	DVA model (Andersen and Sauer, 2007)	Human driver are capable of accurately estimating time to collision (TTC) based on visual angles subtended by the preceding vehicle. Visual angle is used to replace relative spacing from the preceding vehicle, and angular velocity is used to replace relative velocity. DVA model produces similar speed and acceleration profiles, as observed from the actual driving situation.	Driver reaction time is ignored. A constant value for desired velocity is used for simplicity; thus, this model ignores driver heterogeneity.	Visual angle Angular velocity
	Jin et al. (2011)	It is the FVD model using visual angle	No estimation is done for this model; the authors used same parameter values as used for FVD model. Driver reaction time is ignore	Visual angle Angular velocity
Use of prospect theory to model risk-taking behavior	Hamdar et al. (2008)	The subjective probability of being involved in a rear-end collision depends on acceleration, spacing and speed difference. The gains (or losses) in this model are expressed in terms of increase (or decrease) in speed from the previous acceleration instance. Final acceleration is retrieved from a probability density function to reflect stochasticity in driver's response. The model allows risk-taking maneuvers when drivers are uncertain of the leader's future behavior and, consequently, crashes are possible.	The probabilistic nature may create more acceleration noise than other models such as GHR, Gipps and IDM	Risk-taking behavior Maximum desired speed Anticipation time Uncertainties of the preceding vehicle's speed Uncertainties of the spacing Random components of the subjective utility function Reaction time (not explicitly included)
CF models which consider driver error and distraction	Van Winsum (1999)	The model assumes that the driving conditions and mental effort can make substantial difference in desired time headway. The desired time headway can be influenced by visual conditions (such as fog, rain and night driving), driver state (such as fatigue and inebriation), and the mental effort deployed in following the preceding vehicle.	The model use preceding vehicle's deceleration as a parameter which is really difficult to measure accurately for a human driver. Rather, Gipps (1981)uses an estimate to preceding vehicle's deceleration. The model parameters are not estimated The model is not yet tested with real data	Desired time headway Driver reaction time Driving condition
	Yang and Peng (2010)	It is a stochastic CF model. Driver error mechanism is developed from a large scale naturalistic driving database Three major types of driver errors are introduced: perceptual limitation, time delay, and distraction	The effect of distraction is hypothesized as no change in driving condition during distracted period. No real experiment is done to prove the assumption. Homogenous driving population used	Driver reaction time Driver distraction Perceptual thresholds Stochastic error behavior

Figure 147: Comparison of the Car Following methods involving human factors with strengths and weaknesses ([37])

Those parameters are the key of the whole process and are generally selected by experts or relying on basic statistical tests such as ANOVA.

- A **cost function** that expresses the difference between the observed and the modelled data. This cost function can be multidimensional. If travel time if often used as a cost function, for the sake of microscopic behaviour validity, other Measures Of Effectiveness (MOE) can be used such as the desired speed, acceleration and deceleration in the works of Li et al. ([309]). To evaluate the validity of the micro-simulation in terms of safety, the speed can be used but Cunto et al. ([310]) proposed a more specific Crash Potential Index (CPI) which relies on the vehicles speed differences, interdistance and deceleration capacities. As many vehicles are modelled in the simulation, these cost functions are represented as statistical distributions. Thus, the final MOE used are statistical indicator such as the mean difference, Root Mean Square Error (RMSE) or other errors indicators.

- A **database containing all in-situ observations**.

- An **optimisation procedure**. In a majority of research works, evolutionary algorithm are used for the calibration process as they can deal with several parameters but also with multidimensional cost function. In their paper from 2017, Yu et et al. ([311]) proposed a comparison and a combination of a genetic algorithm and a tabu search algorithm and also provided a list of methods used previously on different kind of simulators. In this class of genetic algorithm, Huang et al. used a specific algorithm called NSGA2 as Non Sorting Genetic Algorithm that is famous for performing multi objective optimisation.

5.4.4 Micro-Simulation Modelling Tools for Human-Driven and Autonomous Vehicles involving communication

A number of traffic micro-simulation packages are available, proposing either open source or commercial versions. They can simulate various real-world network configurations, problems, and solutions. Each computer simulation software package employs different car-following behaviours, lane-change, and gap-acceptance models, as discussed in the previous section. The most popular microscopic simulation tools for traffic generation are:

- **Advanced Interactive Microscopic Simulator for Urban and Non-urban Network (AIMSUN)**: The AIMSUN developed by Traffic Simulation System located in Barcelona (Spain) uses the Gipps' safety distance models. This tool is well-known for modelling traffic dynamic assignment, incident management, and ITS applications such as ramp metering and vehicle guidance systems.

- **CORridor SIMulation (CORSIM)**: The CORSIM software was developed in 1988 by the US Federal Highway Administration (FHWA) by combining several previous micro-simulation tools such as the NETSIM, which can simulate urban traffic streams and the FRESIM (FREEway SIMulation). FRESIM uses the Pitt car-following model that can be considered as a stimulus-response model.

- **PARAMICS**: The PARAMICS tool was developed by the UK Department for Transportation (DfT) in 1990. This software uses the Fritzsche car-following model that is an acceleration model based on psychophysical logic.

- **Verkehr In Städten –SIMulationsmodell (VISSIM)**: The PTV VISSIM is one of the most predominant micro-simulation tools developed by a German company PTV Vision (Planung Transport Verkehr) in 1992. The VISSIM uses a psycho-physical car-following model which was first developed by Wiedemann in 1974 and further enhanced in 1992 by Wiedemann and Reiter. VISSIM is a well maintained commercial software providing a user friendly interface with also 3D visualisation.

- **MovSIM**: MovSIM is an interactive Java-based open-source traffic simulator developed by Treiber and Kesting [312], available at www.traffic-simulation.de. It involves IDM for car-following aspect and MOBIL model for lane-changing manoeuvres. The platform has been extended to exhaustively include the modelling of Connected and Autonomous vehicles with simplification of the perception for AVs [284].

- **SUMO**: SUMO is another freely available tool published under the Eclipse Public License V2 ([313]). SUMO is used worldwide and is downloaded over 35.000 times every year. It is a reference for open source traffic simulator. SUMO Offers the opportunity to implement various car-following and lane-changing model, including Krauss, Gipps, IDM... Moreover in SUMO, the travel space is managed with continuous modelling (euclidian space).

- **SymuVia**: It is an open source platform developed by the Gustave Eiffel University. It provides large scale simulation and capabilities to simulate V2X systems (extension SymuCAT). SymuVia can implement several models, but is natively built with the Newell's model to reproduce car-following behaviours. It ensures the compliance between macroscopic traffic flow theory and microscopic behaviours.

- **MATSIM**: The activity-based traffic simulation MATSim is an open source tool, that allows to take into consideration changes in the mobility behaviour (Origin-Destination matrices) according to variations into the supply (e.g. introduction of Autonomous mobility).
- **ParamGrid**: ParamGrid is a modelling framework, developed by Klefstad et. al. [314], with the purpose of performing microscopic simulations for large-scale networks. Especially, it aims at tackling the issues related to computing complexity of large scale road networks. It divides the road network into tiles and manages the content of each tile including the transfer of vehicles from one tile to another.
- **SMART**: Scalable Microscopic Adaptive Road Traffic Simulator ([315]).
- **Paramics**: Simulation of congested traffic networks at the level of individual vehicles.
- **INTEGRATION**: It allows a large number of vehicles to be simulated at the same time while maintaining control over time management during a simulation of the order of 100 ms. This allows a more detailed analysis of acceleration, change of lanes, etc.
- **TransModeler**: TransModeler can import simulation data from Corsim and SimTraffic. The Gipps' model is the default car following model in TransModeler.
- **TRANSIMS**: TRANSIMS is a microscopic simulator based on a cellular automaton (traffic space shared with cells). Each vehicle is placed in a cell, representing a portion of the road. At each time step, it advances a certain number of cells, depending on its maximum speed. Since a cell can only be occupied by one vehicle, if the cell in front of him is occupied, he will have to wait for it to become free.
- **ARCHISIM**: ARCHISIM uses a following model, but it has the particularity of coupling the traffic simulation to a driving simulator. Thus, human users will have to interact with the software agents. This makes it possible to introduce realistic human behaviour into the simulation.

All these traffic simulation platforms are widely used to attempt to reproduce or study the impact of C-ITS services in terms of traffic efficiency or impact on mobility. In [315], the authors propose a set of criteria in order to evaluate 3 traffic simulators platform with SMART. The result of this study is presented in figure 148.

In [316], the authors have used the Gipps and IDM car-following models with a re-calibration stage using data from a simple car-following scenario within a driving simulator to study drivers' performance while engaging in distracting activities: texting, talking on the phone, or eating, and a control scenario with no distracting activity.

In the MovSIM platform, Treiber proposes solutions in order to take into account intuitive parameters corresponding to nearly orthogonal aspects of the driving style (adaptive driving behaviour):

- The agility of the drivers is positively correlated to the acceleration parameter a,
- the degree of anticipation is negatively correlated to the comfortable braking deceleration
- the following distance (aggressiveness if too close) is determined by the desired time headway T .

Figure 148: Comparison of Key Features of Traffic Simulators

This platform also offers a Car-to-Car and Car-to-Infrastructure communication modelling. About this aspect of V2X communication modelling, a set of projects also propose solutions. Among the more efficient and useful solutions, we can quote:

- **iTETRIS**: Proposes to interconnect SUMO and NS-3 and to add a scheduler called iCS (iTETRIS Control System). NS-3 is a well known library of communication modelling and simulation. This platform is open-source. iTETRIS uses TraCI as the communication interface which adopts a very similar command-response approach and a TCP connection.
- **SiVIC MobiCoop**: Propose to interconnect Pro-SiVIC, NS-3, and RTMaps with a multi-computers and multi-OS communication bus (DDS and DDsL). This platform has been developed by Gustave Eiffel University during the projects CooPerCom and SINETIC.
- **VNetInSim**: A dedicated simulation platform for "Vehicular Ad Hoc Network" (VANET). This platform is uses the OPNET communication simulator and the INTEGRATION traffic simulator.
- **VEINS**: Combines the functionalities of the SUMO traffic simulator and the OMNET ++ network simulator. This platform is open-source. It is based on models implemented in network and traffic simulators in order to simulate VANETs as realistic as possible. VEINS uses TraCI as the communication interface which adopts. ARTERY is an extension of VEINS that is built to reproduce the European telecommunication protocols.
- **TraNS**: TraNS federates the traffic simulator SUMO and the networking simulator NS-2. TraNS uses TraCI as the communication interface.
- **EPiCAM**: EPiCAM is a platform developed by Gustave Eiffel University which allows to interconnect SymuVia and Pro-SIVIC MoboCoop (NS-3, Pro-SiVIC, RTMaps, DDsL).

In [38],the authors propose to designs a comprehensive simulation platform for conventional, connected and automated driving from a transportation cyber-physical system perspective, which tightly combines the core components of V2X communication, traffic networks, and autonomous/conventional vehicle model. Specifically, three popular open-source simulators SUMO, Omnet++, and Webots are integrated and connected via the traffic control interface, and the whole simulation platform is deployed in a Client/Server model. Two use cases are demonstrated with this platform, a traffic flow optimization and a vehicle eco-driving system. The proposed platform provides a test-bed to explore issues like social/economic impact of connected and automated driving from the individual level to the large-scale network level. From figure 150 we can observe that this new architecture is clearly very close to the ones proposed by AV Simulation (SCANeR Studio) and Univ. Eiffel/ESI group (Pro-SiVIC MobiCoop and EPiCAM (see figure 124)).

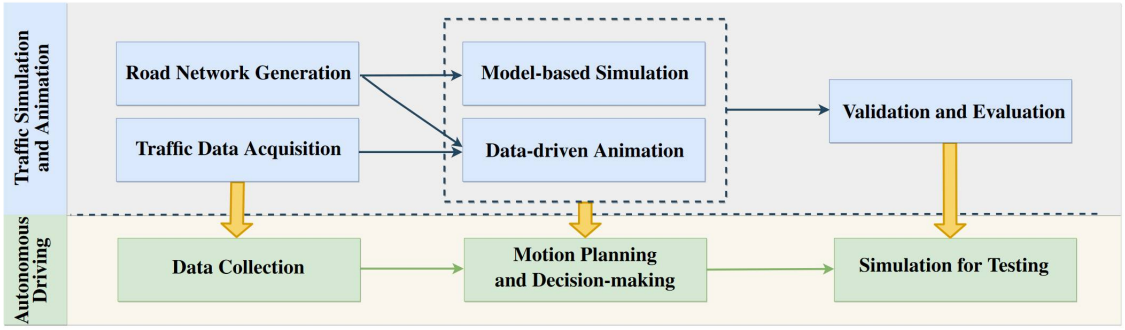

Figure 149: Schema of traffic simulation and animation components

In [39], The CoSAM framework (CoSAM) is proposed and provides several functionalities to enable co-simulation Autoware and MATLAB/Simulink. Autoware, based on Robot Operating System (ROS), is a popular open-source software project developed for the autonomous vehicles. Autoware can be used in embedded systems, such as NVIDIA DRIVE PX2 and Kalray MPPA256. Autoware is composed of a Localisation module, Detection model, Prediction module, Mission module, Motion module, and Actuation module, and cannot only operate the autonomous vehicle but also simulate with actual data (which is rosbag data).

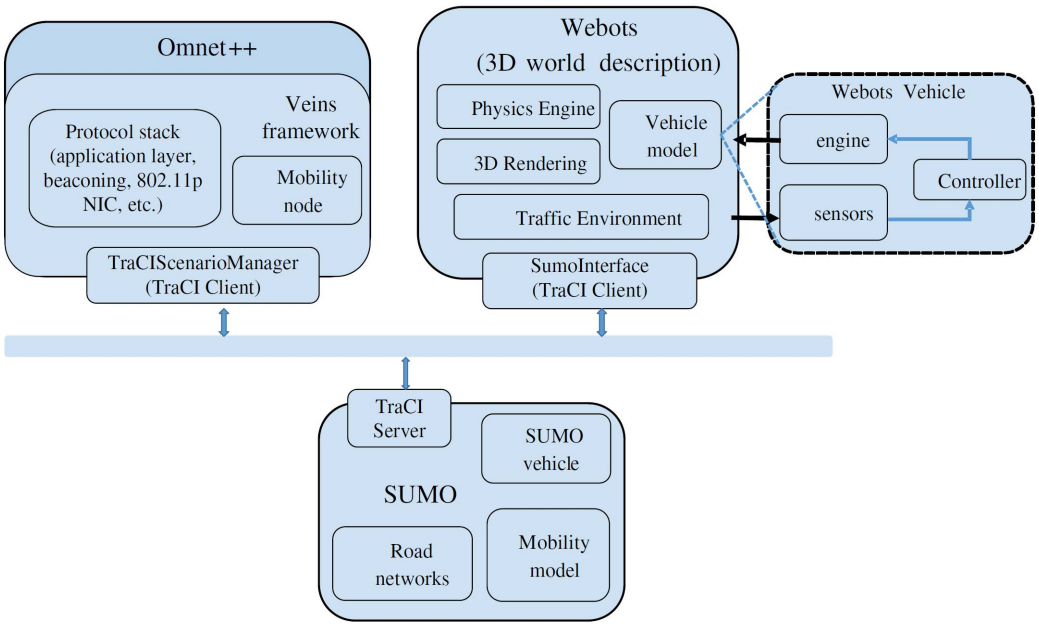

Figure 150: Communication interface among components for integrated simulation platform proposed by [38]

Traffic Fidelity Measure: In [317], Chao et al. provide a Survey on Visual Traffic Simulation involving the models, the evaluations, and the applications of these tools in Autonomous Driving. In this survey, Chao and al proposes a general schema of traffic simulation (see figure 149) and the different class of models (see figure 152).This paper is interesting because it proposes a validation and evaluation process with the concept of fidelity measure. In this study, fidelity measure comparisons among three virtual traffic flows generated by the IDM model is done using three different parameter sets. The initial traffic states of the simulator were set to the same values as the real-world traffic flow. Differences between the simulated traffic and

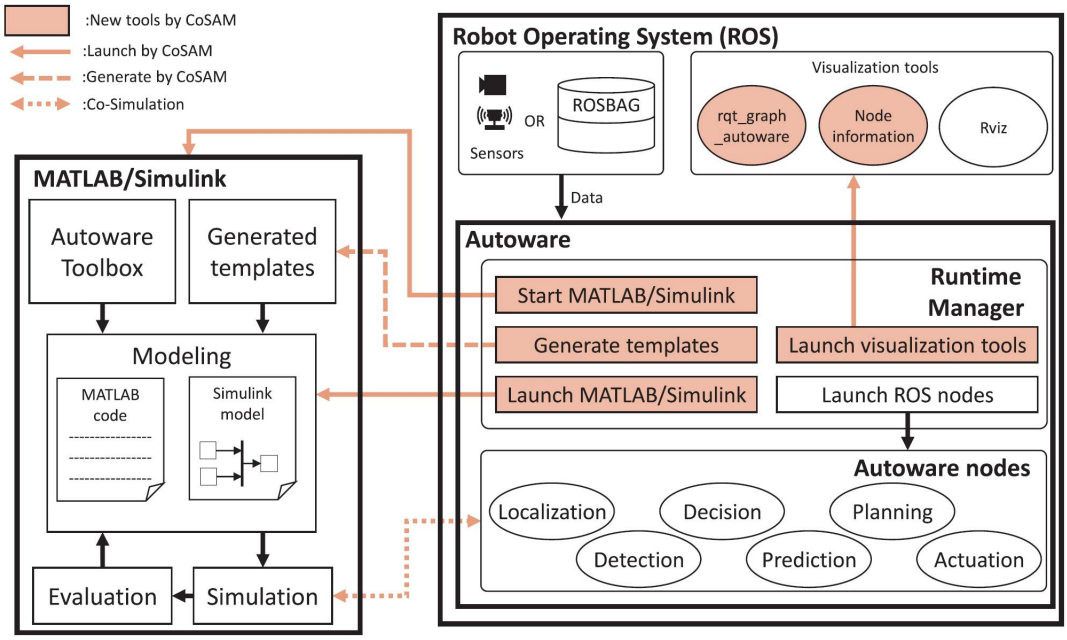

Figure 151: System model of CoSAM ([39])

real-world ground truth are highlighted. For the dictionary-based fidelity evaluation, a smaller value of the metric indicates a higher fidelity of virtual traffic ([40]).

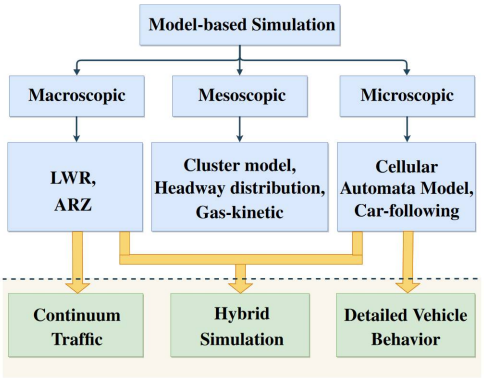

Figure 152: Classification of model-based traffic simulation methods based on the levels of detail

These simulation platforms use for input a modelling of the road networks with all their features. The main road network modeling and specification are:

- OpenDrive
- RoadXML
- Open Street Maps
- IPG format

In [41], a city-scale traffic animation using statistical learning and metamodel-based optimisation is proposed (see figure 154 and figure 155. Li et al. proposed an efficient algorithm to

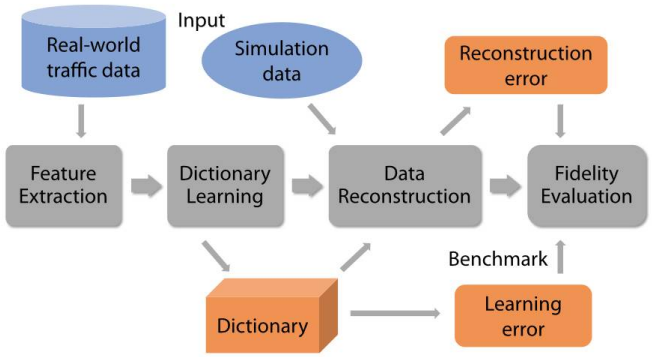

Figure 153: The pipeline of the traffic fidelity measure proposed by [40]. The blue boxes show the input of the system, which contains real-world traffic data-set and simulation data to be evaluated.

reconstruct city-scale traffic from GPS data using statistical learning. To address these issues with incomplete and/or sparse data, a metamodel-based simulation optimisation is proposed to dynamically bridge the "gap" between the reconstructed traffic learned from GPS data and the simulated traffic where the data is incomplete or missing. This approach is able to perform visualisation of city-scale traffic, as well as datadriven 2D and 3D traffic animation in a virtual environment.

Figure 154: 2D animation (left) and visualisation (centre) of reconstructed traffic in virtual San Francisco with 3D vehicle flows (right) using the method presented in [41]

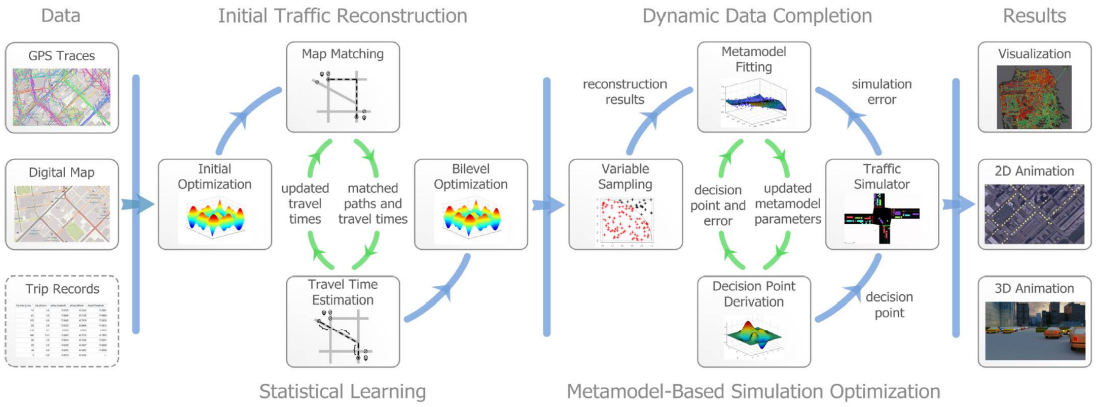

Figure 155: The systematic view of framework proposed by [41]. Trip records are optional as they can be inferred from GPS traces on a digital map.

5.4.5 Human driver modelling

In [318] a set of Driver behaviour models are presented for evaluating automotive active safety From neural dynamics to vehicle dynamics.

For the design of a driver model, a specific scenario is typically considered. Examples of such scenarios are lane keeping, double lane changes, or evasive manoeuvres. Depending in the scenario at hand, the driver model is required to generate different types of control inputs to the vehicle. Two categories of such control inputs are steering and braking behaviour. Steering behaviour is typically given as a steering angle for each time step, and braking behaviour is typically represented, for each time step, as a deceleration value. Looking at the variety of driver models, it is clear that such models cover a large range of different purposes, and are based on different underlying theories of human behaviour. When discussing different driver models from a design point-of-view it is useful to categorise models according to their structure. At least three different design domains can be identified in the development of driver modelling ([319]):

- The Control domain
- The Behaviour domain
- The Cognitive domain

5.4.5.1 Driver modelling based on Control theory

These types of models are mainly based on models addressed in the previous section dedicated to microscopic traffic simulation.

5.4.5.2 Driver modelling based on Behaviour theory

Many models of longitudinal control are designed from a behaviour perspective. Such models typically try to capture observed behaviour by complying to a few simple rules. For example, the GHR (Gazis–Herman–Rothery) model aims to capture observed behaviour when a vehicle is following a lead vehicle. Even though the GHR model is relatively simple, it can represent complex behaviour, especially when simulating a system with several vehicles each controlled by the model. The model has been used in many different applications and has in some cases been modified in order to capture scenario-specific behaviour, such as collisions. There are some driver models of lateral control designed from a behaviour perspective as well. These types of models are also based on models addressed in the previous section dedicated to microscopic traffic simulation.
In [42], the authors have proposed an interesting enhancement of IDM for the modelling of driver behaviour and aggressiveness using bio-behavioural methods.In this report and this work, a literature review is done on Driver behaviour models (VISSIM, Paramics, Urban Traffic Psycho-Physical Mode). From these existing works, a part focus on the modelling of situation awareness (Freeze-probe technique, real-time probe technique, self-rating technique, observer-rating technique), the mental workload (subjective measures, performance measures, physiological measures, sensitivity of the various measures), and the level of activation. The main goal of this work consists to proposed an unified and enhanced model involving these different concepts (situation awareness, mental workload, level of activation, and human performance).

The relationship between the work load, the level of activity, and the driver performance is presented in the figure 157.

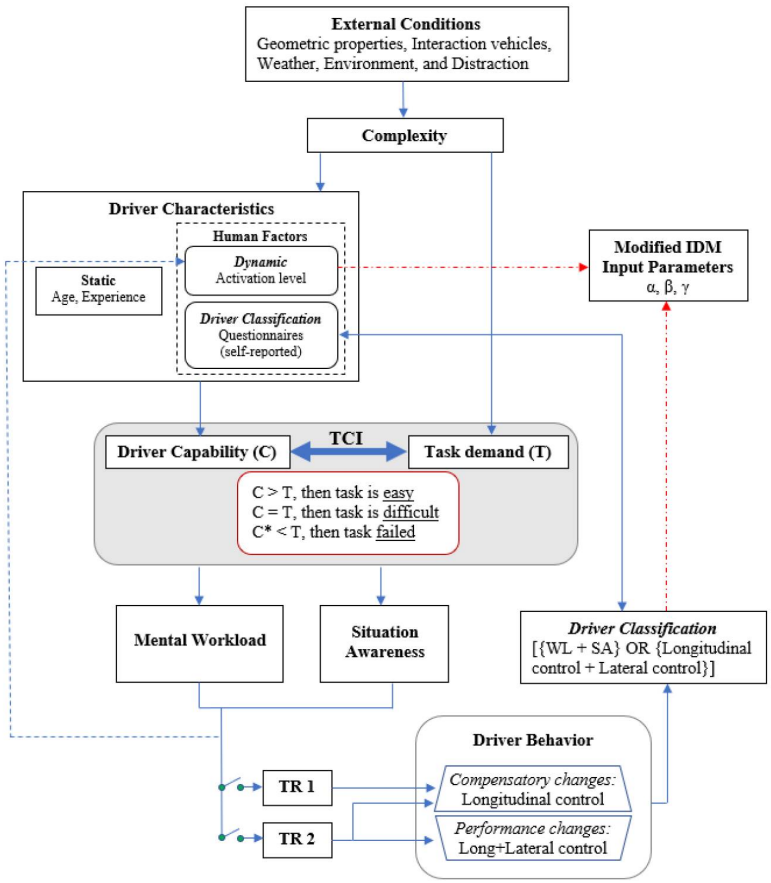

Figure 156: Theoretical framework for incorporating bio-behavioural parameters by [42].

In the figure 156, the following definitions are given and used:

- **Task demand**: concerns the amount of effort required to successfully meet the set requirements of a task, independently of the driver.
- **Driver capability**: The individual/biological characteristics of a driver affecting his/her ability to complete a task. These traits and characteristics include the speed, the reaction time, the information processing ability, the experience, the knowledge of driving, and the motor coordination.
- **Task difficulty**: The strategies or behaviour followed to cope with changes to task demand during a task. Task difficulty is inversely proportional to the difference between task demand and driver capability.
- **Work Load (WL)**: Represents the proportion of mental capacity required by an individual to perform a task.
- **SA**: The ability to perceive, understand, and project future status (prediction and anticipation capability) of elements/objects/components/situations in an environment.

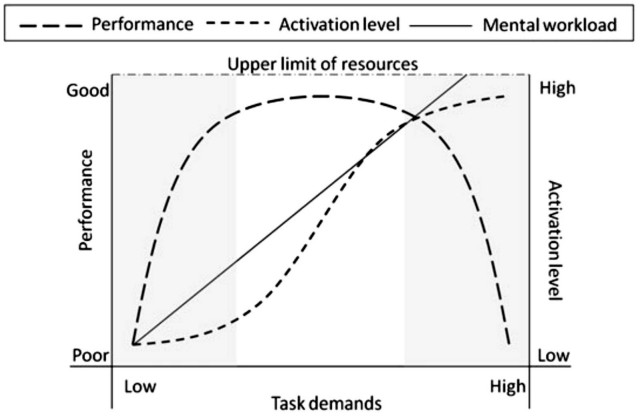

Figure 157: Relationship of WL (Work Load), LA (Level of Activation), and performance by [43]

In this extended IDM model, 2 triggers are implemented (TR1 and TR2). These triggers are link and dependant of the balance between WL and SA. For instance, TR1 is activated through small imbalances between WL and SA (see figure 156). However, if the imbalance between driver capability and task demand is high e.g. task is hard to be successfully completed by the driver's current capability, the driver tries to restore this imbalance resulting in both compensatory and performance effects (setting off TR 2). Where, compensation effects are theorised to only affect longitudinal driving variables while performance effects are theorised to affect longitudinal and lateral (SDLP) driving variables. Essentially, this implies that task difficulty is split into TR1 and TR2, depending on the extent of the imbalance between task demand and driver capability.

5.4.5.3 Driver modelling based on Cognitive theory

Driver modelling can also be approached from a cognitive perspective, using models of the human mind as a framework. In many ways, this approach is more difficult to apply than the other two perspectives mentioned above, the reason simply being that little is known about how the mind actually works. Therefore, models from this perspective are often only applicable to very specific driver phenomena observed in traffic. Still, as exemplified below, such models generally explain driver behaviour in a way that makes them applicable in a large number of scenarios.

A cognitive phenomenon, namely sensitivity to looming (or looming cues), could potentially be used when tuning or redefining the GHR model. In this context, looming refers to the optical expansion of an object in the driver's field of view, when the object is moving towards the driver. It has been shown that looming cues can trigger attention and reflex responses from a driver. Models defined from a cognitive perspective cannot, perhaps, be used in a standalone fashion, but could be used as a part of a model or in order to validate the behaviour of a model.

Another cognitive (or rather perceptual) model uses the perceptual properties of visual direction, and retinal flow in navigation (e.g. negotiating a curve). The visual direction can be explained as a target point in the visual field. In a driver model context, the visual field can be considered as the windshield of a vehicle. For instance, when moving straight towards a visual direction positioned in the centre of the visual field, the visual direction will remain stationary. However, when moving towards a visual direction positioned to the left in the visual field, the visual direction will rotate counterclockwise. The retinal flow can be explained as the motion of the

perceived features, such as colours and objects, projected on the retina. In order to capture human navigation, the model tries to accomplish two things:

- minimising the movement of the visual direction
- minimising the rotation in the retinal flow

In this way, Univ Eiffel has developed an efficient cognitive driver modelling usable in order to evaluate and validate critical situations involving mixed traffic configurations and share driving between virtual driving system for automated driving and the driver. This model is called COSMODRIVRE and has been interconnected with Pro-SiVIC platform (see figure 158)

Figure 158: Univ Eiffel and ESI's cognitive modelling of the human driver behaviour (COSMODRIVE).

5.5 Synthesis of Simulation Platform to Support Autonomous Driving Verification

The requirements for the simulation engine has to consider a set of essential functionalities and capabilities. A part of this function are summarised below:

- open interface to ensure compatibility with other platforms
- user friendly HMI with efficient command script (better if usable in real time during the simulation)
- minimal accessible objects, parameters, features, and data:
  - Ego vehicle state and dynamic information
  - Non-ego vehicle state and information (building, road sign, furnitures ...)
  - Pedestrian state and dynamic information
  - Accurate sensor models for automotive applications: camera, LiDAR, RADAR, GPS
  - Event engine
- Ground truth (states vector, depth map, road sign mask, segmentation, object semantic information ...)
- scenario management: generation, parameters and variable management

- road network modelling and traffic management
- rendering engine close to real environment
  - Lights and shadows manager
  - Materials and textures (static, dynamic)
  - Ray-tracing mechanism and library
  - Meta information mechanism: RCS, BRDF, conductivity, physical property of materials
- physical engine for dynamic modelling of complex object

Another criteria can enter into consideration like cost efficient (licensing, development effort, . . .)

In [44], the authors propose a state of the Art of sensor models for Virtual Testing of Advanced Driver Assistance Systems/Autonomous Driving Functions. In this journal paper, a table provide definition and expected capabilities for different level of sensor modelling (Low fidelity, Medium fidelity, High fidelity).

Moreover, as presented in [320],the development of simulation models have to respect the best practices listed below:

- A model should be designed to answer an operational question for which data are available.
- A model should not be used outside the context for which it is adequate.
- Careful consideration should be given to choices of abstraction and representation in order to construct an adequate simulation of the phenomenon of interest.
- Aspects of synthetic data, such as granularity, resolution, fidelity, and accuracy should be chosen to match the requirements of the context.
- A careful sensitivity analysis should be done to determine appropriate levels of tolerance along multiple dimensions of data and information quality to assess adequacy.
- An adequate simulation must convincingly capture at least some essential causal mechanisms driving the phenomenon of interest in its update mechanism.
- An adequate simulation should provide the means for effective use by domain experts and end-users.
- An adequate simulation is one that is used for ethical and moral good.

Among all the simulation platform presented in the figure 159, the NVIDIA DRIVE CONSTELLATION is interesting. This platform use 2 side-by-side servers in order to build a dedicated platform for running hardware-in-the-loop (HIL) simulation with NVIDIA DRIVE Sim. The 2 servers are dedicated to:

Simulators	Interface Compatibility	Ego-vehicle data access	Non-vehicle data access	Pedestrian data access	Detail and variety of sensors	weather conditions	Rendering performance	Physical engine	Scripting langage	Traffic model (3D map, dynamics)	V2X communication	scene diversity	Cost efficiency	Company
CARLA						yes	GPU	Unreal	Python	SUMO		No	GPL Open source	CVC
Gazebo							Ogre3D	ODE, Bullet, Simbody Art					GPL/ Open Source	Open Robotics
SynCity														
PreScan	XIL						Unreal	CarSim		Vissim			Commercial	Siemens
VTD														
AutonoVi-Sim						Yes	OpenGL	Unreal	C++/Python			No		
Righthook														
IPG CarMaker	HIL													IPG
Racer														Cruden Inc.
SCANeR											iTetris			AV Simulation
AirSim						yes		Unreal	C++, Python, C#, Java			No	GPL Open source	Microsoft
rFpro	HIL						rFpro			SUMO			commercial	
Udacity						yes		Unity	C++, Python			no	GPL/Open source	
Cognata														
Aimsun+Vissim														Aimsum, PTV Group
Pro-SiVIC					GPS, INS, Odo, LiDAR, Radar, cam, conso,	yes	OpenGL/GPU	mgEngine, own ODE, raytracing	LUA, own langage	Local traffic based trk map and controlers, SymuVia	NS-3+DDsL+ propag channel+ant enna			ESI group
Sumo														
CVIS														
RoadView														
USARSim														
PELOPS														
MORSE														
4DVsim														
VRXPERIENCE													Commercial	ANSYS
Metamoto														
Uber platform														
Cruise														
Carcraft (waymo)													restricted	Google
Nvidia Drive constellation						yes	GPU	PhysX/CUDA	C/C++/Python			yes	restricted	Nvidia
Vires Virtual Test Drive														Vires
CarSim														
Deepdrive	HIL					yes		Unreal	C++, Python			no	GPL/Open source	
LGSVL														LG
Sim4CV														
SimLidar						no			C++			no	GPL/Open source	
Helios++						yes	OpenGL	Jmonkey Engine	Java			yes	GPL/Open source	
RADSim						No			Matlab			No	Commercial	
DYNA4	HIL						Unity	Simulink		SUMO			Commercial	
CoInCar-Sim														KIT Univ.
CoppeliaSim														Coppelia Robotics
Matlab AD Tollbox														MathWork
Vdrift														Vdrift group
Autoware														Tier IV Inc
ASM Traffic														Dspace
AutoSim														Nvidia
SIMLidar					LiDAR									
GLIDAR					LiDAR									
RADSim					RADAR									
Udacity														
OpenDaVINCI														Chalmers
CarND-path planning														Jeremy Shannon
TORCS													Open Source	
Webots	User interface:Driving						OpenGL	ODE		SUMO			Open Source/commercial	cyberbotics
CAT							Ogre3D, Gazebo	ODE		SUMO				
USARSim					GPS, IMU, LiDAR, UltraSonic, Radar, IR, cam		Unreal, Karma	Unreal					Open Soource	
dSPACE								ASM						
CoSAM														
BlenSor					GPS, IMU, LiDAR, Ultrasonic, Cameras, ToF cam		OpenGL						Open Source	
MORSE					GPS, IMU, LiDAT, IR, cam		OpenGL	Blender, Bullet					Open Source	
MRDS (Microsoft Robotics Dev. Studio)					GPS, LiDAR, ultrason, IR		DirectX	PhysX					open source	Microsoft
Apollo EM Motion Planner														Baidu
SIMSonic					ultrasonic	No			Matlab, R			No	GPL/Open source	

very poor
poor
not rated or irrelevant
0
good
very good
not yet implemented but can be done

Figure 159: Synthesis of Simulation Platform to Support Autonomous Driving Verification (to be improved, updated,and modified

	Low fidelity	Medium fidelity	High fidelity
Operating principles	Geometrical aspects	Physical aspects, detection probabilities	Rendering (rasterization, ray tracing, etc
input	Object lists	Object lists	3D scene (meshs)
output	Object lists	Object lists or raw data	raw data
Pros	Low computational power needed	Trade-off between computational power and realistic output, a lot of effects can be considered	Most realistic output
Cons	High abstraction level, no realistic output	Lots of training data may be required	High computational power needed
V-model phases	First specification phases	Specification phases in the middle and integration phases	Component specification, implementation and integration phases
Design question	What point(s) or shape represents objects and which need to be in the line of sight for detection?	What point(s) or shape represents objects and which need to be in the line of sight for detection? What effects are considered?	What is the detection threshold? Which effects, material properties, and weather conditions are considered?

Figure 160: Overview of the properties of low-, medium-, and high-fidelity sensor mode ([44])

- First server: the NVIDIA OVX Server uses NVIDIA GPUs running DRIVE Sim software to simulate the virtual world. The simulator generates the sensor output from the virtual car driving in a virtual world. In order to obtain very realistic and accurate modelling and operating of sensors, NVIDIA use their dedicated NVIDIA RTX library for high performance ray-tracing.
- Second server: DRIVE Constellation Vehicle contains the autonomous vehicle target computer that processes the simulated sensor data and feeds driving decisions back to the OVX Server.

Figure 161: NVIDIA Constellation, a distributed architecture with 2 servers for HiL simulation (https://developer.nvidia.com/drive/drive-constellation)

Figure 162: NVIDIA DRIVE Sim, screenshot of the rendering

In [11], a Survey is done about the Scenario-Based Testing for Automated Driving Systems in High-Fidelity Simulation. In this survey, a list of system and simulation platform are

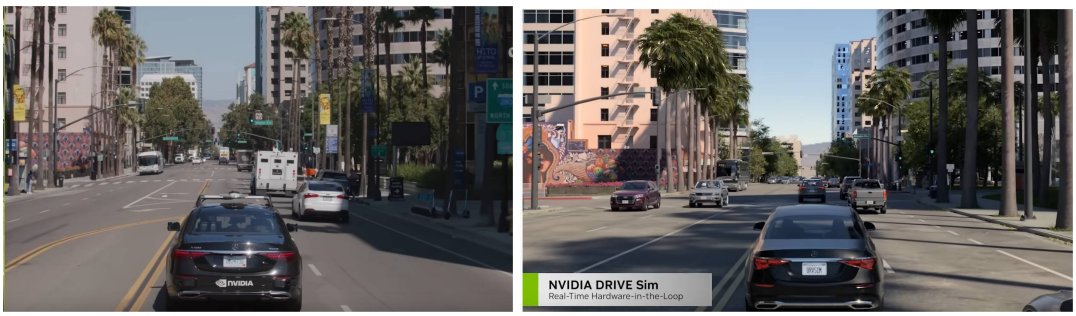

Figure 163: NVIDIA DRIVE Sim compared to same real urban situation

presented and give an interesting overview of the main High Fidelity simulator used for the validation of AV components and systems. In the figure 164, the first column provides the year of the development and publication of the work, the second column provides the reference of the paper, the third one gives the used simulation platform (9 simulation platforms have been mentioned: PreScan, Webots, IPG CarMaker, BeamNG, Vires VTD, AirSim, SVL, Paracosm, NVIDIA Drive Sim), the fourth one provides the system under test, the firth one the testing objectives, then after, the author provide the layer(s) containing the modified parameters and the type of implementation for the tested algorithm. In this comparison, the authors have used a previous version of the environment description layers with only 5 layers. These 5 layers allows to model both the environment ans the scenario class of elements. This modelling proposes a format to structure the parameters describing a scenario. In PEGASUS project, this description has been extended to 6 layers. In PRISSMA project, we have extended this 6 layers with a new structure dedicated to the ego-vehicle (4 new Layers), and a structure for event and time management with 3 Layers.

- Layer1 describes the layout of the road and which parameters influence the road's layout. This parameters include markings, topology (e.g., curvature, number of lanes, types of junction), and surface properties (e.g., friction coefficient).
- Layer2 defines traffic infrastructures (e.g., traffic signs/lights).
- Layer3 is about the temporary manipulation of Layer1 or Layer2.
- Layer4 describes all the dynamic objects, their manoeuvres, and interactions in a scenario. Parameters of layer4 influence the objects and their behaviours. The parameters of this Layer can be shared into three sub-categories:
  - control at initialisation (denoted as 4(1)),
  - control for activated behaviour (denoted as 4(2) for one-time activated behaviour parameters or 4(k) for multiple-times activated behaviour parameters),
  - and control at every step (denoted as 4(n)).

  Parameters belonging to control at initialisation usually only influence an object's behaviour at the initialisation time. Parameters belonging to control for activated behaviour usually define some activation criteria of an object's behaviour change and its behaviour after being activated. Parameters belonging to control at every step usually control an object's behaviour at every time step during the simulation.

- Finally, Layer5 describes the environmental condition like weather and lighting.

Y	Tool Name	Simulator	System	Testing Objective	Layer	Algorithm
16	NSGA2-SM [43]	PreScan	FCW	Severe (Critical,Responsible)	1,4(1),5	Sim,EA
18	FITEST [44]	PreScan	2*ADAS	More (Integration-Induced),Subjective	2,4(1),5	Sim,EA
18	NSGA2-DT [45]	PreScan	AEB	More/Severe (Critical,Responsible),Input	1,4(1),5	Sim,EA
18	Tuncali et al. [46]	Webots	CCF	More (Critical,Responsible),Input	4(1)	Sim,SA
18	Nitsche et al. [47]	IPG CarMaker	2*AEB	Collision Rate, Task Failure Rate	1,4(1)	N,MC
18	Zhou et al. [48]	IPG CarMaker	ACC	Critical Boundaries	4(1)	Sim,BO
19	AC3R [49]	BeamNG.tech	DD	More (Consistent with Description)	U	N,Other
19	ASFAULT [50]	BeamNG.tech	BeamNG.AI,DD	More (WrongLane,Diverse),Input Distance	1	Sim,EA
19	Kluck et al.2 [51]	Vires VTD	AEB	First (Critical)	4(1)	Sim,SA
19	Kluck et al. [52]	Vires VTD	AEB	Severe (Critical)	4(1)	Sim,EA
19	FAILMAKER-ADVRL [53]	AirSim	U	More (Critical,Natural),All	4(n)	Step,DRL
19	Abey et al. [54]	CARLA	CVILLF	First (Critical)	4(n)	Sim,BO
19	Gangopadhyay et al. [55]	IPG CarMaker	U	More (Critical,Diverse), Input Region	4(1)	Sim,BO
20	Sim-ATAV [56]	Webots	CMSTF	First (Requirements-Violating)	4(1)	Sim,SA
20	AVfuzzer [57]	SVL	Apollo3.5	More (Critical, Diverse),Subjective	4(k)	Sim,EA
20	Norden et al. [58]	CARLA	OPENPILOT0.5	Critical Rate	4(n)	Sim,IS
20	Kuutti et al. [59]	IPG CarMaker	2*Nav(RL, IL)	First/More (Critical),All	4(n)	Step,DRL
20	Ding et al. [60]	CARLA	CARLA PID	More (Critical),All	4(2)	Sim,ARM
20	Zhu et al. [61]	PreScan	ACC	More (Critical, Diverse),Input	4(1),5	Sim,Other
20	Bussler et al. [62]	Vires VTD	U	Severe (Critical)	4(2)	Sim,EA
20	Li et al. [63]	Vires VTD	AEB	More (Critical),All	4(1)	N,Other
20	Saquib et al. [64]	CARLA	CARLA PID	Critical Value	4(1)	Sim,Other
21	AutoFuzz [65]	CARLA	CARLA PID, LBC	More (Critical/WrongLane,Responsible,Diverse),Input Distance	1,4(2),5	Sim,EA
21	Paracosm [66]	Paracosm	Nvidia CNN LF	More (Critical, Diverse),Input Dispersion	1,4(2),5	Sim,CLS
21	FusionFuzz [67]	CARLA	OPENPILOT0.8.5	More (Critical,Fusion-Induced,Avoidable),Trajectory	4(k),5	Sim,EA
21	Ding et al.2 [68]	CARLA	6*Nav(RL)	More (Critical, Diverse),All	4(2)	Sim,Other
21	Chen et al. [69]	CARLA	2*ALC2	More (Critical,Task Failure,Rule-Following,Diverse),All	4(n)	Step,DRL
21	ASF [70]	SVL	Apollo5.0	More (Critical,Diverse),Subjective	4(1)	Sim,EA
21	Ghodsi et al. [71]	Nvidia Drive Sim	U	More (Critical),All	4(2)	N,Other
21	Tang et al. [72]	SVL	Apollo5.0	More (Critical,Diverse),Subjective	1	N,Other
21	Tang et al.2 [73]	SVL	Apollo5.0	More (Critical,Diverse),Subjective	1,4(1)	Sim,CLS
21	Tang et al.3 [74]	SVL	Apollo6.0	More (Critical,Task Failure,Diverse),Subjective	1,4(1)	Sim,EA

Figure 164: Overview of a set of works made in the 5 past years on automotive systems evaluation with simulation ([11])

The figure 165 presents the 9 simulators' functions and capabilities.

Simulator	Source	Engine	Sensors	Assets	Map	Scenario	Compatibility
CARLA [76]	Y	UE4	C L R G	V P	Urban Rural Highway	API	Python C++ ROS1/2
SVL [82]	Y	Unity	C L R G	V P	Urban	API GUI	Python ROS1/2 CyberRT
AirSim [83]	Y	UE4	C L G	V P	Urban Rural	API	Python C++ C# Java ROS1
Paracosm [66]	Y	Unity	C	V P	Urban	API	Python
Webots [84]	Y	ODE	C L R G	V P	Urban Rural Highway	GUI	Python Matlab ROS1/2
BeamNG.tech [85]	N(A)	Proprietary	C L R G	V	Urban Rural Highway	API GUI	Python
PreScan [86]	N	Proprietary	C L R G	V P	Urban Rural Highway	API GUI	Python C++ Matlab
CarMaker [87]	N	Proprietary	C L R G	V P	Urban Rural Highway	API GUI	Matlab C
VTD [88]	N	Proprietary	C L R G	V P	Urban Rural Highway	GUI	U

Figure 165: Overview of simulators covered. C:Camera, L:LiDAR, R:Radar, G:GPS, V:Vehicle, P:Pedestrian. ([11])

6 Actions in progress (WG, projects, initiatives, developments, etc.) and actors involved

Ongoing automotive actions are described in more detail in deliverable 8.4 (This deliverable covers a great part of the needs of this section) but in this section we give a shirt overview of relevant projects, actions, and initiatives.

6.1 France

6.1.1 MOOVE dataset with annotation

This project coordinated by VEDECOM and involving Renault, Stellantis, and Valeo are Building a dataset for AD evaluation and validation from a huge number of data logging made in a large number of road configuration. On of the objective of this project is to provide sensor data and road scene annotation (generation of a ground truth).[321][322]

6.1.2 Working Group sensors and sensors simulation for all weather condition ADAS

The core group of this Working Group involved University Gustave Eiffel, CEREMA, and CARA. The objective is to Position a new type of ”all weather” sensors or the ”all weather” multi-sensor perception architecture at the heart of the Autonomous Vehicle Industry. The second objective is to propose models about the quality of sensors and data in order to propose an efficient way to use sensors in function of their current state, behavior, and capacity. The first objective is to propose the definition and the design of an all-weather sensing and perception system for AD systems. This architecture would be based on three levels of interaction between the equipped vehicles and their environment. About the simulation part and the topics of evaluation and validation of sensors, perception, decision, actions modules, the partners propose to:

- Adapt the ImPACT 3D simulator to extend it to the problem of industrial vehicles in particular (buses, trucks, shuttles, etc.)
- Validate sensor and perception architectures and capabilities in representative environments on the existing tracks of TRANSPOLIS (urban circuit with accurate geometric measurement and digital twin in progress), PAVIN (rain, fog), Satory (existing digital twin), and ZEHNS in Bordeaux (digital twin in progress)
- Validate the modelling of the estimation of the sensors quality and performances in a controlled simulated environment.
- define test, evaluate, and validate procedures for sensor technologies, capabilities, behaviours, weaknesses, limits. This part addresses too the question of the more relevant and efficient configurations and topologies of different technologies and adaptive perception architectures.
- address the operating safety and cyber attacks problems at the sensor level.
- Deploy real use cases on open road areas (experimentation) and in digital twins of these road areas.

6.1.3 MOSAR

MOSAR platform involves a set of French partners working on a project to industrialise a scenario library and the associated tool-chain allowing to:

- Answer the question of ADS safety and validation (SAE level 1 up to 3) with the proposal of a scenarios library built from multiple sources (i.e MOVE dataSet).
- address different driving operational domains
- take into account multiple cultural expertise
- cover a great part of accidents and incidents situations and configurations
- provide sufficient observation time
- establish the “common reference” in a unique structured library, with open export format to initialise the combinatory for exhaustive simulation

The scenario library is supported by the French government as French scenarios library for AD&ADAS safety validation. This initiative enters in PFA Proposals for AD safety validation and French scenarios library generation.

6.2 Europe

6.2.1 Regulation 2022/1426/EU - Type-approval of the Automated Driving System (ADS) of Fully Automated Vehicles

The European Commission proposes a recent document explaining how to implement the regulation (EU) 2019/2144. This document has been published in 05th August 2022 (REGULATION (EU) 2022/1426). More accuratly, the content of this regulation document address the laying down rules for the application of Regulation (EU) 2019/2144 of the European Parliament and of the Council as regards uniform procedures and technical specifications for the type-approval of the automated driving system (ADS) of fully automated vehicles. In the appendix 1 of this document, an interesting diagram (see figure 166) is presented about the principles, functions, and relationships to be followed to derive scenarios relevant for the ODD/OEDR of the AD.

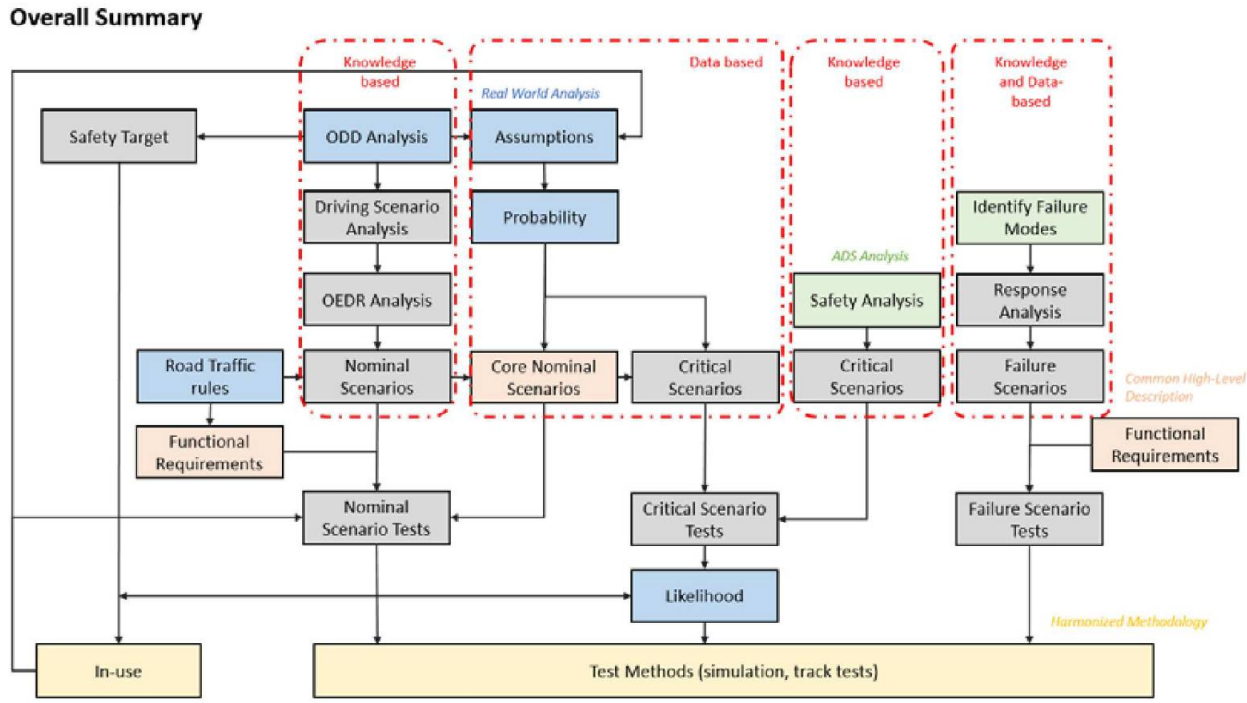

Figure 166: principles, functions, and relationships to be followed to derive scenarios relevant for the ODD/OEDR of the AD ([45])

In this regulation document, some definitions are given about the ODD and OEDR:

- **The ODD** consists of scenery elements (e.g., physical infrastructure), environmental conditions, dynamic elements (e.g., traffic, vulnerable road users) and operational constraints to the specific ADS application. The aim of this analysis is to identify the characteristics of the ODD, allocate properties and define interactions between the objects. Here the effect of ODD on the behaviour competencies of the ADS is explored.

- **The OEDR** provides the behaviour competency identification. Once the objects and relevant properties have been identified, it is possible to map the appropriate ADS response. The ADS response is modelled on applicable functional requirements and by applying the performance requirements of this regulation and the traffic rules of the country of operation. The outcome of the OEDR analysis is also a set of competences that can be mapped to the behavioural competences applicable to the ODD, to ensure compliance with the relevant regulatory and legal requirements.

Moreover, both critical and failure scenarios are defined accuratly. This document can be found and downloaded in the following website: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32022R1426

6.2.2 V-Suites^{TM} and Foretify^{TM} platforms from Foretellix

The Foretify^{TM} platform is an industry's leading coverage-driven verification and validation platform for ADAS/AV development. This product is built on open industry standards and methodology, Foretify^{TM} proposes a solution for a massive-scale and measurable objective approach to verifying and validating autonomous systems' safety and productivity. Foretify^{TM} provides a massive-scale generation of millions of ODD/Map relevant tests to expose bugs, edge cases, and unknowns. This Open Platform with Open Standards is based on open Industry standards such as OSC 2 and supporting over 15 testing platforms and requirements systems. After the scenario definition, generation, and management, a set of simulation tools can be used in order to generate data allowing to analyse, evaluate and validate services, systems, functions, and components under test in specific and critical scenarios. This downstream process is done with the Safety productivity dashboard.

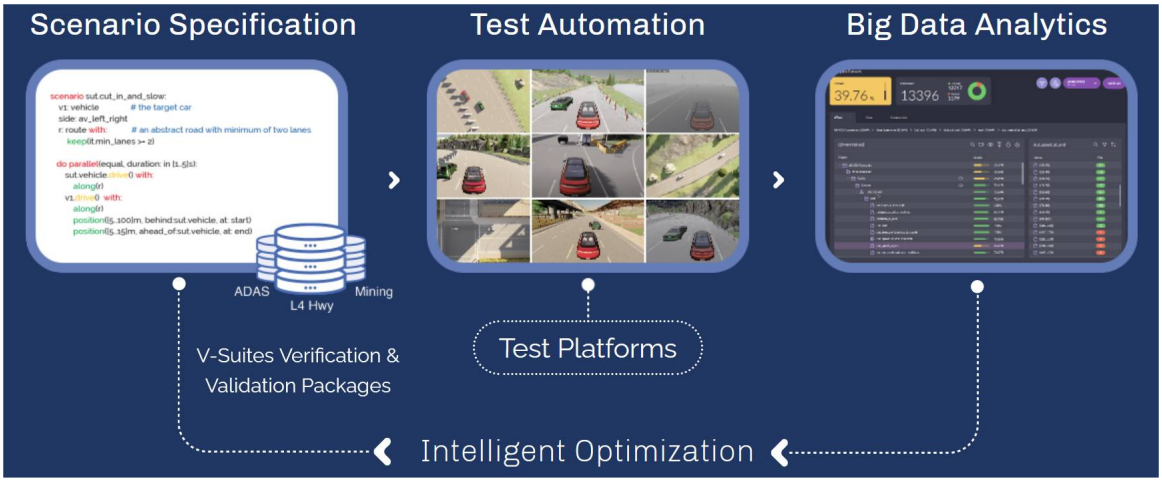

Figure 167: Overview of the foretify^{TM} process (https://www.foretellix.com/technology/#)

V-Suites^{TM} are Verification Validation (VV) libraries, offering millions of test scenarios. V-Suites^{TM} are targeted at common ADAS/AV functions and ODDs and are written in ASAM OpenSCENARIO 2.0.

V-Suites^{TM} enable to manage a testing program with predefined verification plans, abstract scenarios, maps, coverage, KPIs, and checkers. V-Suites^{TM} are built to test specific functions, regulations, use cases, and ODDs to assist users with various validation needs. V-Suites^{TM}

Main Features are Executable Verification Plans, Library of Scenarios, Maps, and Coverage goals, KPIs, and Checkers.

6.2.3 Validation Methods of Automated Driving – Scenarios Sub-working Group

This Scenarios Sub-working Group from the Working Group Validation Methods of Automated Driving is an international initiative with information coming from several continents. The first objective is to address the development/implementation of the VMAD safety validation methodology. In this context, the Sub Group 1a has tackled its efforts on the development of a methodology to identify scenarios in a structured approach. Initially, focusing on a more simple operational design domain (ODD) such as divided-highway driving, and in the longer-term focus on other ODDs, including the possibility of edge cases. When developing scenarios for ODD, the group focuses its attention on the development of functional scenarios. By limiting the scope of work at this time to functional scenarios, SG 1a would not define specific parameter ranges (logical scenarios) or specific values for the scenario elements (concrete scenarios) that are tested via simulation, track and real-world testing. As the project progresses, SG 1a works with other sub-working groups (i.e., SG2a and SG2b) to focus its attention on more detailed/technical scenarios (i.e., logical and concrete scenarios). the SG 1a sub-group has made a literature review to identify and leverage existing materials from groups, such as international standards setting bodies, government departments, industry, and academia, who are active in developing scenarios. The purpose of the literature review is: 1) Identify/leverage areas of prior research/work to prevent duplication and recognise the work of others, which could assist with developing a scenarios catalogue; and 2) Identify gaps in the research/work, including conflicts in previous studies, and/or open questions/next steps raised from previous research that require attention by VMAD.

The literature review has been done based on the following topics/concepts:

- Methods for identifying scenarios
- Scenario description language
- Definitions
- Levels of Abstraction
- Operational Design Domain and Scenario Elements/Properties
- Measuring Safety/ scenario coverage

6.2.4 Streetwise from TNO

The StreetWise methodology provides automatic identification and characterisation of scenarios from object level data provided by state-of-the-art sensors for Advanced Driver Assistance and Automated Driving systems (ADAS/AD) and focuses on using such scenario statistics for safety argumentation. This methodology is under continuous development with international partners (Germany, Japan, Austria, Netherlands, Singapore). An important ambition is to identify and characterise all highway scenarios out of a significant amount of hours of driving on European highways. In this way a continuously growing scenario data base is set up, from which tests for ADAS and AD functions can be derived. Partners in such collaboration share scenarios and thereby share efforts in determining the important scenarios for testing, however,

without sharing sensitive data. Automated scenario identification and characterisation has been developed in order to deal with:

- The huge number of possible scenarios, and the large variety in scenarios (e.g. different velocities, number of relevant traffic participants, etc.)
- Variations in occurrence of scenarios are captured in automated scenarios to result in parameter distributions for the characteristic scenario parameters. These distributions are used in metrics that quantify the completeness of a scenario database.
- Additionally, such automated methods will show whether all captured data can be applied to known scenario categories. In case a set of data is found that does not fit in an existing scenario category, an additional scenario category is required to be defined. This also relates to the completeness paradigm.
- All scenario statistics are captured in an online accessible database from which test cases can be sampled based on scenario parameter values and tags. These test cases can be used for massive simulation or other (physical) tests.
- Automatic scenario extraction has been developed for highway. The method is extended towards urban driving.

6.2.5 MUSICC: An open catalogue for CAV certification scenarios

The MUSICC (Multi User Scenario Catalogue for CAVs) project was led by the UK Department for Transport and Connected Places Catapult. This project has for main objective to generate and to build an open source scenario database. The second objective is to prove the capability of such database and underlying concepts to be used in an evaluation and validation framework for CAV approval. This framework should evaluate highly complex systems in a transparent and fair way, while restricting innovation as little as possible. In [46], the requirements, format and implementation of the database are outlined and discussed in the context of the wider system and test process. The main key requirement is the ability to search for and export a subset of scenarios appropriate to certification of a particular ADS in a particular territory. This implies support for finding all scenarios that fall within a specified ODD. Other requirements for the database include:

- Provide easy remote access to scenarios stored in a machine-readable format.
- Store version history for scenarios.
- Support scenario management and approvals processes.
- Provide an easy-to-use API for scenario export, to allow tool integration.
- Provide a web-based UI, supporting searches based on ODD, edit, user management, etc.
- Scale to store a sufficient number of scenarios in a robust way.

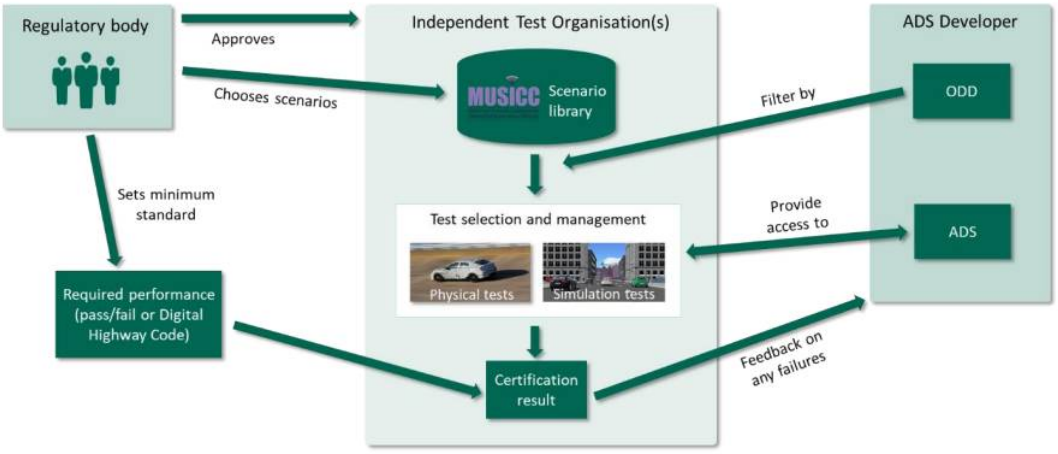

Figure 168: Illustration of MUSICC in the context of a certification process ([46])

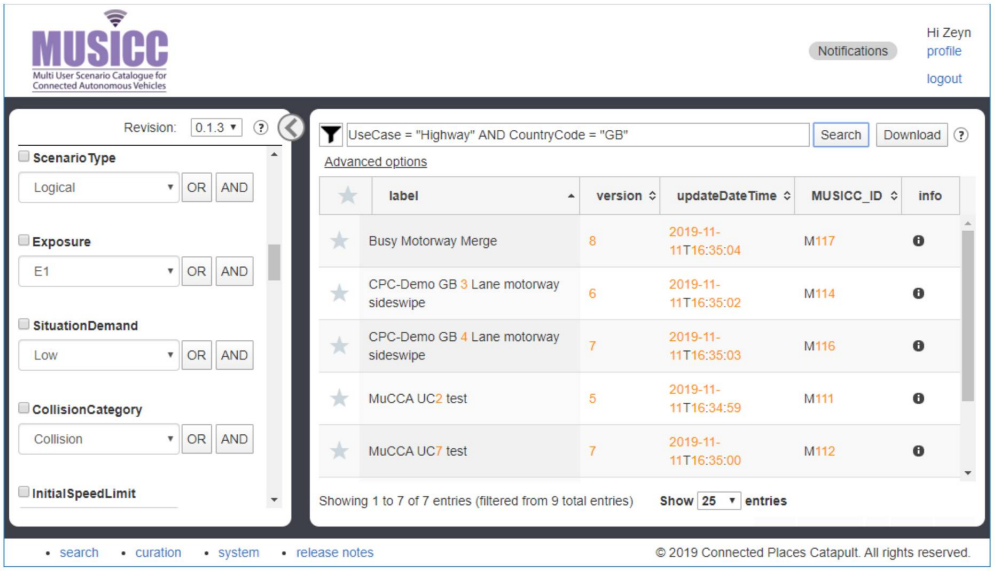

Figure 169: Screenshot of MUSICC's main page ([46])

MUSICC has produced a system (open library) to store scenarios for use in AV safety tests. Scenarios are stored with metadata that allow them to be searchable and to be used for scenario selection, based on certain criteria:

- general properties of the scenario (information about the test case which the scenario describes);
- ODD specification (to identify scenarios matching a specified ODD)
- admin data (e.g. ownership version control and reference to specifications and regulations)
- custom data (user extensible).

MUSICC's SDL represents scenarios using three records: an OpenDRIVE, OpenSCENARIO, and MUSICC record (all stored as XML). The MUSICC record stores additional necessary data beyond the OpenX standards: scenario metadata and data to enable parameter stochastics. This system has been released as open source and can be downloaded with the following address: https://gitlab.com/connected-places-catapult/musicc. As a result, developers can store and manage their own test scenarios in a private instance of MUSICC within their own organization and still synchronize with the regulatory scenarios. The whole community will then be able to contribute to the development of an open library for AV certification scenarios. Next steps of this project include:

- the development of a more sophisticated language to better match ODD description with all likely use cases and help inform development efforts
- development of a framework for assessing an ADS's behavioral safety in normal operation. The framework will include a highly automated means of evaluating test results

6.2.6 H2020 HeadStart project

The HEADSTART (Harmonised European Solutions for Testing Automated Road Transport) project is an EU funded project which started on the 1st of January of 2019. The project involved 17 European partners, under the coordination of IDIADA. This project aims to define testing and validation procedures of Connected and Automated Driving functions including key technologies such as communications, cyber-security and positioning. The tests will be in both simulation and real-world fields to validate safety and security performance according to the key users' needs.

The HEADSTART project proposes to cluster the most relevant existing initiatives, develop methodologies, procedures and tools and drive in a harmonized European solution for testing and validation of automated road vehicles. The main Headstart objectives are:

- Identify the existing methodologies, procedures and tools for testing, validation and certification;
- Harmonise the existing testing and validation approaches;
- Define and develop test, validation and certification methodologies and procedures for CAD functions;
- Demonstrate the developed methodologies, procedures and tools through testing 4 CAD use cases;
- Reach consensus by creating and managing an expert network of CAD testing to promote adoption of the project results considering multi-stakeholder needs.

In some partners of HeadStart propose the design and the implementation of an Ontology for Semantic Labeling and Testing called "Automotive Global Ontology" (AGO) (see the figure 170). An overview of the general methodology and architecture proposed in the HEADSTART project is presented in the figure 171

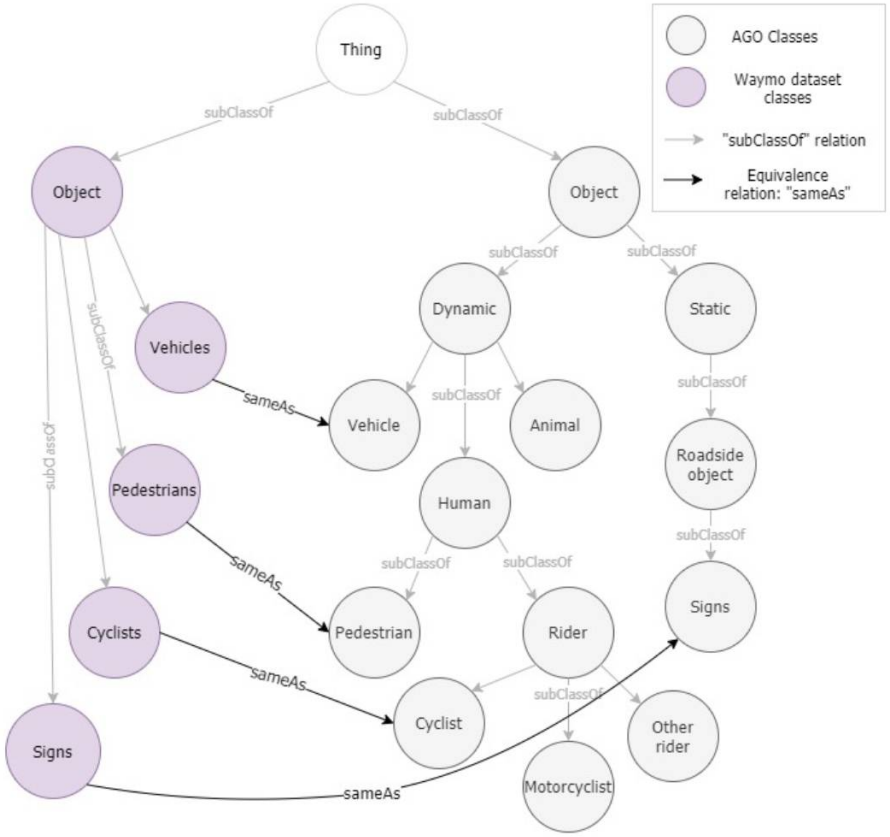

Figure 170: Waymo dataset mapping to AGO example. Matched classes represent the common terms among the different datasets ([47])

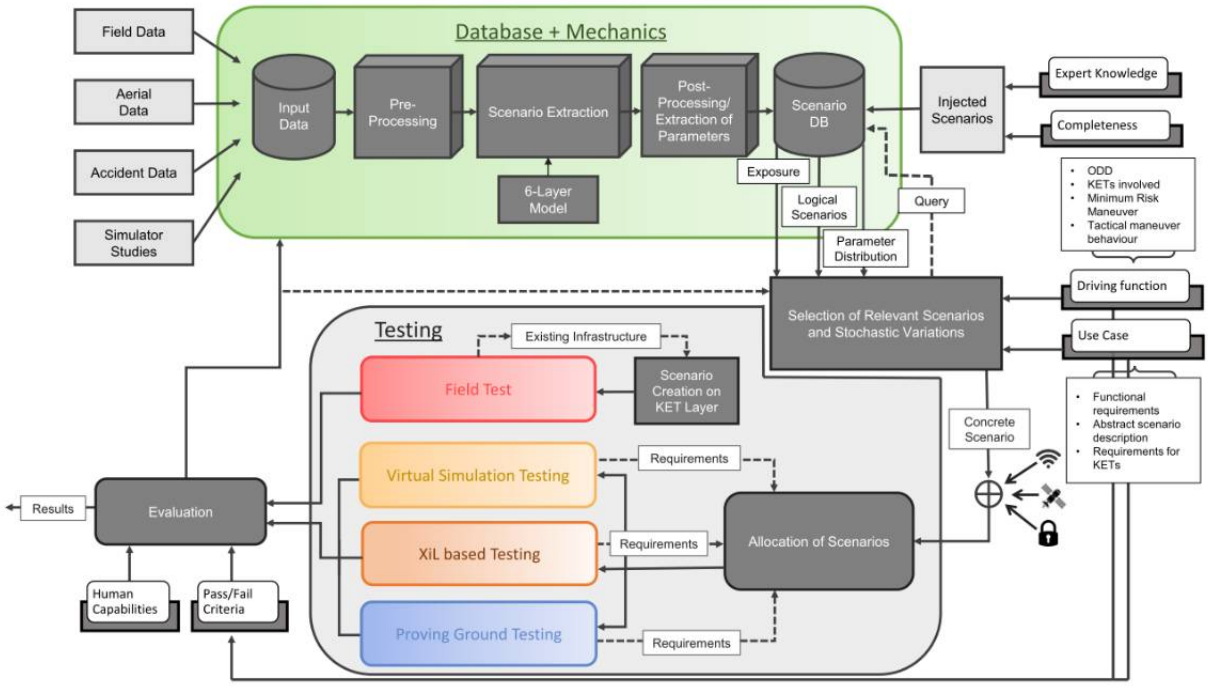

Figure 171: The general methodology and architecture proposed in the HEADSTART project ([47])

In this type of architecture, a map of capabilities (MoC) (figure 172) for each testing method is available with the definition of three main categories of tools and testing methods. Moreover to this definition is added “resource-based” capabilities like time, costs and availability (e.g. available area of a proving ground).

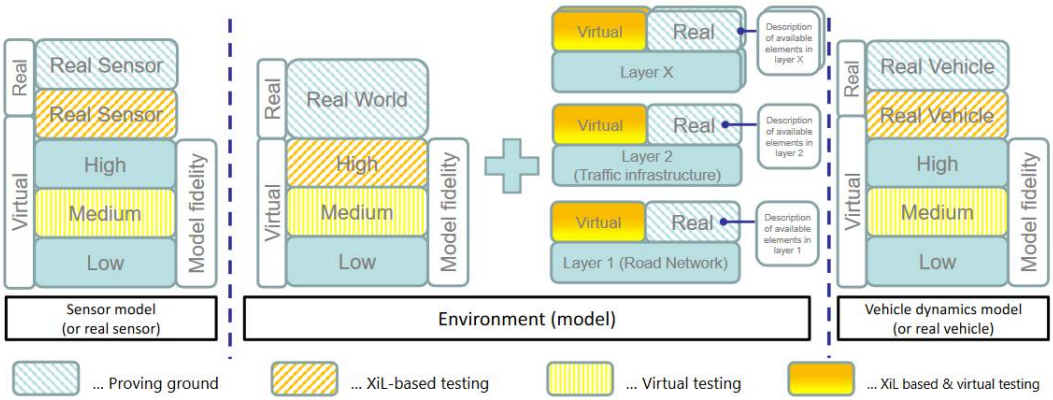

Figure 172: Map of capabilities proposed in the HEADSTART project for the different test methods ([48])

6.2.7 Horizon Europe ROADVIEW project (2022-2026)

ROADVIEW: Robust Automated Driving in Extreme Weather is led by Halmstadt University (Sweden). Partners are Lapin ammattikorkeakoulu Oy, FI, Technische Hochschule Ingolstadt, DE, Statens väg- och transportforskningsinstitut, SE, CEREMA - Centre d etudes et d expertise sur les risques l environnement la mobilite et l amenagement, FR, RISE Research Institutes of Sweden AB, SE Maanmittauslaitos, FI, Synthetic Data Solutions AB, SE, Konrad GmbH, DE, Ford Otomotiv Sanayi A. S, TR Canon Research Centre France S.A.S., FR, ZF Friedrichshafen AG, DE, University of Warwick, UK, accelopment Schweiz AG, CH.

The project aims to develop robust and cost-efficient in-vehicle perception and decision-making systems for connected and automated vehicles with enhanced performance under harsh weather conditions and different traffic scenarios. https://roadview-project.eu/

The ROADVIEW workplan is as follows:

- WP2: ODDs expansion and definition of the ROADVIEW system setup;
- WP3: digital models enhanced by controlled and real-world environments;
- WP4: secure sensor data processing and data quality;
- WP5: perception system and collaborative perception performance;
- WP6: control and decision-making system.

The implementation and validation of the X-in-the-loop test environment approach is considered in WP7 and the final integration and demonstration of the perception, control and decision-making systems in the city and highway driving will be carried out in WP8.

6.2.8 PEGASUS project

PEGASUS project start from the report that safety approval cannot be achieved for highly automated vehicles with available methods and tools within a limited time and budget. In this context, PEGASUS project have proposed to develop a framework for AD assessment using the highway chauffeur as exemplary test object, i.e. a SAE L3 conditional automation system (https://www.pegasusprojekt.de/en/pegasus-method). The central element of PEGASUS is a data base and an according data base processing toolchain. The toolchain must be capable to include and use different data sources and therefore heterogenic data and data quality. The proposed data base concept can realise an efficient and effective data processing in a common framework with a common tool chain. In the project, scenarios are derived via combining a data driven (main focus) and a knowledge-based approach:

- data driven: using available, not strictly confidential data to determine scenarios that are usually encountered in traffic: Naturalistic driving studies (NDS), field operational tests (FOT), test drives driving simulator data, German In-Depth Accident Study (GIDAS)
- knowledge-based: using further sources to also cover rare events that are possibly critical scenarios: Road traffic regulations, traffic sign catalog, guidelines, laws and standards, expert knowledge

A first set of scenarios is stored in the PEGASUS database. To allow for this, PEGASUS developed a scenario description language, which is now managed and further developed by ASAM as the freely available standard OpenX. PEGASUS project ended in 2019 but work is being continued in the PEGASUS family with projects “VV-Methods” and “SET Level 4 to 5”.

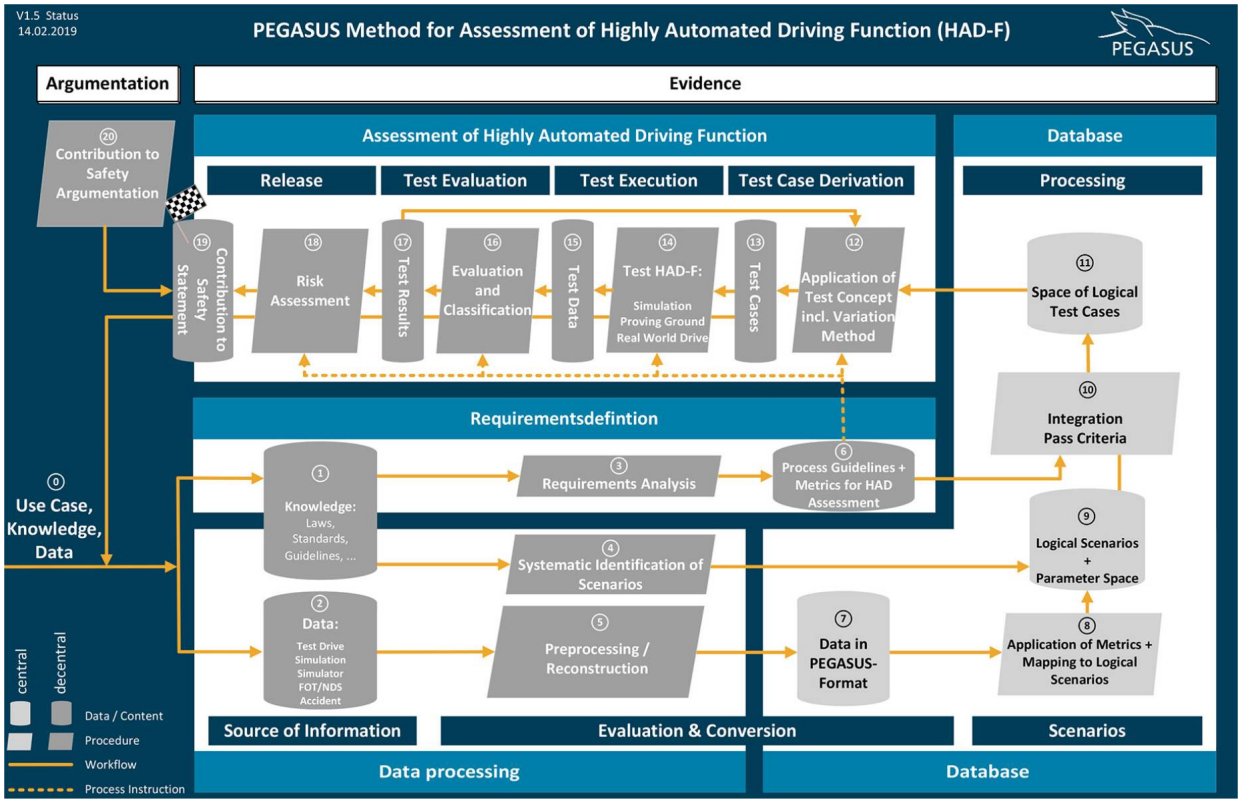

Figure 173: The general methodology and architecture proposed in the PEGASUS project

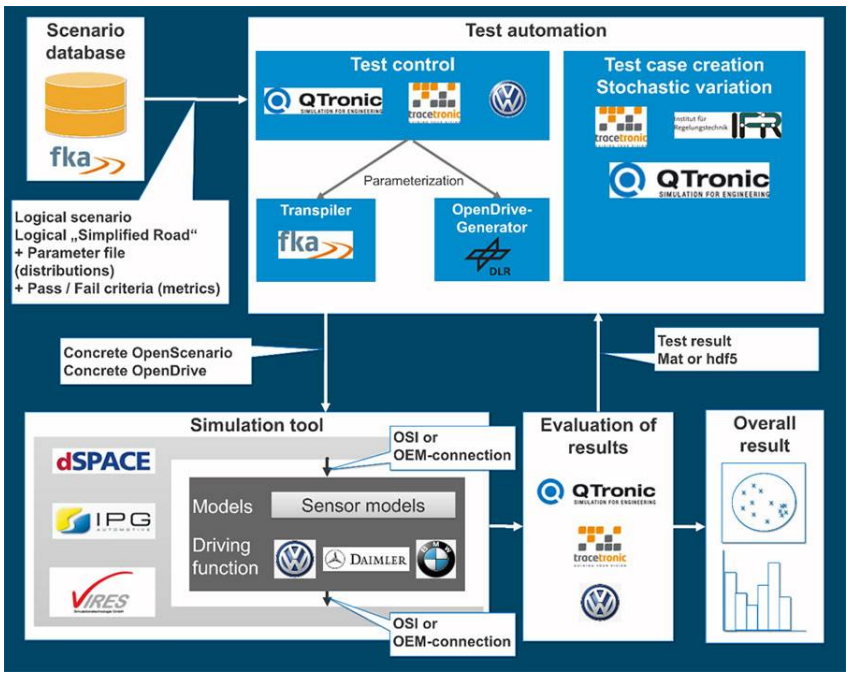

Figure 174: An overview of the different partners with tools and skills in the PEGASUS project

6.3 World

6.3.1 China

In Xian (China), the Joint Laboratory for Internet of Vehicles, Ministry of Education, China Mobile Communications Corporation proposed a research platform and architecture on performance and function testing of V2X in a Closed Test Field. The V2X and cooperative vehicle infrastructure system (CVIS) are essential in the development of Connected and Automated vehicle deployment. Efficient reliable and robust information interactions through V2V, V2I, V2P, and V2N, are critical in reducing traffic accidents and improving traffic efficiency. The complex technical characteristics of V2X for AV and highly reliable service demand of typical V2X applications call for the test needs before the large-scale deployment of CAV. The services, applications, modules, functions involving V2X capabilities must to be systematically tested and evaluated in extreme and boundary conditions of driving and communication environments before being broadly deployed and applied in infrastructures. In order to achieve this objective, [49] propose a modular testing platform for V2X performance and function testing with 2 levels: virtual testing and real controlled environment testing. An overview of this architecture is presented in the figure 175.

6.3.2 USA

Scenic Environment Modeling and Scenario Description Language is proposed by University of Berkeley and in open access on the following website: `http://github.com/BerkeleyLearnVerify/Scenic`.

VerifAI is a toolkit for design and verification of AI-based systems. This toolkit is provided on the websit `http://github.com/BerkeleyLearnVerify/VerifAI`.

United States National Highway Traffic Safety Administration has wrote a report addressing *A Framework for Automated Driving System Testable Cases and Scenarios*. This report

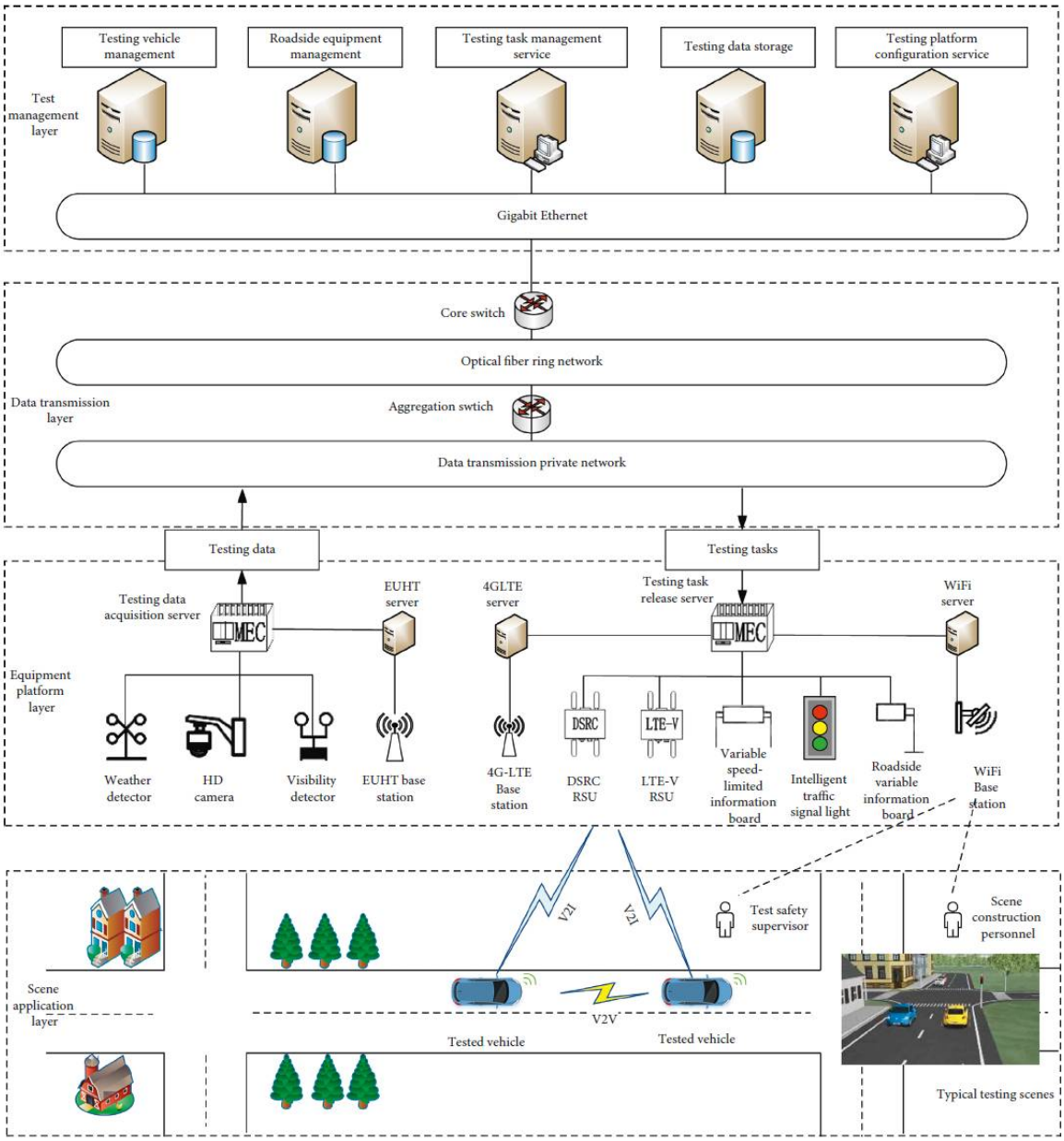

Figure 175: Modular virtual and real testing platform for V2X performance and function testing ([49])

describes a framework for establishing sample preliminary tests for Automated Driving Systems. The focus is on light duty vehicles exhibiting higher levels of automation, where the system is required to perform the full dynamic driving task, including lateral and longitudinal control, as well as object and event detection and response (https://wiki.unece.org/download/attachments/87622238/FRAV-01-14.pdf?api=v2). This report from 2018 proposes:

- Risk analysis to prioritize scenario selection, including frequency of occurrence and severity of outcomes (31 events drawn from NHTSA pre-crash scenario analyses).
- Defines seven generic categories of features (based on 24 conceptual features): L4 Highly Automated Vehicle/Transportation Network Company (TNC), L4 Highly Automated Highway Drive, L4 Highly Automated Low Speed Shuttle, L4 Highly Automated Valet Parking, L4 Highly Automated Emergency Takeover, L3 Conditional Automated Highway Drive, L3 Conditional Automated Traffic Jam Drive.
- Lists traffic events for L3 traffic jam and highway drive systems and for L4 highway drive systems.
- Describes six functional scenarios further broken down into 6-12 concrete scenarios.
- Structures test scenarios based on ODD, tactical/OEDR behaviors, parameter values/measurements, and failure/fallback.
- Distinguishes between failsafe (fallback to ready user or MRM) and fail-operational (degradation/redundancy permitting continued ADS operation).
- Provides ODD checklist to define parameters covering physical infrastructure, operational constraints, objects, environmental conditions, connectivity and (operating) zones.
- Covers modeling and simulation, closed track, and open road test methods.
- Performance based on ADS detection of a safety-critical object or event and response with a stable control action or maneuver that allows the ADS to maintain a safe avoidance distance from all relevant obstacles in the immediate vicinity while respecting applicable driving rules and etiquette to the extent possible.
- Notes AdaptIVe and PEGASUS programs, California PATH minimum behavioral competencies, NIST 4D/RCS Reference Model Architecture for Unmanned Vehicle Systems.

6.3.3 Japan

6.3.3.1 SAKURA project (Safety Assurance KUdos for Reliable Autonomous vehicles)

The SAKURA project (Safety Assurance KUdos for Reliable Autonomous vehicles) is a large scale coordinated initiative funded by the Japanese Ministry of Economy, Trade and Industry (METI) that aims at harmonizing data acquisition, developing research methodologies and coordinating standardization activities through joint efforts by vehicle manufacturers and traffic safety research institutions. (https://www.sakura-prj.go.jp/) SAKURA has for main goals:

- To develop an automated vehicle system safety assurance engineering process that accounts for all foreseeable safety relevant scenarios, with a particular focus on motorways.
- To research and develop the fundamental technology necessary to enable the developed safety assurance process.
- To lead international standardization activities towards the harmonization of scenario structure, parameter range, and safety criteria establishment towards globally accepted common approaches.

In this project 3 levels strongly link to PRISSMA project are addressed: The engineering framework for AD vehicle test scenarios (figure 176), the social contextualization of the engineering framework (figure 177), the test scenario generation process for AD safety assurance (figure 178).

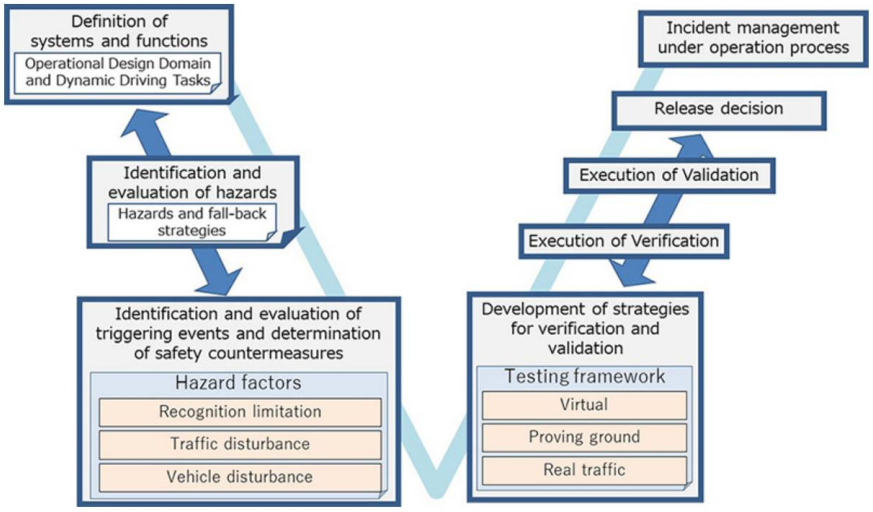

Figure 176: Overall scheme of the safety assurance process (source: SAKURA's web page)

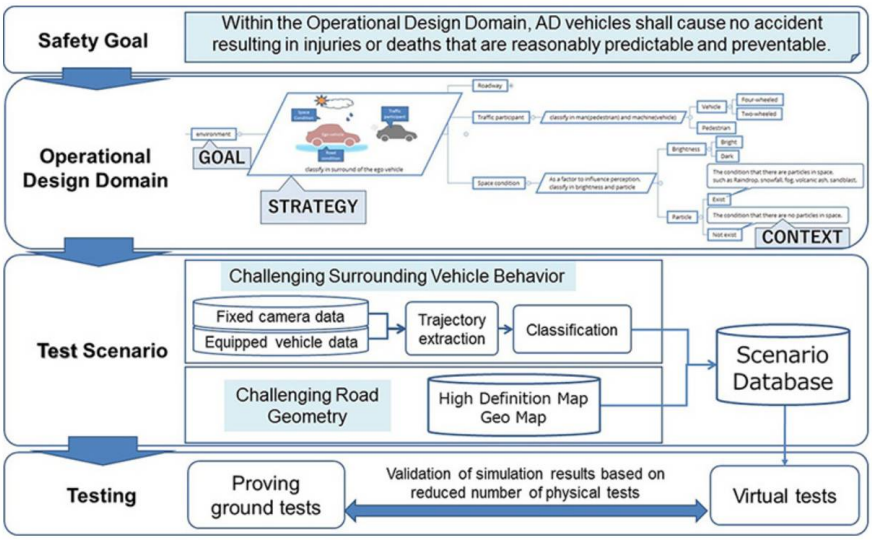

Figure 177: Top-down approach for social contextualization of the engineering framework for AD safety assurance (source: SAKURA's web page)

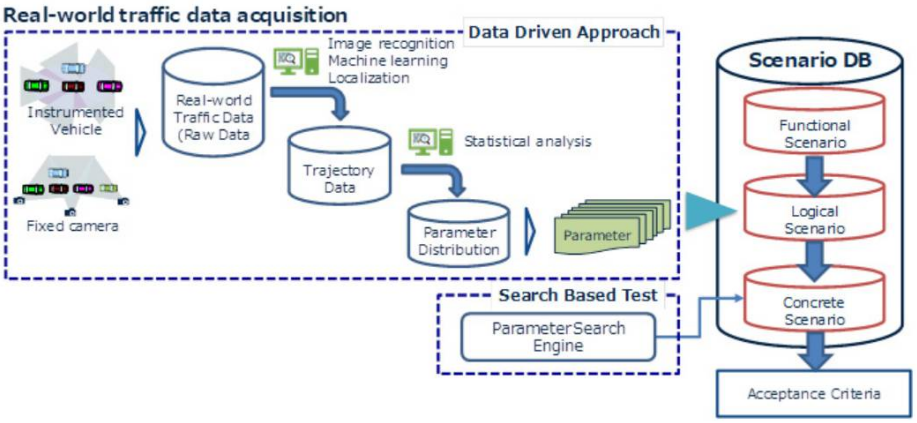

Figure 178: Test Scenario Generation Process for AD Safety Assurance (source: SAKURA's web page)

6.3.3.2 Japan Automobile Manufacturers Association, Inc. (JAMA)

In Japan, in October 2020, the Japan Automobile Manufacturers Association, Inc. (JAMA) which is an AD Safety Assurance Expert Group has proposed a report focused on "Automated Driving Safety Evaluation Framework Ver.1.0" [24]. (http://www.jama-english.jp/publications/Automated_Driving_Safety_Evaluation_Framework_Ver1.0.pdf). In this very interesting report, the main topics about AD system evaluation and evaluation/validation scenarios are addressed:

- Scenario-Based Safety Assurance Process with the safety argumentation scheme (Steps of the V-shaped model)
- Scenario structure with Traffic disturbance scenarios (figure 180), Perception disturbance scenarios (figure 181), and Vehicle Stability Disturbance Scenarios.
- Scenario Database with 3 layers of extraction (figure 182): three elements of driving actions, namely, "perception," "judgement," and "operation" can be systematically structured under the three scenarios of "perception disharmony," "traffic disturbance," and "vehicle movement disturbance,".

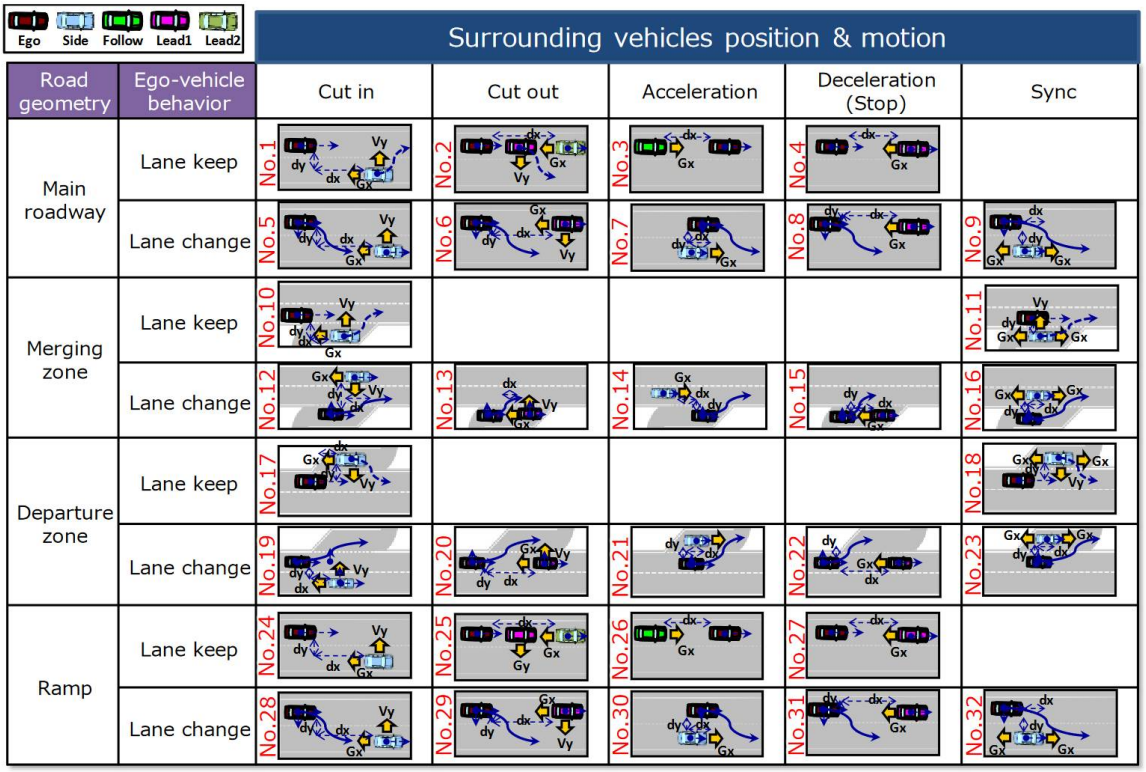

Figure 179: General vehicle traffic disturbance scenarios (source: JAMA's report [24])

A new version of this scenario description involving surrounding traffic participants' position and behaviour is given in the report.

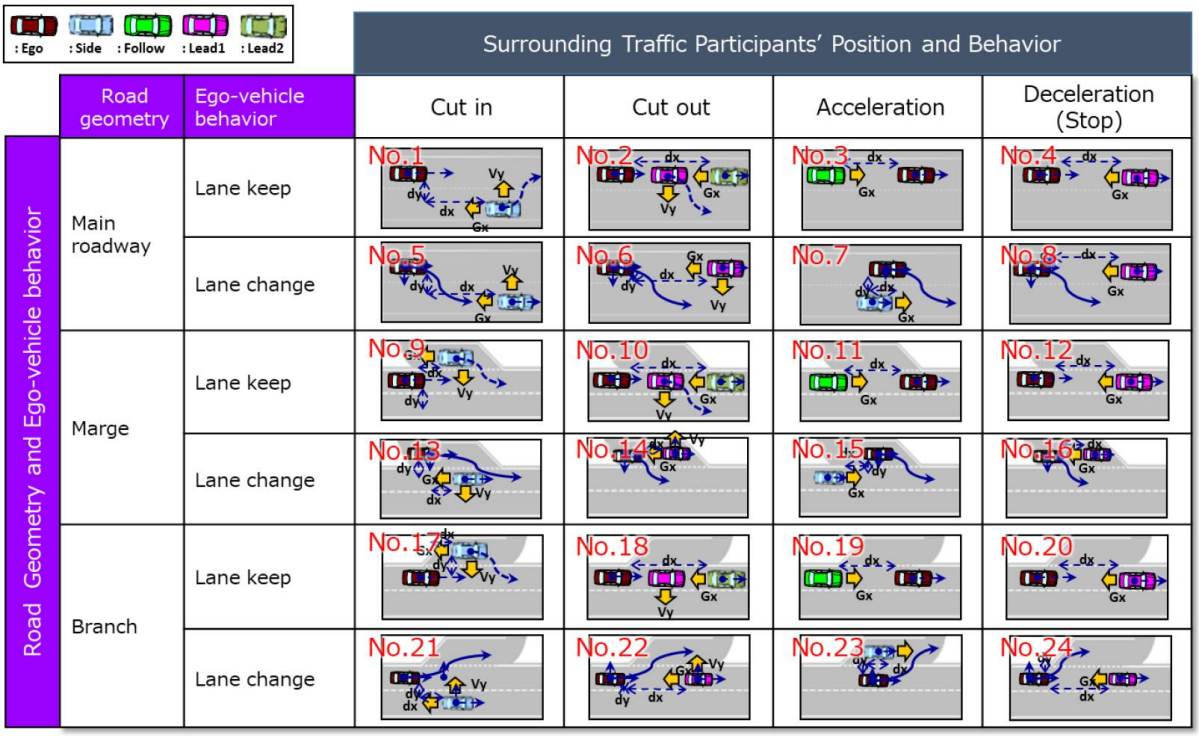

Figure 180: scenario description involving surrounding traffic participants' position and behaviour (source: JAMA's report December 2021)

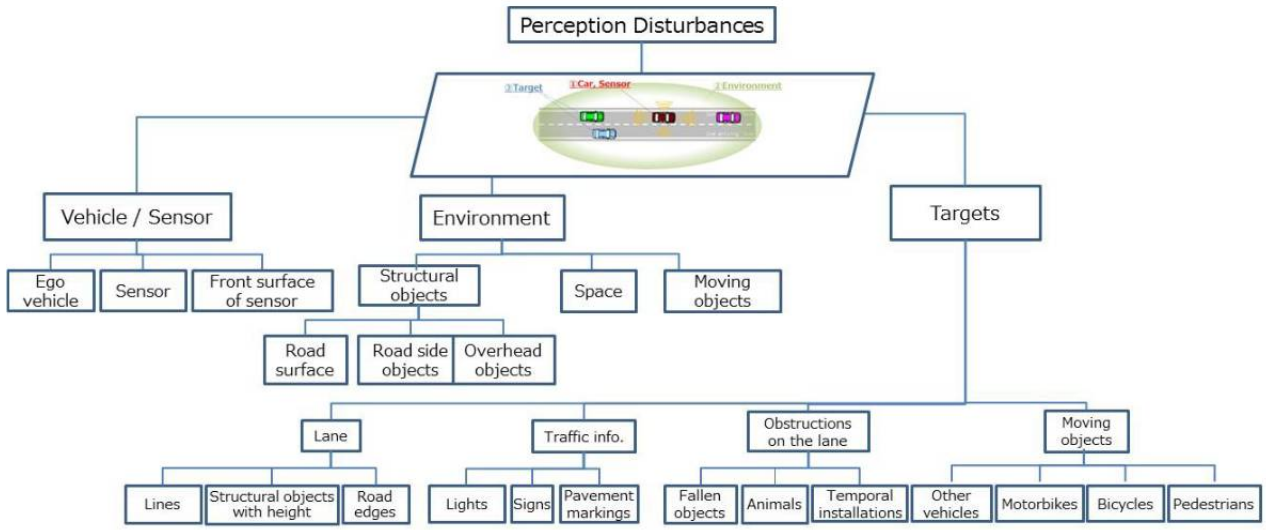

Figure 181: System diagram of perception disturbance factors (source: JAMA's report [24])

Ego vehicle / Surrounding vehicle / Blind spot vehicle		Blind spot vehicle motion									
		Cut-in		Cut-out		Acceleration		Deceleration		Sync	
Road geometry	Ego-vehicle behaviour	Surrounding vehicle motion									
		Lane Keep	Lane Change	Lane Keep	Lane Change	Lane Keep	Lane Change	Lane Keep	Lane Change	L/K	L/C
Main road	Lane Keep	No. 1	No. 2	–	–	No. 3	No. 4	No. 5	No. 6	–	–
	Lane Change	–	No. 7	No. 8	No. 9	No. 10	No. 11	No. 12	No. 13	–	–
Merge zone	Lane Keep	No. 14	No. 15	–	–	–	–	–	–	–	–
	Lane Change	–	No. 16	No. 17	No. 18	No. 19	No. 20	No. 21	No. 22	–	–
Departure zone	Lane Keep	No. 23	No. 24	–	–	–	–	–	–	–	–
	Lane Change	–	No. 25	No. 26	No. 27	No. 28	No. 29	No. 30	No. 31	–	–
Ramp	Lane Keep	No. 32	No. 33	–	–	–	No. 34	–	No. 35	–	–
	Lane Change	–	No. 36	No. 37	No. 38	No. 39	No. 40	No. 41	No. 42	–	–

Figure 182: Perception disturbance scenarios related to blind spots generated by surrounding vehicles (source: JAMA's report [24])

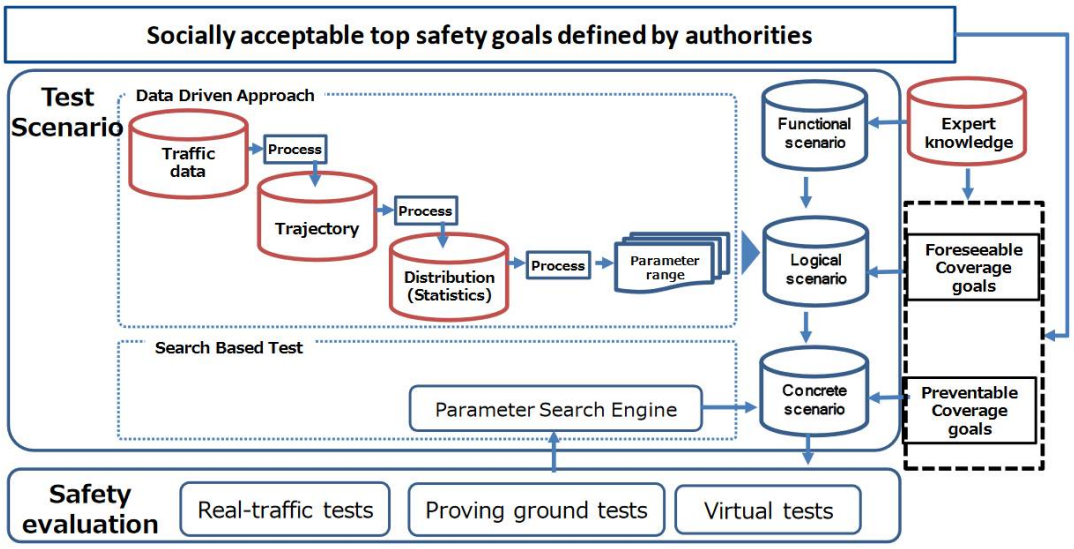

Figure 183: Process of developing and applying data-driven AD safe scenarios (source: JAMA's report [24])

JAMA wrote a new version of its report in December 2021 (https://www.jama.or.jp/english/reports/docs/Automated_Driving_Safety_Evaluation_Framework_Ver2.0.pdf). In this new report, the scenario description is processed in depth in order to covert and to model the main driving situation causing disturbances and in some configuration a near accident or accident situation. It is interesting to mention that this report provide accurate explanation of a majority of the disturbances on the sensors, the perception functions, the vehicles, and the traffic. The operation scope of automatic driving vehicles is defined at the initial stage as the operation design scope (ODD). The contents of the ODD must include, at a minimum, information such as the road type, position on the road, vehicle velocity scope and environmental condition. Moreover, a fallback strategy for transition to outside the ODD boundary must be designed; moreover, the AD system must detect whether it is operating within the defined ODD. The definition of OD must be structured in such a manner as to enable notification to the users, as well as allow them to understand, trust and operate the AD system. The authors affirm that mapping the ODD system and the scenario system as shown in Figure 39, it becomes possible to select the evaluation scenario following the ODD range.

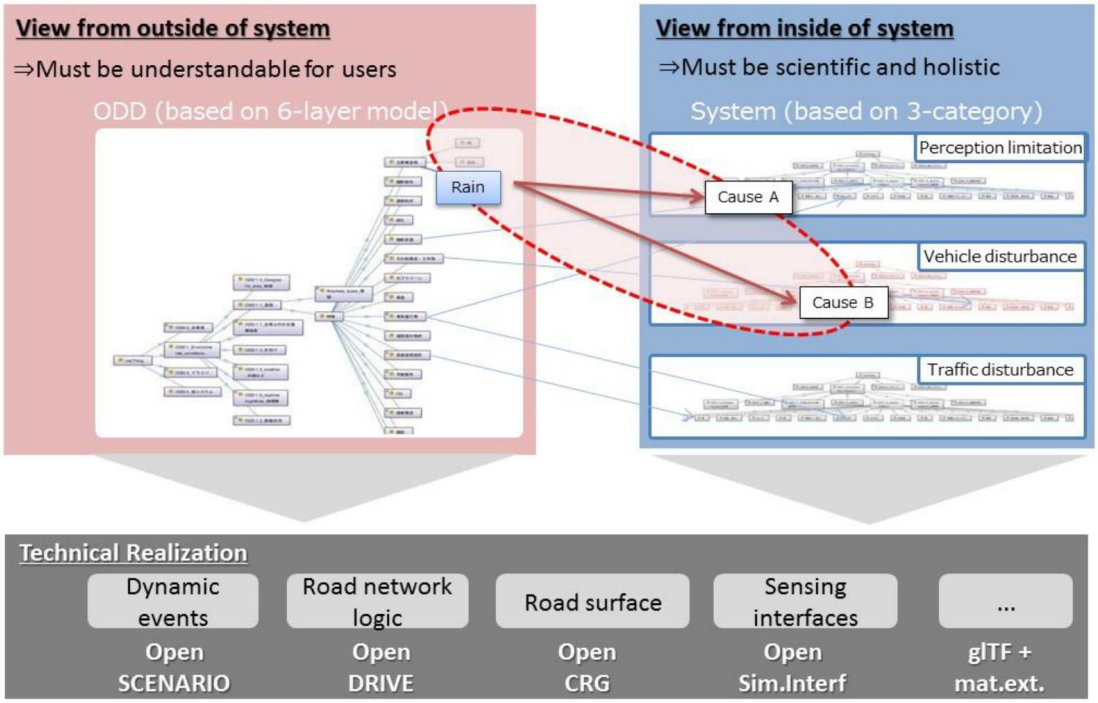

Figure 184: ODD scenario classification and relationship diagram of the system level classification based on the three category scenario level (source: JAMA's report 2021)

6.3.4 World standardization activities around AI-bases systems

Standards development organizations (SDO) are earnestly working to develop standards in AI, including the safety and trustworthiness of AI systems. Some AI standardization efforts are enumerated below:

- **ISO: Expert group to carry out standardization activities for AI**: The subcommittee (SC) 42 is part of the joint technical committee ISO/IEC JTC 1, and has a working group on foundational standards to provide a framework and a common vocabulary, and several other working groups on computational approaches and characteristics of AI systems, the trustworthiness, the use cases, the applications, and the big data processing aspects. (https://www.iso.org/committee/6794475.html)
- **ISO 34502 Road vehicles — Test scenarios for automated driving systems — Scenario based safety evaluation framework** (https://www.iso.org/standard/78951.html). This document provides guidance and a state-of-the-art engineering framework for ADS test scenarios and a scenario-based safety evaluation process. The focus is about Highway AD system (upper Level 3/SAE definition). This document specifies the state-of-the-art holistic coverage test scenarios and scenario-based safety assurance process within the product development. This document provides Scenario definition (Based-on Hazardous scenario), Scenario structure, and Safety target.
- **ISO 22736 / SAE AWI PAS (Under Development)**: Intelligent transport systems — Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles (https://www.iso.org/standard/73766.html)
- **The IEEE P7000 series of projects**: These project are part of the IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems, launched in 2016. In these

project, “Fail-Safe Design of Autonomous and Semi-Autonomous Systems” is one of 13 standards addressed in the series. (https://standards.ieee.org/project/7009.html)

- **ANSI/UL 4600**: A released standard has been done on “Standard for Safety for the Evaluation of Autonomous Products”. (https://ul.org/UL4600)
- **The SAE G-34**: Dedicated to Artificial Intelligence in Aviation, this Committee is responsible for creating and maintaining SAE Technical Reports, including standards, on the implementation and certification aspects related to AI technologies inclusive of any on or off-board system for the safe operation of aerospace systems and aerospace vehicles. (https://www.sae.org/works/committeeHome.do?comtID=TEAG34)
- **SAE International, Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles J3016**: This SAE Recommended Practice describes motor vehicle driving automation systems that perform part or all of the dynamic driving task (DDT) on a sustained basis. It provides a taxonomy with detailed definitions for six levels of driving automation, ranging from no driving automation (level 0) to full driving automation (level 5), in the context of motor vehicles (hereafter also referred to as “vehicle” or “vehicles”) and their operation on roadways. These level definitions, along with additional supporting terms and definitions provided herein, can be used to describe the full range of driving automation features equipped on motor vehicles in a functionally consistent and coherent manner. (https://www.sae.org/standards/content/j3016_201806/)
- **PAS 1883:2020, Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) - Specification**: Wrote by The British Standards Institution (BSI). The PAS introduces requirements for specifying an ODD to enable the safe deployment of ADS. The document is intended for trialling organizations developing safety cases for automated vehicle trials and testing. Sponsored by Centre for Connected Autonomous Vehicles and the UK Department for Transport. (https://www.bsigroup.com/en-GB/CAV/pas-1883/)

6.3.5 ASAM Organization

Association for Standardization of Automation and Measuring System (ASAM) is an international non-profit organization that aims at :

- promoting actively standardization within the automotive industry.
- providing a neutral collaborative platform. The goal is to identify common, non-competitive challenges and solve them together .
- defining interfaces, protocols, file formats and data models for development and testing

ASAM organizes their activity around 7 core activities :

- 1. **Measurement & Calibration Standards** for working with ECU variables and parameters.
- 2. **Diagnostics Standards** for describing and testing the diagnostic subsystems of ECUs.

- 3. **ECU Networks** Standards for describing and testing ECU networks.
- 4. **Software Development** Standards supporting the ECU software development process.
- 5. **Test Automation** Standards for working with test systems.
- 6. **Data Management** & Analysis Standards for storing, retrieving and analyzing large amounts of data captured during simulation, testing, production and the operation of vehicles.
- 7. **Simulation** Standards that support the automotive industry in furthering the state of autonomous driving, especially with respect to (virtual) validation and verification.

The one that is of interest for us is the domain **Simulation Standards**. This domain is organised through projects that are listed that all start with the prefix Open:

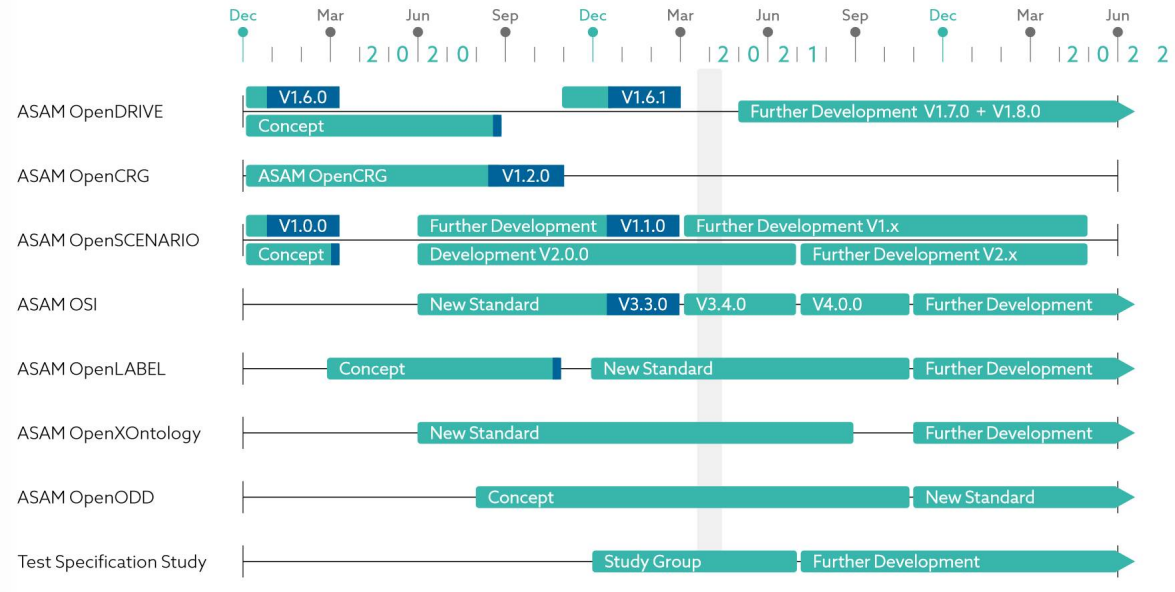

Figure 185: ASAM Activities in the simulation domain: OpenX projects (https://www.asam.net/fileadmin/News/Brochures/ASAM_SIM-Guide_Online.pdf)

The goal of this state of the art document is not to make a exhaustive deep-dive of all existing standard in Simulation domain but rather to see some already applicable (or in development) standards that would directly or indirectly be of interest for PRISSMA.

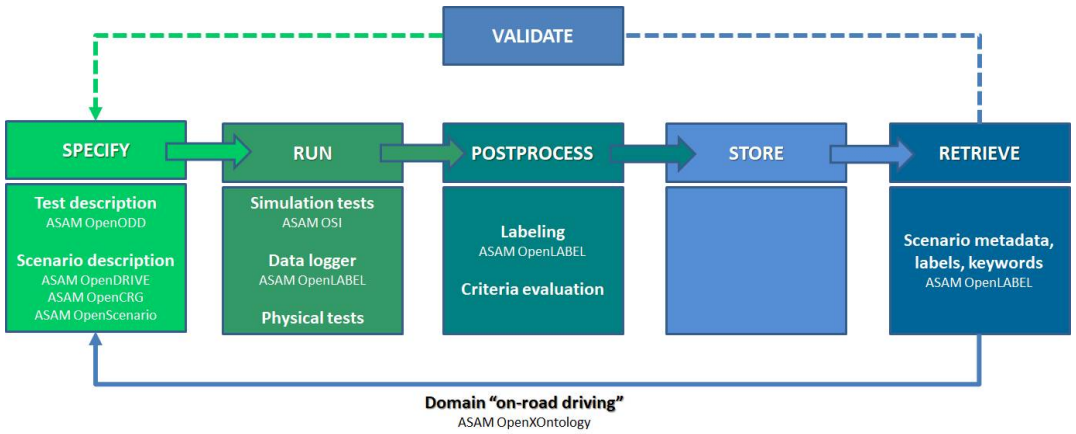

Figure 186: Scenario-based testing with ASAM OpenX (https://www.asam.net/fileadmin/News/Brochures/ASAM_SIM-Guide_Online.pdf)

6.3.5.1 Standards and ASAM OpenX

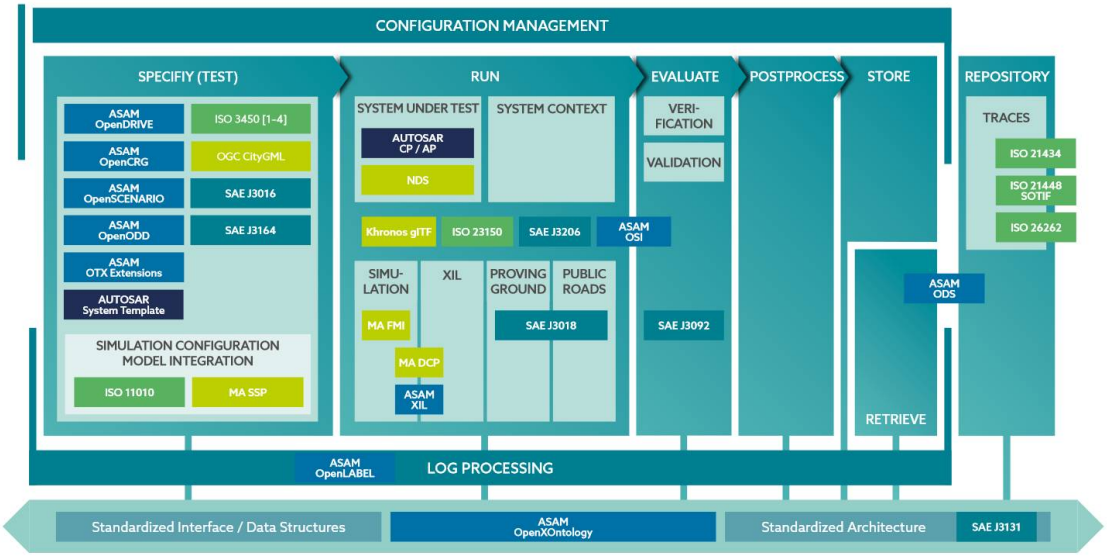

Figure 187: Overview of a generic architecture of a Simulation Environment with links and relation with standards

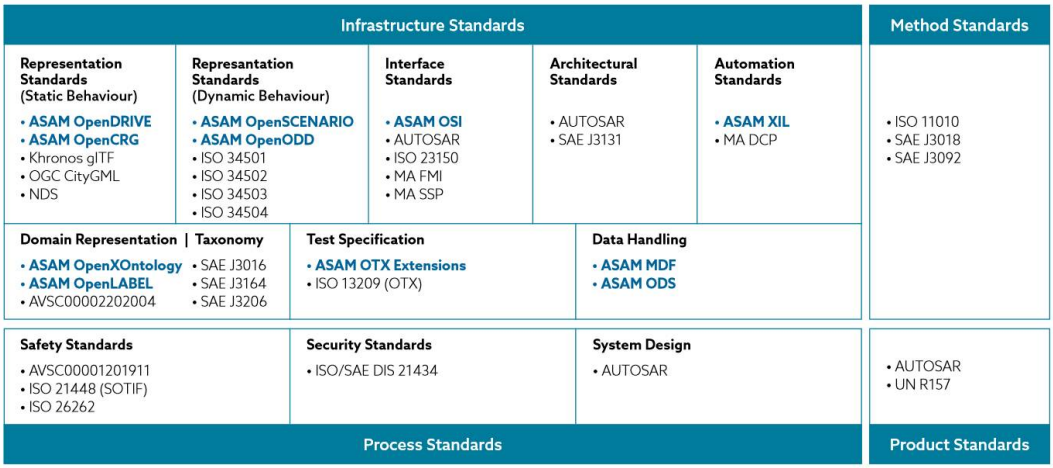

Figure 188: Overview on Standards for Simulation Across Organizations

6.3.5.2 OpenDRIVE ®

The ASAM OpenDRIVE format provides a common base for describing road networks with extensible markup language (XML) syntax, using the file extension xodr. The data that is stored in an ASAM OpenDRIVE.

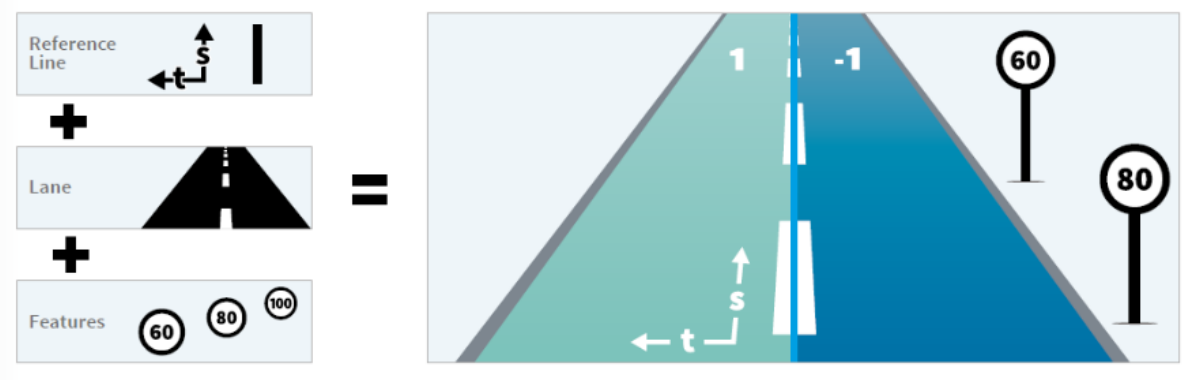

Figure 189: Illustration of some ASAM OpenDRIVE elements

The ASAM OpenDRIVE road network is modelled along the reference line, which is the core piece of every road. Roads, lanes, incl. their elevation profiles are all attached to the reference line.

In ASAM OpenDRIVE several roads form a road network and can be connected. ASAM OpenDRIVE can be seen as a construction kit of different road sections. The overall road network is composed of individual sections interconnected with each other.

Today ASAM OpenDRIVE is a quite well spread standard in the industry and all Simulation SW are able to parse an OpenDRIVEC file (.xodr) to create the equivalent road network in the Simulation SW.

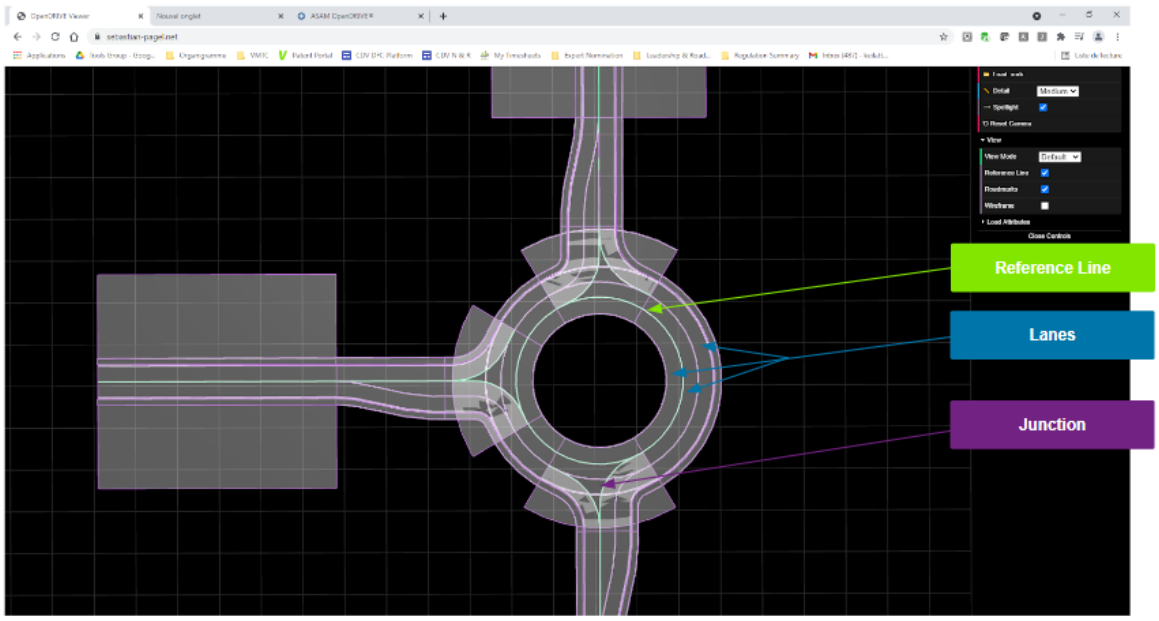

Figure 190: Illustration an Open DRIVE file in a simple viewer

6.3.5.3 OpenSCENARIO ®

ASAM OpenSCENARIO defines the dynamic content of the world, for example, behavior of traffic participants and how these are expected to interact with each other and the environment.

ASAM currently actively develops two parallel versions of it. The so called 1.x pipeline, and the 2.x pipeline (first version released July 2022). ASAM OpenSCENARIO 1.x defines a data model and a derived file format for the description of scenarios. The primary use-case of ASAM OpenSCENARIO 1.x is to describe complex, synchronized Maneuvers that involve multiple instances of Entity, like Vehicles, Pedestrians and other traffic participants.

The description of a scenario may be based on driver Actions, for example, performing a lane change, or on instances of Trajectory, for example, derived from a recorded driving Maneuver. The standard provides the description methodology for scenarios by defining hierarchical elements from which scenarios, their attributes and relations are constructed.

The data for scenario descriptions in ASAM OpenSCENARIO is organized in a hierarchical structure and serialized in an XML file format with the extension xosc.

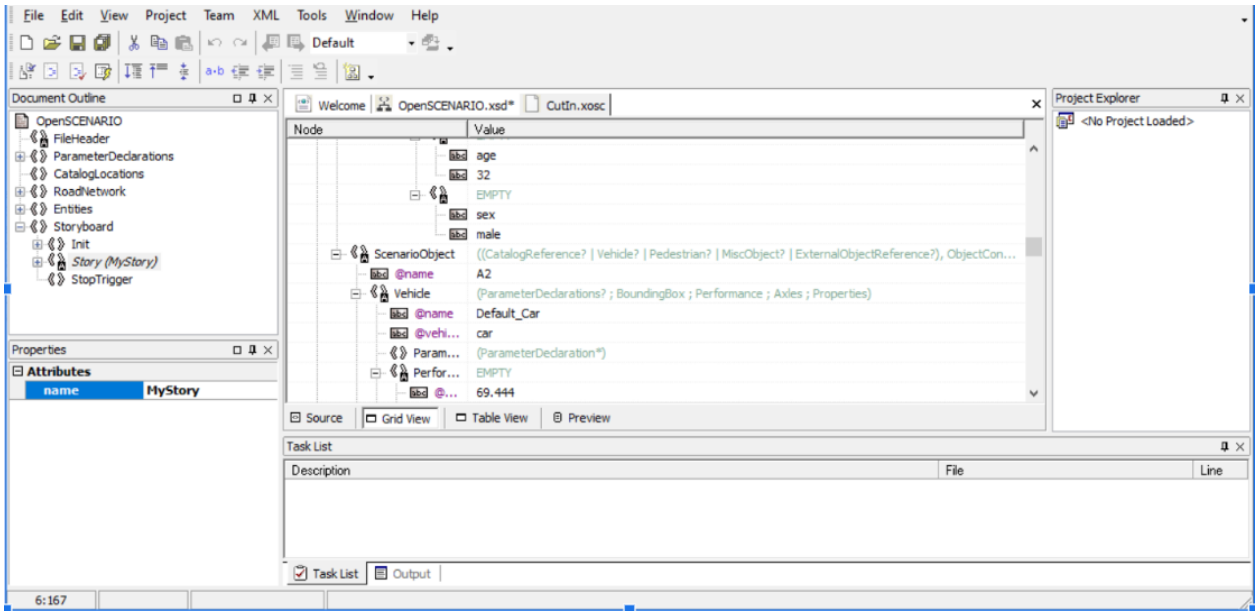

Figure 191: Illustration an Open SCENARIO file in a xml viewer

OpenSCENARIO 2.0 is proposed to be founded on the concept of a domain-specific language, that should support all levels of scenario description, from the very abstract to the very concrete in a suitable way.

In comparison to OpenSCENARIO 1.0, a more detailed set of actions and attributes for the relevant simulation models shall be defined to allow for a more comprehensive scenario description and to improve exchangeability. This is addressed by the introduction of a domain model. The foundational concept of OpenSCENARIO 2.0 is to establish a domain specific language of a declarative nature.

- A declarative language describes what should happen on scenario execution (including the required parameterization/variation), rather than how to do it.
- A declarative language can also have a dual interpretation, i.e. provide a single scenario

description which can be used to describe both how to make it and how to monitor that it indeed happened..

This is important if we want to condition some operation on the fact that some other scenario has happened before, without having to describe in detail how to cause that scenario. Here some main functionality of such DSL language

```
s: speed
v1: car
```

Figure 192: parameter field - defines something which you can change/influence in the invocation:

```
!ttcat_end: time = sample(get_time_to_collision(), @end)
```

Figure 193: variable field - defines a location to update during computations (e.g. to hold KPIs)

```
keep(v1.color != v2.color)
speed([10..15]kph, faster_than: v1)
set_map("my_map.xodr")
```

Figure 194: modifier/constraint - modify (influence) scenario behavior

OpenSCENARIO is still a young standard, especially OpenSCENARIO v2.0. Today AD Simulation SW are not able to parse those specification. It will require some time before the industry players being able to implement specific solver to create simulable test run from Open SCENARIO v1.x / v2.0 . An opportunity that can be leveraged is to use third party tool between the specification and the simulation SW to handle this.

6.3.6 ISO Standards :

A number of standards on the use of simulation for VAs have recently been published by ISO, addressing both the use of simulation for the validation of VAs and the use of software tools for VAs/

- **ISO 11010 : Simulation model classification.** A systematic framework has been created that facilitates the definition of the requirements of simulation models for certain applications and driving manoeuvres in a standardized manner. the first part deals with vehicle dynamics models, the second with perception sensor models and the last with tire models.
- **ISO 19364 : Vehicle dynamic simulation and validation** This standard deals with how to compare vehicle simulation results with ground truth calibration data. The aim is to validate the simulation tool but this remains limited to the scop of use (ground truth corresponds to the case specified in standard 4138 i.e. steady-state circular driving tests or the Slowly Increasing Steer Test).

REFERENCES

[1] M. Gradu and D. Heeren. (2021, August) Designing and assessing vehicle safety functions with a use case approach. SEA report. [Online]. Available: https://www.sae.org/news/2021/08/designing-and-assessing-vehicle-safety-functions-with-a-use-case-approach

[2] H. Belani, M. Vuković, and Željka Car, "Requirements engineering challenges in building ai-based complex systems," in Sixth International Workshop on Artificial Intelligence for Requirements Engineering (AIRE'19), 2019. [Online]. Available: https://arxiv.org/abs/1908.11791

[3] M. Webster, D. Wester, D. Araiza-Illan, C. Dixon, K. Eder, M. Fisher, and A. Pipe, "A corroborative approach to verification and validation of human-robot teams," Journal of Robotics Research, vol. 39, no. 1, 2020. [Online]. Available: https://arxiv.org/pdf/1608.07403.pdf

[4] J. Gao, X. Chunli, and T. Chuanqi, "Big data validation and quality assurance – issues, challenges and needs," in 32016 IEEE Symposium on Service-Oriented System Engineering (SOSE), Oxford, UK. IEEE, 2016, pp. 433–441. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7473058

[5] S. Hakuli and M. Krug, "Virtuelle integration," in Handbuch Fahrerassistenzsysteme, 2015, pp. 128–138. [Online]. Available: https://doi.org/10.1007/978-3-658-05734-3_8.

[6] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in 2015 IEEE 18th International Conference on Intelligent Transportation Systems, 2015, pp. 982–988.

[7] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," 2018 IEEE Intelligent Vehicles Symposium (IV), pp. 1821–1827, 2018.

[8] L. Raffaelli, F. Vallée, G. Fayolle, P. De Souza, X. Rouah, M. Pfeiffer, S. Géronimi, F. Pétrot, and S. Ahiad, "Facing adas validation complexity with usage oriented testing," arXiv preprint arXiv:1607.07849, 2016.

[9] J. Ma, X. Che, Y. Li, and E. M.-K. Lai, "Traffic scenarios for automated vehicle testing: A review of description languages and systems." Machines, vol. 9, no. 342, December 2021. [Online]. Available: https://www.mdpi.com/2075-1702/9/12/342

[10] W. Chen, "Formal modeling and automatic generation of test cases for the autonomous vehicle." Thesis for the degree of Licentiate of Engineering in Machine and Vehicle Systems, Université Paris-Saclay (UVSQ), École doctorale n○580 : sciences et technologies de l'information et de la communication (STIC), Spécialité de doctorat: Informatique, Versailles, France., 2020.

[11] Z. Zhong, Y. Tang, Y. Zhou, V. de Oliveira Neves, Y. Liu, and B. Ray, "A survey on scenario-based testing for automated driving systems in high-fidelity simulation," Preprint Arxiv, 2021. [Online]. Available: https://arxiv.org/pdf/2112.00964.pdf

[12] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, "Survey on scenario-based safety assessment of automated vehicles," IEEE Access, vol. 8, pp. 87 456–87 477, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.2993730

[13] J. R. Ward, G. Agamennoni, S. Worrall, and E. M. Nebot, "Vehicle collision probability calculation for general traffic scenarios under uncertainty," 2014 IEEE Intelligent Vehicles Symposium Proceedings, pp. 986–992, 2014.

[14] F. Jiménez, J. Naranjo, and F. García, "An improved method to calculate the time-to-collision of two vehicles," International Journal of Intelligent Transportation Systems Research, vol. 11, no. 1, pp. 34–42, 2013.

[15] A. Paigwar, E. Baranov, A. Renzaglia, C. Laugier, and A. Legay, "Probabilistic collision risk estimation for autonomous driving: Validation via statistical model checking," in IEEE Intelligent Vehicles Symposium (IV). IEEE, 2020, pp. 737–743.

[16] H. HUANG, J. WANG, C. FEI, X. ZHENG, Y. YANG, J. LIU, X. WU, and Q. XU1, "A probabilistic risk assessment framework considering lane-changing behavior interaction," Science China Information Sciences, vol. 63, August 2020.

[17] J. Leroy, D. Gruyer, O. Orfila, and N.-E. E. Faouzi, "Five key components-based risk indicators ontology for the modelling and identification of critical interaction between human driven and automated vehicles," in 3rd IFAC Conference on Cyber-Physical & Human-Systems (CPHS), Shanghai, China, Dec. 3-5., 2020, 2020.

[18] H. Pan, M. Kokkolaras, G. Hulbert, M. Castanier, and D. Lamb, "Model validation for simulations of vehicle systems," in MODELING & SIMULATION, TESTING AND VALIDATION (MSTV) MINI-SYMPOSIUM, AUGUST 14-16, MICHIGAN, 2012.

[19] F. Kluck and F. Wotawa, "Using ontologies for test suites generation for automated and autonomous driving functions," in Proceedings of the 29th IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW2018), Memphis, TN, USA (2018). IEEE, 2018.

[20] L. Myllyaho, M. Raatikainen, T. Männistö, T. Mikkonen, and J. Nurminen, "Systematic literature review of validation methods for ai systems," The Journal of Systems & Software, vol. 181, July 2021.

[21] L. Li, W.-L. Huang, Y. Liu, N.-N. Zheng, and F.-Y. Wang, "Intelligence testing for autonomous vehicles: A new approach," IEEE Transactions on Intelligent Vehicles, vol. 1, no. 2, pp. 158–166, 2016.

[22] T. Bokc, M. Maurer, and G. Farber, "Validation of the vehicle in the loop (vil); a milestone for the simulation of driver assistance systems," in 2007 IEEE Intelligent vehicles symposium. IEEE, 2007, pp. 612–617.

[23] M. Revilloud, D. Gruyer, and E. Pollard, "Generator of road marking textures and associated ground truth. applied to the evaluation of road marking detection," in IEEE ITSC 2012, Anchorage, AK, USA ; 16 Sep - 19 Sep 2012, 2012.

[24] JAMA. (2020, October) Automated driving safety evaluation framework ver.1.0. Report from Japan Automobile Manufacturers Association, Inc. (JAMA), AD Safety Assurance Expert Group. [Online]. Available: http://www.jama-english.jp/publications/Automated_Driving_Safety_Evaluation_Framework_Ver1.0.pdf

[25] R. Song, J. Wetherall, S. Maskell, and J. F. Ralph, "Wheather effects on obstacle detection for autonomous car," 2020. [Online]. Available: https://livrepository.liverpool.ac.uk/id/eprint/3079344

[26] Real time rendering of heterogenous fog based on the graphics hardware acceleration. CENTRAL EUROPEAN SEMINAR ON COMPUTER GRAPHICS, 2004. [Online]. Available: https://old.cescg.org/CESCG-2004/papers/34_ZdrojewskaDorota.pdf

[27] W. Li, C. Pan, R. Zhang, J. Ren, and al, "Aads: Augmented autonomous driving simulation using data-driven algorithms," Science Robotics, vol. 4, pp. 1–12, March 2019. [Online]. Available: https://www.science.org/doi/10.1126/scirobotics.aaw0863

[28] T. Genevois, J.-B. Horel, A. Renzaglia, and C. Laugier, "Augmented reality on lidar data: Going beyond vehicle-in-the-loop for automotive software validation," in 2022 IEEE Intelligent Vehicles Symposium (IV), 2022, pp. 971–976.

[29] D. Z. S. J. S. W. Jin Fang, Xinxin Zuo and L. Zhang, "Lidar-aug: A general rendering-based augmentation framework for 3d object detection," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE/CVF, 2021. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2021/papers/Fang_LiDAR-Aug_A_General_Rendering-Based_Augmentation_Framework_for_3D_Object_Detection_CVPR_2021_paper.pdf

[30] T. B. B. C. J. Vargas Rivero, J.R.; Gerbich, "Data augmentation of automotive lidar point clouds under adverse weather situations," Sensors, vol. 21, June 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/13/4503

[31] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," IEEE Access, vol. 8, 2020. [Online]. Available: https://arxiv.org/pdf/1906.05113.pdf

[32] D. Feng, C. Haase-Schutz, L. Rosenbaum, H. Hertlein, C. Glaser, F. Timm, W. Wiesbeck, and K. Dietmayer, "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets,methods, and challenges," IEEE Transactions on Intelligent Transportation Systems, vol. 99, pp. 1–20, February 2020. [Online]. Available: https://arxiv.org/abs/1902.07830

[33] P. Ji, R. Li, Y. Xue, Q. Dong, L. Xiao, and R. Xue. (2021, June) Perspective, survey and trends: Public driving datasets and toolsets for autonomous driving virtual test. [Online]. Available: https://arxiv.org/pdf/2104.00273.pdf

[34] D. Gruyer, R. Belaroussi, X. Li, B. Lusetti, M. Revilloud, and S. Glaser, "Persee: A central fusion sensors electronic control unit for the development of perception-based adas," in 14th IAPR Conference on Machine Vision Applications (MVA 2015), 18-22 may 2015, Tokyo. IAPR, May 2015. [Online]. Available: http://www.mva-org.jp/Proceedings/2015USB/papers/10-04.pdf

[35] B. Vanholme, D. Gruyer, S. Glaser, and S. Mammar, "A legal safety concept for highly automated driving on highways," in 2011 IEEE Intelligent Vehicles Symposium (IV), 2011, 2011, pp. 563–570.

[36] M. Rahman, M. Chowdhury, Y. Xie, and Y. He, "Review of microscopic lane-changing models and future research opportunities," IEEE transactions on intelligent transportation systems, vol. 14, pp. 1942–1956, 2013.

[37] M. Saifuzzaman and Z. Zheng, "Incorporating human-factors in car-following models: A review of recent developments and research needs," Transportation Research Part C, vol. 48, pp. 379–403, October 2014.

[38] D. Jiaa, J. Suna, A. Sharmaa, Z. Zhenga, and B. Liub, "Integrated simulation platform for conventional, connected and automated driving: A design from cyber–physical systems perspective," Transportation Research Part C: Emerging technologies, vol. 124, March 2021.

[39] K. Miura, S. Tokunaga, Y. Horita, Y. Oda, and T. Azumi1, "Cosam: Co-simulation framework for ros-based self-driving systems and matlab/simulink," Journal of Information Processing, vol. 29, p. 227–235, March 2021.

[40] Q. Chao, Z. Deng, Y. Xiao, D. He, Q. Miao, and X. Jin, "Dictionary-based fidelity measure for virtual traffic," IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, vol. 26, pp. 1490–1501, March 2020.

[41] W. Li, DavidWolinski, and M. C. Lin, "City-scale traffic animation using statistical learning and metamodel-based optimization," ACM Transactions on Graphics, vol. 36, November 2017.

[42] A. Kondyli, V. C. Kummetha, and E. G. Chrysikou, "Modeling driver behavior and aggressiveness using biobehavioral methods – phase ii," Report on Research Sponsored by Mid-America Transportation Center, Tech. Rep., January.

[43] M. S. Young, K. A. Brookhuis, C. D. Wickens, and P. A. Hancock, "State of science: mental workload in ergonomics." Ergonomics, vol. 58, no. 1, pp. 1–17, 2015. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/00140139.2014.956151?journalCode=terg20

[44] B. Schlager, S. S. Muckenhuber, S. Schmidt, and al, "State-of-the-art sensor models for virtual testing of advanced driver assistance systems/autonomous driving functions," SAE International Journal of Connected and Automated Vehicles, vol. 3, pp. 233–261, October 2020. [Online]. Available: https://www.researchgate.net/publication/346794909_State-of-the-Art_Sensor_Models_for_Virtual_Testing_of_Advanced_Driver_Assistance_SystemsAutonomous_Driving_Functions

[45] E. Commision, "Regulation 2022/1426/eu - type-approval of the automated driving system (ads) of fully automated vehicles," 2022. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32022R1426

[46] Z. Saigol, R. Myers, A. Peters, and T. Edwards, "Musicc: An open catalogue for cav certification scenarios," in Virtual ITS European Congress, 2020. [Online]. Available: http://www.zeynsaigol.com/ITS2020MUSICC-Overview.pdf

[47] I. Urbieta, M. Nieto, M. García, and O. Otaegui, "Design and implementation of an ontology for semantic labeling and testing: Automotive global ontology (ago)," Applied Science, vol. 11, August 2021.

[48] N. Wagener, P. Weißensteiner, J.-B. Coget, L. Eckstein, and A. Bracquemond, "Common methodology for data-driven scenario-based safety assurance in the headstart project," in Virtual ITS European Congress, 9-10 November 2020, November 2020.

[49] R. Wang, X. Zhang, Z. Xu, X. Zhao, and X. Li, "Research on performance and function testing of v2x in a closed test field," Journal of Advanced Transportation, vol. 7, no. 5, pp. 643–653, August 2021.

[50] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in International Conference on Learning Representations (ICLR 2015), May 2015. [Online]. Available: https://arxiv.org/pdf/1412.6572.pdf

[51] D. Gunning, "Explainable artificial intelligence (xai)," in In IJCAI 2016 Workshop on Deep Learning for Artificial Intelligence (DLAI), July 2016.

[52] S. A. Seshia, D. Sadigh†, and S. S. Sastry. (2020, July) Towards verified artificial intelligence. [Online]. Available: https://arxiv.org/pdf/1606.08514.pdf

[53] A. B. Arrieta, N. Díaz-Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," Information Fusion, vol. 58, p. 82–115, June 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253519308103

[54] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. (2013, December) Intriguing properties of neural networks. [Online]. Available: https://www.researchgate.net/publication/259440613_Intriguing_properties_of_neural_networks

[55] J. M. Faria, "Machine learning safety: An overview," in Proceedings of the 26th Safety-Critical Systems Symposium, York, UK, February 2018. [Online]. Available: https://www.researchgate.net/publication/320567319_Machine_Learning_Safety_An_Overview

[56] R. V. Yampolskiy. (2020, July) On controllability of artificial intelligence. Report from Speed School of Engineering, Computer Science and Engineering, University of Louisville. [Online]. Available: https://philpapers.org/archive/YAMOCO.pdf

[57] T. Miller. (2018, August) Explanation in artificial intelligence: Insights from the social sciences. Report from School of Computing and Information Systems, University of Melbourne, Melbourne, Australia. [Online]. Available: https://arxiv.org/pdf/1706.07269.pdf

[58] X. Xie, J. W. K. Hob, C. Murphy, G. Kaiser, B. Xu, and T. Y. Chen, "Testing and validating machine learning classifiers by metamorphic testing," Journal of System Softwares, vol. 84, no. 1, p. 544–558, April 2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3082144/pdf/nihms260635.pdf

[59] E. Breck, C. Shanqing, E. Nielsen, M. Salib, and D. Sculley, "What's your ml test score? a rubric for ml production systems." in 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona Spain, 2016. [Online]. Available: https://www.eecs.tufts.edu/~dsculley/papers/ml_test_score.pdf

[60] J. Uesato, B. O'Donoghue, A. van den Oord, and P. Kohli, "Adversarial risk and the dangers of evaluating against weak attacks," in 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018., 2018. [Online]. Available: https://arxiv.org/pdf/1802.05666.pdf

[61] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks." in International Conference on Computer Aided Verification (CAV 2017), vol. 10426. Springer, 2017, pp. 3–29. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-63387-9_1

[62] C. Picardi, C. Paterson, R. D. Hawkins, R. Calinescu, and I. Habli, "Assurance argument patterns and processes for machine learning in safety-related systems," in Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI 2020), February 2020, pp. 23–30.

[63] PFA. (2020) Automated driving safety validation: proposals from the french eco-system. [Online]. Available: https://www.ecologie.gouv.fr/sites/default/files/2020%2001%2009%20-%20autonomous%20driving%20-%20safety%20validation%20-%20french%20views%20-%20Vdef.pdf

[64] JORF. (2021, Juin) Décret n° 2021-873 du 29 juin 2021 portant application de l'ordonnance n° 2021-443 du 14 avril 2021 relative au régime de responsabilité pénale applicable en cas de circulation d'un véhicule à délégation de conduite et à ses conditions d'utilisation. [Online]. Available: https://www.legifrance.gouv.fr/loda/id/LEGIARTI000043734793/2021-07-02/

[65] J. Wishart, S. Como, U. Forgione, J. Weast, L. Weston, A. Smart, G. Nicols, and S. Ramesh, "Literature review of verification and validation activities of automated driving systems," SAE International Journal of CAV, vol. 3, 2020. [Online]. Available: https://www.researchgate.net/publication/349067126

[66] F. Batsch, S. Kanarachos, M. Cheah, R. Ponticelli, and M. Blundell, "A taxonomy of validation strategies to ensure the safe operation of highly automated vehicles," JOURNAL OF INTELLIGENT TRANSPORTATION SYSTEMS, March 2020.

[67] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, T. Weißgerber, K. Bengler, R. Bruder, F. Flemisch, and H. Winner, "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," IET Intelligent Transport Systems, vol. 8, no. 3, pp. 183–189, 2014. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-its.2012.0188

[68] G. Bagschik, T. Menzel, and M. Maure, "Ontology based scene creation for the development of automated vehicles," in IEEE Intelligent Vehicles Symposium (IV). IEEE, April 2018. [Online]. Available: https://arxiv.org/pdf/1704.01006.pdf

[69] Technical Committee ISO/TC 22 and Subcommittee SC 32, "Road vehicles safety of the intended functionality," International Organization for Standardization, techreport ISO/PAS 21448:2019, 2019. [Online]. Available: https://www.iso.org/standard/70939.html

[70] H. Hungar, F. Köster, and J. Mazzega, "Test specifications for highly automated driving functions: Highway pilot," in Autonomous Vehicle Test & Development Symposium 2017, June 2017. [Online]. Available: https://elib.dlr.de/117384/

[71] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," 1970.

[72] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-6, no. 6, pp. 721–741, 1984.

[73] Technical Committee ISO/TC 204, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," International Organization for Standardization, techreport ISO/SAE PAS 22736, 2021. [Online]. Available: https://www.iso.org/standard/73766.html

[74] ——, "Intelligent transport systems — low-speed automated driving (lsad) systems for predefined routes — performance requirements, system requirements and performance test procedures," International Organization for Standardization, techreport ISO 22737:2021, 2021. [Online]. Available: https://www.iso.org/standard/73767.html

[75] B. Gangopadhyay, S. Khastgir, S. Dey, P. Dasgupta, G. Montana, and P. A. Jennings, "Identification of test cases for automated driving systems using bayesian optimization," in Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand. IEEE, 2019, pp. 1961–1967. [Online]. Available: https://doi.org/10.1109/ITSC.2019.8917103

[76] M. Althoff and S. Lutz, "Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles," IEEE Intelligent Vehicles Symposium (IV), pp. 1326–1333, 2018.

[77] W. Ding, B. Chen, M. Xu, and D. Zhao, "Learning to collide: An adaptive safety-critical scenarios generating method," in International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 2243–2250.

[78] Y. Li, J. Tao, and F. Wotawa, "Ontology-based test generation for automated and autonomous driving functions," Information and software technology, vol. 117, p. 106200, 2020.

[79] C. E. Tuncali, T. P. Pavlic, and G. E. Fainekos, "Utilizing s-taliro as an automatic test generation framework for autonomous vehicles," in International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2016, pp. 1470–1475.

[80] Y. Annpureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan, "S-taliro: A tool for temporal logic falsification for hybrid systems," in Tools and Algorithms for the Construction and Analysis of Systems, P. A. Abdulla and K. R. M. Leino, Eds. Springer Berlin Heidelberg, 2011, pp. 254–257.

[81] T. Ishimatsu, N. G. Leveson, J. P. Thomas, C. H. Fleming, M. Katahira, Y. Miyamoto, R. Ujiie, H. Nakao, and N. Hoshino, "Hazard analysis of complex spacecraft using systems-theoretic process analysis," Journal of Spacecraft and Rockets, vol. 51, no. 2, pp. 509–522, 2014.

[82] S. Khastgir, G. Dhadyalla, S. Birrell, S. Redmond, R. Addinall, and P. Jennings, "Test scenario generation for driving simulators using constrained randomization technique," SAE Technical Paper, Tech. Rep., 2017.

[83] Vertizan. (2016) Vitaq test automation tool. [Online]. Available: https://vitaq.io/

[84] S. Khastgir, S. A. Birrell, G. Dhadyalla, and P. A. Jennings, "Identifying a gap in existing validation methodologies for intelligent automotive systems: Introducing the 3xd simulator," in IEEE Intelligent Vehicles Symposium, IV 2015, Seoul, South Korea, 2015, pp. 648–653.

[85] M. Klischat and M. Althoff, "Generating critical test scenarios for automated vehicles with evolutionary algorithms," in IEEE Intelligent Vehicles Symposium (IV), 2019, pp. 2352–2358. [Online]. Available: https://doi.org/10.1109/IVS.2019.8814230

[86] R. Krajewski, T. Moers, D. Nerger, and L. Eckstein, "Data-driven maneuver modeling using generative adversarial networks and variational autoencoders for safety validation of highly automated vehicles," in International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018, pp. 2383–2390. [Online]. Available: https://doi.org/10.1109/ITSC.2018.8569971

[87] C. Gnandt, T. Ponn, and F. Diermeyer, "An optimization-based method to identify relevant scenarios for type approval of automated vehicles," in Proceedings of the ESV—International Technical Conference on the Enhanced Safety of Vehicles, Eindhoven, The Netherlands, 01 2019, pp. 10–13.

[88] E. Rocklage, H. Kraft, A. Karatas, and J. Seewig, "Automated scenario generation for regression testing of autonomous vehicles," in 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp. 476–483, ISSN: 2153-0017.

[89] S. Route, "Test optimization using combinatorial test design: Real-world experience in deployment of combinatorial testing at scale," in 2017 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), pp. 278–279.

[90] C. E. Tuncali and G. Fainekos, "Rapidly-exploring random trees for testing automated vehicles," in 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pp. 661–666.

[91] D. Åsljung, J. Nilsson, and J. Fredriksson, "Comparing collision threat measures for verification of autonomous vehicles using extreme value theory," vol. 49, pp. 57–62.

[92] ——, "Using extreme value theory for vehicle level safety validation and implications for autonomous vehicles," vol. 2, no. 4, pp. 288–297, conference Name: IEEE Transactions on Intelligent Vehicles.

[93] D. Zhao, H. Lam, H. Peng, S. Bao, D. J. LeBlanc, K. Nobukawa, and C. S. Pan, "Accelerated evaluation of automated vehicles safety in lane-change scenarios based on importance sampling techniques," vol. 18, no. 3, pp. 595–607, conference Name: IEEE Transactions on Intelligent Transportation Systems.

[94] Y. Akagi, R. Kato, S. Kitajima, J. Antona-Makoshi, and N. Uchida, "A risk-index based sampling method to generate scenarios for the evaluation of automated driving vehicle safety," in 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pp. 667–672.

[95] S. Prialé Olivares, N. Rebernik, A. Eichberger, and E. Stadlober, "Virtual stochastic testing of advanced driver assistance systems," in Advanced Microsystems for Automotive Applications 2015, ser. Lecture Notes in Mobility, T. Schulze, B. Müller, and G. Meyer, Eds. Springer International Publishing, pp. 25–35.

[96] P. Feig, J. Schatz, L. Audi, and T. Leonhardt, "Assessment of technical requirements for level 3 and beyond automated driving systems based on naturalistic driving and accident data analysis."

[97] F. Fahrenkrog, L. Wang, T. Platzer, A. Fries, and F. Raisch, "PROSPECTIVE SAFETY EFFECTIVENESS ASSESSMENT OF AUTOMATED DRIVING FUNCTIONS – FROM THE METHODS TO THE RESULTS," p. 11.

[98] J. J. So, I. Park, J. Wee, S. Park, and I. Yun, "Generating traffic safety test scenarios for automated vehicles using a big data technique," vol. 23, no. 6, pp. 2702–2712. [Online]. Available: https://doi.org/10.1007/s12205-019-1287-4

[99] F. Gao, J. Duan, Y. He, and Z. Wang, "A test scenario automatic generation strategy for intelligent driving systems," vol. 2019, pp. 1–10.

[100] Q. Xia, J. Duan, F. Gao, T. Chen, and C. Yang, "Automatic generation method of test scenario for ADAS based on complexity."

[101] Q. Xia, J. Duan, F. Gao, Q. Hu, and Y. He, "Test scenario design for intelligent driving system ensuring coverage and effectiveness," vol. 19, no. 4, pp. 751–758. [Online]. Available: https://doi.org/10.1007/s12239-018-0072-6

[102] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer, "Adaptive stress testing for autonomous vehicles." [Online]. Available: http://arxiv.org/abs/1902.01909

[103] A. Corso, P. Du, K. Driggs-Campbell, and M. J. Kochenderfer, "Adaptive stress testing with reward augmentation for autonomous vehicle validation." [Online]. Available: http://arxiv.org/abs/1908.01046

[104] H. Beglerovic, M. Stolz, and M. Horn, "Testing of autonomous vehicles using surrogate models and stochastic optimization," in 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pp. 1–6, ISSN: 2153-0017.

[105] R. Ben Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing advanced driver assistance systems using multi-objective search and neural networks," in 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 63–74.

[106] G. E. Mullins, "ADAPTIVE SAMPLING METHODS FOR TESTING AUTONOMOUS SYSTEMS," accepted: 2018-09-12T05:38:36Z. [Online]. Available: https://drum.lib.umd.edu/handle/1903/21225

[107] B. Gangopadhyay, S. Khastgir, S. Dey, P. Dasgupta, G. Montana, and P. Jennings, "Identification of test cases for automated driving systems using bayesian optimization," in 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pp. 1961–1967.

[108] M. Nabhan, M. Schoenauer, Y. Tourbier, and H. Hage, "Optimizing coverage of simulated driving scenarios for the autonomous vehicle," in 2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE), pp. 1–5, ISSN: 2378-1297.

[109] H. Abbas, M. O'Kelly, A. Rodionova, and R. Mangharam, "Safe at any speed: A simulation-based test harness for autonomous vehicles."

[110] C. E. Tuncali, G. Fainekos, D. Prokhorov, H. Ito, and J. Kapinski, "Requirements-driven test generation for autonomous vehicles with machine learning components." [Online]. Available: http://arxiv.org/abs/1908.01094

[111] S. S.-S. S. Shammah and A. Shashua, "On a formal model of safe and scalable self-driving cars," https://arxiv.org/abs/1708.06374, 2017.

[112] D. Nister, H.-L. Lee, J. Ng, and Y. Wang. (2017) An introduction to the safety force field. Withe paper from NVIDIA. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/self-driving-cars/safety-force-field/an-introduction-to-the-safety-force-field-v2.pdf

[113] M. Minderhoud and P. Bovy, "Extended time-to-collision measures for road traffic safety assessment," Accident; analysis and prevention, vol. 33, no. 1, p. 89—97, January 2001. [Online]. Available: https://doi.org/10.1016/s0001-4575(00)00019-1

[114] V. S. J. Hillenbrand, K. Kroschel, "Situation assessment algorithm for a collision prevention assistant," Proceedings of the 2005 Intelligent Vehicles Symposium, 2005.

[115] J. Eggert and T. Puphal, "Continuous risk measures for adas and ad," in 2017 4th International Symposium on Future Active Safety Technology towards Zero-Traffic-Accidents (FAST-zero), Sep. 18 - 21, 2017, Nara Kasugano International Forum, Nara, Japan, 2017.

[116] A. Lambert, D. Gruyer, and G. S. Pierre, "A fast monte carlo algorithm for collision probability estimation," in International Conference on Control, Automation, Robotics and Vision 2008 (ICARCV 2008), Hanoi, Vietnam, December 2008.

[117] Y. Li, J. Lu, and K. Xu, "Crash risk prediction model of lane-change behavior on approaching intersections," Discrete Dynamics in Nature and Society, vol. 2017, p. 1–12, August 2017.

[118] S. Alvarez, Y. Page, U. Sander, F. Fahrenkrog, and T. Helmer, "Prospective effectiveness assessment of adas and active safety systems via virtual simulation: A review of the current practices," in 25th International Technical Conference on the Enhanced Safety of Vehicles (ESV). NHTSA, 2017.

[119] S. Hallerbach, Y. Xia, U. Eberle, and F. Koester, "Simulation-based identification of critical scenarios for cooperative and automated vehicles," SAE Technical Paper, vol. 1066, p. 1–12, April 2018. [Online]. Available: https://arxiv.org/pdf/1704.01006.pdf

[120] J. Stellet, P. Vogt, J. Schumacher, W. Branz, and J. Zollner, "Analytical derivation of performance bounds of autonomous emergency brake systems," in 2016 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2016, p. 220–226.

[121] D. Althoff, "Safety assessment for motion planning in uncertain and dynamic environments," in PhD thesis, Technische Universitat Munchen, Germany, 2014. PhD Thesis, Technische Universitat Munchen., 2014.

[122] C. Katrakazas, M. Quddus, and W.-H. Chen, "A new methodology for collision risk assessment of autonomous vehicles," in In Proceedings of Transportation Research Board 96th Annual Meeting (TRB 2017), volume 8, Washington D.C., USA, January 2017, 2017.

[123] A. Philipp and D. Goehring, "Analytic collision risk calculation for autonomous vehicle navigation," in In Proceedings of IEEE 2019 International Conference on Robotics and Automation (ICRA'19), page 1744–1750, Montreal, QC, Canada, May 2019., 2019.

[124] L. Rummelhard, A. Négre, and C. Laugier, "Conditional monte carlo dense occupancy tracker," in IEEE 18th International Conference on Intelligent Transportation Systems, 2015, pp. 2485–2490.

[125] L. Rummelhard, A. Nègre, M. Perrollaz, and C. Laugier, "Probabilistic grid-based collision risk prediction for driving application," in International Synposium on Experimental Robotics, Springer, Ed., 2014.

[126] T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet, "Approximate probabilistic model checking," in Proceedings of the 5th International Conference on Verification, Model Checking, and Abstract Implementations. Springer Berlin Heidelberg, 2004, vol. 2937, pp. 73–84.

[127] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in 1st Annual Conference on Robot Learning, 2017, pp. 1–16.

[128] P. Ledent, A. Paigwar, A. Renzaglia, R. Mateescu, and C. Laugier, "Formal validation of probabilistic collision risk estimation for autonomous driving," in 2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM). IEEE, 2019, pp. 433–438.

[129] R. Mateescu and H. Garavel, "XTL: A Meta-Language and Tool for Temporal Logic Model-Checking," in Proceedings of the International Workshop on Software Tools for Technology Transfer (STTT'98), Aalborg, Denmark, 1998, pp. 33–42.

[130] H. Garavel, F. Lang, R. Mateescu, and W. Serwe, "CADP 2011: A Toolbox for the Construction and Analysis of Distributed Processes," Springer International Journal on Software Tools for Technology Transfer (STTT), vol. 15, no. 2, pp. 89–107, 2013.

[131] L. A. S. Guardini, A. Spalanzani, C. Laugier, P. Martinet, A.-L. Do, and T. Hermitte, "Employing severity of injury to contextualize complex risk mitigation scenarios," in IEEE Intelligent Vehicles Symposium (IV), 2020, pp. 1839–1845.

[132] S. Demmel, D. Gruyer, J.-M. Burkhardt, S. Glaser, G. Larue, O. Orfila, and A. Rakotonirainy, "Global risk assessment in an autonomous driving context: Impact on both the car and the driver," in special session "Human-Machine Interactions in Autonomous Driving", 2nd IFAC Cyber-Physical & Human Systems (CPHS 2018), December 14-15 2018, Miami, USA., 2018.

[133] J. Leroy, D. Gruyer, O. Orfila, and N.-E. E. Faouzi, "Adapted risk indicator for autonomous driving system with uncertainties and multi-dimensional configurations modeling," in 24th IEEE International Conference on Intelligent Transportation (ITSC2021), September 19-22, 2021, Indianapolis, IN, United States, 2021.

[134] S. S. Mahmud, L. Ferreira, S. Hoque, and A. Tavassoli, "Application of proximal surrogate indicators for safety evaluation: A review of recent developments and research needs," IATSS Research, vol. 41, pp. 153–163, December 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0386111217300286?via%3Dihub

[135] J. Pinnow, M. Masoud, M. Elhenawy, and S. Glaser, "A review of naturalistic driving study surrogates and surrogate indicator viability within the context of different road geometries," Accident Analysis and Prevention, vol. 157, April 2021.

[136] Y. Akagi, R. Kato, S. Kitajima, J. Antona-Makoshi, and N. Uchida, "A risk-index based sampling method to generate scenarios for the evaluation of automated driving vehicle safety," in 2019 IEEE Intelligent Transportation Systems Conference (ITSC) Auckland, NZ, October 27-30, 2019. IEEE, October 2019, pp. 667–672. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8917311

[137] S. M. X. Z. X. Y.-D. L. K.-H. T. A. Golbraikh, A., "Rational selection of training and test sets for the development of validated qsar models." Journal of computer-aided molecular design, vol. 17, no. 1, p. 241–253, 2003.

[138] R. K. Roy, P.P., "On some aspects of variable selection for partial least squares regression models." QSAR Combinatorial Science, vol. 27, no. 1, p. 302–313, 2008.

[139] T. R. Frank, I.E., The data analysis handbook. Elsevier, 1994.

[140] A. A. S. M. Gandomi, A.H., "New formulation for compressive strength of cfrp confined concrete cylinders using linear genetic programming." Materials and Structures, vol. 43, no. 1, p. 963–983, 2010.

[141] B. A. Golafshani, E.M., "Automatic regression methods for formulation of elastic modulus of recycled aggregate concrete." Applied Soft Computing, vol. 64, no. 1, p. 377–400, 2018.

[142] F. P. P. D. Cheng, M.-Y., "High-performance concrete compressive strength prediction using genetic weighted pyramid operation tree (gwpot)." Engineering Applications of Artificial Intelligence, vol. 29, no. 1, p. 104–113, 2014.

[143] N. Gählert, N. Jourdan, M. Cordts, U. Franke, and J. Denzler, "Cityscapes 3d: Dataset and benchmark for 9 dof vehicle detection," in CVPR 2020 Workshop on Scalability in Autonomous Driving, 2020. [Online]. Available: https://arxiv.org/pdf/2006.07864.pdf

[144] U. D. of Defense. (2009, December) Dod modeling and simulation (m&s) verification, validation, and accreditation (vv&a). DoD Instruction 5000.61.

[145] D. Cook and J. M. Skinner, "How to perform credible verification, validation, and accreditation for modeling and simulation," CrossTalk: The Journal of Defense Software Engineering, May 2005.

[146] R. Lewis, Independent Verification and Validation. Wiley & Sons, 1992.

[147] Y. Li, J. Tao, and F. Wotawa, "Ontology-based test generation for automated and autonomous driving functions," Information and Software Technology, vol. 117, October 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584918302271?via%3Dihub

[148] S. Riedmaier, D. Schneider, D. Watzenig, F. Diermeyer, and B. Schick, "Model validation and scenario selection for virtual-based homologation of automated vehicles," Applied Sciences, vol. 11, 2021.

[149] D. Foures, "Validation de modèles de simulation," in PhD thesis defended in University Toulouse III Paul Sabatier, Automatique, 2015. tel-01200720, 2015.

[150] S. Luna, A. Lopes, H. Y. S. Tao, F. Zapata, and R. Pin, "Integration, verification, validation, test, and evaluation (ivvt&e) framework for system of systems (sos)," Procedia Computer Science, vol. 20, p. 298 – 305, 2013. [Online]. Available: https://reader.elsevier.com/reader/sd/pii/S1877050913010764?token=DDB34A64EAF42C666491627D568603867F403E1F9C928D01591CE395FBE8496B81228FEC49C originRegion=eu-west-1&originCreation=20211020062924

[151] D. H. . 623. (2018) A framework for automated driving system testable cases and scenarios. NHTSA report. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.gov/files/documents/13882-automateddrivingsystems_092618_v1a_tag.pdf

[152] F. Jiménez, J. E. Naranjo, J. J. Anaya, F. García, A. Ponz, and J. M. Armingol, "Advanced driver assistance system for road environments to improve safety and efficiency," Transportation research procedia, vol. 14, pp. 2245–2254, 2016.

[153] L. Pullum, "Verification and validation of systems in which ai is a key element," Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), Tech. Rep., 2021.

[154] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, and M. Young, "Machine learning: The high interest credit card of technical debt," 2014.

[155] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," arXiv preprint arXiv:1412.6572, 2014.

[156] Y. K. Dwivedi, L. Hughes, E. Ismagilova, G. Aarts, C. Coombs, T. Crick, Y. Duan, R. Dwivedi, J. Edwards, A. Eirug et al., "Artificial intelligence (ai): Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy," International Journal of Information Management, p. 101994, 2019.

[157] B. Mittelstadt, C. Russell, and S. Wachter, "Explaining explanations in ai," in Proceedings of the conference on fairness, accountability, and transparency, 2019, pp. 279–288.

[158] H. Ziade, R. A. Ayoubi, R. Velazco et al., "A survey on fault injection techniques," Int. Arab J. Inf. Technol., vol. 1, no. 2, pp. 171–186, 2004.

[159] S. Jha, S. S. Banerjee, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer, "Avfi: Fault injection for autonomous vehicles," in 2018 48th annual ieee/ifip international conference on dependable systems and networks workshops (dsn-w). IEEE, 2018, pp. 55–56.

[160] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in Conference on robot learning. PMLR, 2017, pp. 1–16.

[161] L. Li, D. Wen, N.-N. Zheng, and L.-C. Shen, "Cognitive cars: A new frontier for adas research," IEEE Transactions on Intelligent Transportation Systems, vol. 13, no. 1, pp. 395–407, 2011.

[162] M. M. Trivedi, T. Gandhi, and J. McCall, "Looking-in and looking-out of a vehicle: Computer-vision-based enhanced vehicle safety," IEEE Transactions on Intelligent Transportation Systems, vol. 8, no. 1, pp. 108–120, 2007.

[163] C. Ebert and M. Weyrich, "Validation of autonomous systems," IEEE Software, vol. 36, no. 5, pp. 15–23, 2019.

[164] C. M. Kang, S.-H. Lee, and C. C. Chung, "Multirate lane-keeping system with kinematic vehicle model," IEEE Transactions on Vehicular Technology, vol. 67, no. 10, pp. 9211–9222, 2018.

[165] S. Jung, S. Hwang, H. Shin, and D. H. Shim, "Perception, guidance, and navigation for indoor autonomous drone racing using deep learning," IEEE Robotics and Automation Letters, vol. 3, no. 3, pp. 2539–2544, 2018.

[166] Y. Zhou, J.-J. Yang, and L.-Y. Zheng, "Multi-agent based hyper-heuristics for multi-objective flexible job shop scheduling: A case study in an aero-engine blade manufacturing plant," Ieee Access, vol. 7, pp. 21 147–21 176, 2019.

[167] C. Domínguez, J.-M. Martínez, J. V. Busquets-Mataix, and H. Hassan, "Human–computer cooperation platform for developing real-time robotic applications," The Journal of Supercomputing, vol. 75, no. 4, pp. 1849–1868, 2019.

[168] S. Riedmaier, J. Nesensohn, C. Gutenkunst, T. Düser, B. Schick, and H. Abdellatif, "Validation of x-in-the-loop approaches for virtual homologation of automated driving functions," in 11th Graz Symposium Virtual Vehicle, 2018.

[169] W. Huang, K. Wang, Y. Lv, and F. Zhu, "Autonomous vehicles testing methods review," in 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2016, pp. 163–168.

[170] Z. Papp, K. Labibes, A. Thean, and M. Van Elk, "Multi-agent based hil simulator with high fidelity virtual sensors," in IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No. 03TH8683). IEEE, 2003, pp. 213–218.

[171] J. Li, F. Yu, J. Zhang, J. Feng, and H. Zhao, "The rapid development of a vehicle electronic control system and its application to an antilock braking system based on hardware-in-the-loop simulation," Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, vol. 216, no. 2, pp. 95–105, 2002.

[172] R. Isermann, J. Schaffnit, and S. Sinsel, "Hardware-in-the-loop simulation for the design and testing of engine-control systems," Control Engineering Practice, vol. 7, no. 5, pp. 643–653, 1999.

[173] Y. Fu, A. Terechko, T. Bijlsma, P. J. Cuijpers, J. Redegeld, and A. O. Örs, "A retargetable fault injection framework for safety validation of autonomous vehicles," in 2019 IEEE International Conference on Software Architecture Companion (ICSA-C). IEEE, 2019, pp. 69–76.

[174] D. Portugal, P. Alvito, E. Christodoulou, G. Samaras, and J. Dias, "A study on the deployment of a service robot in an elderly care center," International Journal of Social Robotics, vol. 11, no. 2, pp. 317–341, 2019.

[175] M. R. Zofka, S. Ulbrich, D. Karl, T. Fleck, R. Kohlhaas, A. Rönnau, R. Dillmann, and J. M. Zöllner, "Traffic participants in the loop: A mixed reality-based interaction testbed for the verification and validation of autonomous vehicles," in 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018, pp. 3583–3590.

[176] S. A. Fayazi and A. Vahidi, "Vehicle-in-the-loop (vil) verification of a smart city intersection control scheme for autonomous vehicles," in 2017 IEEE Conference on Control Technology and Applications (CCTA). IEEE, 2017, pp. 1575–1580.

[177] M. Butenuth, R. Kallweit, and P. Prescher, "Vehicle-in-the-loop real-world vehicle tests combined with virtual scenarios," ATZ worldwide, vol. 119, no. 9, pp. 52–55, 2017.

[178] T. Tettamanti, M. Szalai, S. Vass, and V. Tihanyi, "Vehicle-in-the-loop test environment for autonomous driving with microscopic traffic simulation," in 2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES). IEEE, 2018, pp. 1–6.

[179] J. Coronel-Reyes, I. Ramirez-Morales, E. Fernandez-Blanco, D. Rivero, and A. Pazos, "Determination of egg storage time at room temperature using a low-cost nir spectrometer and machine learning techniques," Computers and Electronics in Agriculture, vol. 145, pp. 1–10, 2018.

[180] M. Dikmen and C. M. Burns, "Autonomous driving in the real world: Experiences with tesla autopilot and summon," in Proceedings of the 8th international conference on automotive user interfaces and interactive vehicular applications, 2016, pp. 225–228.

[181] J. Choi, J. Lee, D. Kim, G. Soprani, P. Cerri, A. Broggi, and K. Yi, "Environment-detection-and-mapping algorithm for autonomous driving in rural or off-road environment," IEEE Transactions on Intelligent Transportation Systems, vol. 13, no. 2, pp. 974–982, 2012.

[182] R. K. Vithanage, C. S. Harrison, and A. K. De Silva, "Autonomous rolling-stock coupler inspection using industrial robots," Robotics and Computer-integrated Manufacturing, vol. 59, pp. 82–91, 2019.

[183] D. Gruyer, M. Grapinet, and P. Desouza, "Modeling and validation of a new generic virtual optical sensor for adas prototyping," in in IEEE Intelligent Vehicle symposium, 2012, Alcalá de Henares, June 3-7, Spain, 2012.

[184] M. Grapinet, P. Desouza, J. Smal, and J. Blosseville, "Characterization and simulation of optical sensors," in TRA 2012 conference, 23-26 April 2012, Athens, Greece., 2012.

[185] B. DC, "Decentering distortion of lenses," Photogrammetric Engineering, vol. 7, pp. 444–462, 1966.

[186] W. Smith, Modern optical engineering: the design of optical systems. Fourth Edition, McGraw-Hill Companies, 2008.

[187] I. S. ISO14524(2009). (2009) Photography - electronic still-picture cameras — methods for measuring opto-electronic conversion functions.

[188] F. P. Garcia Marquez, F. Jimenez, J. Naranjo, J. Zato, F. Aparicio Izquierdo, J. Armingo, and A. de la Escalera, "Analysis of lidar sensors for new adas applications. usability in moving obstacles detection," 2009.

[189] M. Hadj-Bachir and P. de Souza, "Lidar sensor simulation in adverse weather condition for driving assistance development," 2019.

[190] M. Hadj-Bachir, P. de Souza, P. Nordqvist, and N. Roy, "Modelling of lidar sensor disturbances by solid airborne particles," arXiv preprint arXiv:2105.04193, 2021.

[191] M. Hadj-Bachir, P. de Souza, J. Shaik, and G. Dominique, "Evaluating autonomous functions performance through simulation using interoperable sensor, vehicle, and environment models," in FISITA Web Congress 2020 & World Congress 2021, 2020.

[192] M. Hadj-Bachir, P. De Souza, and D. Gruyer, "Virtuelle tests autonomer funktionen durch simulation," ATZextra, vol. 25, no. 1, pp. 18–21, 2020.

[193] J. Lombacher, K. Laudt, M. Hahn, J. Dickmann, and C. Wöhler, "Semantic radar grids," in 2017 IEEE intelligent vehicles symposium (IV). IEEE, 2017, pp. 1170–1175.

[194] F. Meinl, M. Stolz, M. Kunert, and H. Blume, "An experimental high performance radar system for highly automated driving," in 2017 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM). IEEE, 2017, pp. 71–74.

[195] T. Visentin, J. Hasch, and T. Zwick, "Calibration of a fully polarimetric 8× 8 mimo fmcw radar system at 77 ghz," in 2017 11th European Conference on Antennas and Propagation (EuCAP). IEEE, 2017, pp. 2530–2534.

[196] J. Dickmann, J. Klappstein, M. Hahn, N. Appenrodt, H.-L. Bloecher, K. Werber, and A. Sailer, "Automotive radar the key technology for autonomous driving: From detection and ranging to environmental understanding," in 2016 IEEE Radar Conference (RadarConf). IEEE, 2016, pp. 1–6.

[197] A. Arage, W. M. Steffens, G. Kuehnle, and R. Jakoby, "Effects of water and ice layer on automotive radar," in Proc. of the German Microwave Conf. Citeseer, 2006.

[198] M. Holder, P. Rosenberger, H. Winner, T. D'hondt, V. P. Makkapati, M. Maier, H. Schreiber, Z. Magosi, Z. Slavik, O. Bringmann et al., "Measurements revealing challenges in radar sensor modeling for virtual validation of autonomous driving," in 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018, pp. 2616–2622.

[199] J.-C. Kedzia, P. de Souza, and D. Gruyer, "Advanced radar sensors modeling for driving assistance systems testing," in 2016 10th European Conference on Antennas and Propagation (EuCAP). IEEE, 2016, pp. 1–2.

[200] D. Gruyer, S. Laverdure, J.-S. Berthy, P. Desouza, and M. Hadj-Bachir, "From virtual to real, how to prototype, test, evaluate and validate adas for the automated and connected vehicle?" in From AI to Autonomous and Connected Vehicles, Advanced Driver-Assistance Systems (ADAS), ser. Digital Science, T. Bapin and A. Bensrhair, Eds. New York: ISTE Ltd., London, and John Wiley and Sons, 2021, ch. 4. [Online]. Available: http://iste.co.uk/book.php?id=1800

[201] J. Stevens, "The how and why of open architecture," Undersea Warfare, no. 37, pp. 6–9, 2008.

[202] S. Yoo and A. Jerraya, "Introduction to hardware abstraction layers for soc," in 2003 Design, Automation and Test in Europe Conference and Exhibition, 2003, pp. 336–337.

[203] A. Gazis and E. Katsiri, "Middleware 101: What to know now and for the future," Queue, vol. 20, no. 1, pp. 10–23, 2022.

[204] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," Annual Review of Control, Robotics, and Autonomous Systems, 2018.

[205] M. J. Kochenderfer, Decision making under uncertainty: theory and application. MIT press, 2015.

[206] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 4693–4700.

[207] J. Chen, B. Yuan, and M. Tomizuka, "Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety," arXiv preprint arXiv:1903.00640, 2019.

[208] C.-J. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, and M. J. Kochenderfer, "Combining planning and deep reinforcement learning in tactical decision making for autonomous driving," IEEE Transactions on Intelligent Vehicles, vol. 5, no. 2, pp. 294–305, 2019.

[209] G. Rill, "Vehicle modeling by subsystems," Journal of the Brazilian Society of Mechanical Sciences and Engineering, vol. 28, 2006. [Online]. Available: https://www.researchgate.net/publication/239409981_Vehicle_modeling_by_subsystems

[210] G. Rill and A. A. Castro, Road Vehicle Dynamics: Fundamentals and Modeling with MATLAB®. CRC Press, Taylor and Francis Group, May 2020.

[211] S. Glaser. (2004, Janvier) Modélisation et analyse d'un véhicule en trajectoires limites application au développement de systèmes d'aide à la conduite. THÈSE DE DOCTORAT de l'Université d'Evry Val d'Essonne, spécialité Automatique, Systèmes Productiques et Robotique.

[212] L. Li. (2021, January) Modélisation et contrôle d'un véhicule tout-terrain à deux trains directeurs. THÈSE DE DOCTORAT de l'Université de recherche Paris Sciences et Lettres PSL Research University, Mines ParisTech, École doctorale no432 SCIENCE DES MÉTIERS DE L'INGÉNIEUR Spécialité MATHÉMATIQUES ET INFORMATIQUE TEMPS-RÉEL.

[213] B. Chretien. (2012, January) Simulation of a new automotive concept based on a centralized approach for driver assistance system activation decision. THÈSE DE DOCTORAT de l'Université d'Evry Val d'Essonne, spécialité Automatique, Systèmes Productiques et Robotique.

[214] J. Vargas, S. Alsweiss, O. Toker, R. Razdan, and J. Santos, "An overview of autonomous vehicles sensors and their vulnerability to weather conditions," Sensors, MDPI, 2021.

[215] I. Gultepe, R. Tardif, S. C. Michaelides, J. Cermak, A. Bott, J. Bendix, M. D. Müller, M. Pagowski, B. Hansen, G. Ellrod, W. Jacobs, G. Toth, and S. G. Cober, "Fog research: A review of past achievements and future perspectives," Pure and Applied Geophysics, 2007.

[216] P. Duthon, M. Colomb, and F. Bernardin, "Fog classification by their droplet size distributions: Application to the characterization of cerema's platform," Atmosphere, vol. 11, no. 6, 2020. [Online]. Available: https://www.mdpi.com/2073-4433/11/6/596

[217] L. Zhang, A. Zhu, and Y. Zhou, "Simulation of atmospheric visibility impairment," vol. 30, pp. 8713–8726, 2021.

[218] L. Li, P. Kooi, M. Leong, and T. Yeo, "On the simplified expression of realistic raindrop shapes," Microwave and optical technology letters, vol. 7, no. 4, pp. 201–205, Mar. 1994.

[219] P. Räisänen, A. Kokhanovsky, G. Guyot, O. Jourdan, and T. Nousiainen, "Parameterization of single-scattering properties of snow," The Cryosphere, 2015.

[220] A. S. Mohammed, A. Amamou, F. K. Ayevide, S. Kelouwani, K. Agbossou, and N. Zioui, "The perception system of intelligent ground vehicles in all weather conditions: A systematic literature review," Sensors, MDPI, 2020.

[221] I. R. Assembly, "Specific attenuation model for rain for use in prediction methods," International Telecommunication Union, Tech. Rep., 2005, iTU-R P.838-3. [Online]. Available: https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.838-3-200503-I!!PDF-E.pdf

[222] S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, and M. A. Kaafar, "The impact of adversary weather conditions on autonomous vehicles," IEEE Vehicular Technology Magazine, 2019.

[223] S. Hasirlioglu, "A novel method for simulation-based testing and validation of automotive surround sensors under adverse weather conditions/submitted by sinan hasirlioglu," Ph.D. dissertation, Universität Linz, 2020. [Online]. Available: https://epub.jku.at/obvulihs/content/titleinfo/4837383/full.pdf

[224] I. J. Xique, W. Buller, Z. B. Fard, E. Dennis, and B. Hart, "Evaluating complementary strengths and weaknesses of adas sensors," in 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall), IEEE Vehicular Technology Conference. Chicago, IL, USA: IEEE, Aug. 2018. [Online]. Available: https://www.researchgate.net/profile/William-Buller/publication/328968313_Evaluating_Complementary_Strengths_and_Weaknesses_of_ADAS_Sensors_IEEE_88th_Vehicular_Technology_Conference_Aug_2018_Chicago_USA/links/5beddbf3299bf1124fd5d73e/Evaluating-Complementary-Strengths-and-Weaknesses-of-ADAS-Sensors-IEEE-88th-Vehicular-Tech pdf

[225] K. Garg and S. Nayar, "When does a camera see rain?" IEEE, 2005, pp. 1067–1074 Vol. 2. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1544839

[226] J. Wojtanowski, M. Zygmunt, M. Kaszczuk, Z. Mierczyk, and M. Muzal, "Comparison of 905 nm and 1550 nm semiconductor laserrangefinders' performance deterioration due to adverseenvironmental conditions," Opto-Electronic Review, 2014.

[227] A. Filgueira, H. Gonzalez-Jorge, S. Lagüela, L. Diaz-Vilarino, and P. Arias, "Quantifying the influence of rain in lidar performance," Measurement, vol. 95, pp. 143–148, 2016.

[228] Y. Zhou, L. Liu, H. Zhao, M. Lopez-Benitez, L. Yu, and Y. Yue, "Towards deep radar perception for autonomous driving: Datasets, methods, and challenges," 2022.

[229] P. H. Chan, G. Dhadyalla, and V. Donzella, "A framework to analyze noise factors of automotive perception sensors," IEEE Sensors Letters, vol. 4, no. 6, Jun. 2020.

[230] R. Rasshofer and K. Gresser, "Automotive radar and lidar systems for next generation driverassistance functions," Advances in Radio Science, vol. 3, pp. 205–209, 2005. [Online]. Available: https://ars.copernicus.org/articles/3/205/2005/ars-3-205-2005.pdf

[231] M. Modest, "Radiative heat transfer," Elsevier Science, 01 2003.

[232] D. Deirmendjian, Electromagnetic Scattering on Spherical Polydispersions. Santa Monica, CA: RAND Corporation, 1969.

[233] W. J. Wiscombe, Improved mie scattering algorithms. Appl. Opt., 19, 1980.

[234] H. Van de Hulst, "Light scattering by small particles," 1957.

[235] H. Koschmieder, "Theorie der horizontalen sichtweite," Beiträge zurPhysik der freien Atmosphäre, vol. 12, pp. 33–55, 1924.

[236] Z. Lee and S. Shang, "Visibility: How applicable is the century-old koschmieder model?" Journal of the Atmospheric Sciences, vol. 73, no. 11, pp. 4573 – 4581, 2016. [Online]. Available: https://journals.ametsoc.org/view/journals/atsc/73/11/jas-d-16-0102.1.xml

[237] A. Ben-Daoued, P. Duthon, and F. Bernardin, "SWEET: A Realistic Multiwavelength 3D Simulator for Automotive Perceptive Sensors in Foggy Conditions," vol. 9, no. 2, 2023. [Online]. Available: https://www.mdpi.com/2313-433X/9/2/54

[238] P. Duthon, M. Colomb, and F. Bernardin, "Light transmission in fog: The influence of wavelength on the extinction coefficient," Applied Sciences, vol. 9, no. 14, 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/14/2843

[239] M. Colomb, K. Hirech, P. André, J. Boreux, P. Lacôte, and J. Dufour, "An innovative artificial fog production device improved in the european project "fog"," Atmospheric Research, vol. 87, no. 3, pp. 242–251, 2008, third International Conference on Fog, Fog Collection and Dew. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169809507002037

[240] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan, "Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks?" in IEEE International Conference on Robotics and Automation (ICRA), Jul. 2017, pp. 746–753.

[241] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," vol. 33, no. 12, pp. 2341–2353, Dec. 2011. [Online]. Available: https://www.researchgate.net/profile/Xiaoou-Tang/publication/220182411_Single_Image_Haze_Removal_Using_Dark_Channel_Prior/links/54e9997a0cf25ba91c7f2a3e/Single-Image-Haze-Removal-Using-Dark-Channel-Prior.pdf

[242] L. M. Tang, L. H. Lim, and P. Siebert, "Removal of visual disruption caused by rainusing cycle-consistent generative adversarial networks," in European Conference on Computer Vision (ECCV) 2018, R. S. Leal-Taixé, L., Ed., vol. 11133. Springer, 2018. [Online]. Available: https://eprints.gla.ac.uk/170782/7/170782.pdf

[243] X. Fu, J. Huang, W. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture forsingle-image rain removal," IEEE Transactions on Image Processing (TIP), 2017. [Online]. Available: https://arxiv.org/pdf/1609.02087

[244] H. Guo, X. Wang, and H. Li, "Density estimation of fog in image based on dark channel prior," Atmosphere MDPI, vol. 13, no. 170, Apr. 2022.

[245] N. Zhang, L. Zhang, and Z. Cheng, Towards Simulating Foggy and Hazy Imagesand Evaluating Their Authenticity. Springer International, 2017, ch. Towards Simulating Foggy and Hazy Images and Evaluating Their Authenticity, pp. 405–415.

[246] C. Sakaridis, D. Dai, and L. Van Gool, "Semantic foggy scene understanding with synthetic data," International Journal of Computer Vision, vol. 126, no. 9, pp. 973–992, 2018. [Online]. Available: https://arxiv.org/pdf/1708.07819.pdf

[247] J. I. Gordon, "Daytime visibility, a conceptual review," no. 11, p. 22, Nov. 1979, sIO Ref. 80-1. [Online]. Available: http://misclab.umeoce.maine.edu/education/VisibilityLab/reports/SIO_80-1.pdf

[248] F. Guo, J. Tang, and X. Xiao, "Foggy scene rendering based on transmission map estimation," International Journal of Computer Games Technology, vol. 2014, p. 13, 2014.

[249] A. Giroud and V. Biri, "Modeling and rendering heterogeneous fog in real-time using b-spline wavelets," in WSCG 2010. Plzen, CZ: WSCG 2010, Feb. 2010, pp. 145–152. [Online]. Available: https://otik.uk.zcu.cz/bitstream/11025/11054/1/Giroud.pdf

[250] R. Gallen, N. Hautière, and E. Dumont, "Static estimation of the meteorological visibility distance in night fog with imagery," IEICE Transactions on Information and Systems, pp. 300–303, 2010. [Online]. Available: https://www.jstage.jst.go.jp/article/transinf/E93.D/7/E93.D_7_1780/_pdf

[251] B. Sun, R. Ramamoorthi, S. G. Narasimhan, and S. K. Nayar, "A practical analytic single scattering model for real time rendering," ACM Transactions on Graphics, vol. 24, no. 3, pp. 1040–1049, 2005. [Online]. Available: https://cseweb.ucsd.edu/~ravir/papers/singlescat/scattering.pdf

[252] A. V. Bernuth, G. Volk, and O. Bringmann, "Simulating photo-realistic snow and fog on existing images for enhanced cnn training and evaluation," in 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE Intelligent Transportation Systems Conference, Oct. 2019. [Online]. Available: https://embedded.uni-tuebingen.de/assets/publications/vonBernuth-Volk-Bringmann_Snow_Fog.pdf

[253] G. Koh, "Physical and optical properties of falling snow," US Army Corps of Engineers, Tech. Rep. [Online]. Available: https://apps.dtic.mil/sti/pdfs/ADA212432.pdf

[254] M. Hahner, C. Sakaridis, D. Dai, and L. V. Gool, "Fog simulation on real lidar point cloudsfor 3d object detection in adverse weather," in IEEE International Conference on Computer Vision (ICCV 2021), 2021. [Online]. Available: https://www.trace.ethz.ch/publications/2021/lidar_fog_simulation/HahnerICCV21.pdf

[255] R. Rasshofer, M. Spies, and H. Spies, "Influences of weather phenomena on automotive laser radar systems," Advance in Radio Science, pp. 49–60, 2011. [Online]. Available: https://pdfs.semanticscholar.org/3166/fe32780437824e35dbd23b79c8e923d5d51c.pdf

[256] C. Goodin, D. Carruth, M. Doude, and C. Hudson, "Predicting the influence of rain on lidar in adas," Electronics, MDPI, 2019. [Online]. Available: https://www.mdpi.com/2079-9292/8/1/89/htm

[257] P. A. Lewandowski, W. E. Eichinger, A. Kruger, and W. F. Krajewski, "Lidar-based estimation of small-scale rainfall: Empirical evidence," Journal of Atmospheric and Oceanic Technology, vol. 26, pp. 656–664, Mar. 2009. [Online]. Available: https://journals.ametsoc.org/view/journals/atot/26/3/2008jtecha1122_1.xml?tab_body=pdf

[258] M. Byeon and S. W. Yoon, "Analysis of automotive lidar sensor model considering scattering effects in regional rain environments," IEEE Access, vol. 8, pp. 102 669–102 679, May 2020. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9097838

[259] M. Hahner, C. Sakaridis, M. Bijelic, F. Heide, F. Yu, D. Dai, and L. V. Gool, "Lidar snowfall simulation for robust 3d object detection," in IEEE/CVF Conference on Computer Vision and Pattern Recognition 2022, 2022. [Online]. Available: https://light.princeton.edu/wp-content/uploads/2022/04/LiDAR_Snowfall_sim_paper_CVPR_2022.pdf

[260] S. R. Richter, H. A. AlHaija, and V. Koltun. (2021, May) Enhancing photorealism enhancement. [Online]. Available: https://arxiv.org/pdf/2105.04619.pdf

[261] Y. Chen, S. Chen, T. Xiao, S. Zhang, Q. Hou, and N. Zheng, "Mixed test environment-based vehicle-in-the-loop validation - a new testing approach for autonomous vehicles," in 2020 IEEE Intelligent Vehicles Symposium (IV), 2020, pp. 1283–1289.

[262] Y. Feng, C. Yu, S. Xu, H. X. Liu, and H. Peng, "An augmented reality environment for connected and automated vehicle testing and evaluation," in IEEE Intelligent Vehicles Symposium (IV), 2018.

[263] Z. Szalay, "Next generation x-in-the-loop validation methodology for automated vehicle systems," IEEE Access, 2021.

[264] C. Park, S. Chung, and H. Lee, "Vehicle-in-the-loop in global coordinates for advanced driver assistance system," Applied Sciences, 2020.

[265] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in Conference on Computer Vision and Pattern Recognition. IEEE, 2012.

[266] K. web site. (2017) 3d object detection evaluation 2017. [Online]. Available: http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d

[267] A. Simonelli, S. R. R. Bulò, L. Porzi, M. López-Antequera, and P. Kontschieder. (2019) Disentangling monocular 3d object detection. Computer Vision and Pattern Recognition. [Online]. Available: https://arxiv.org/abs/1905.12365

[268] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial cnn for traffic scene understanding," in Thirty-Second AAAI Conference on Artificial Intelligence. AAAI, 2018.

[269] S. Muckenhubera, E. Museljic, and G. Stettinger, "Performance evaluation of a state-of-the-art automotive radar and corresponding modeling approaches based on a large labeled dataset," JOURNAL OF INTELLIGENT TRANSPORTATION SYSTEMS, July 2021. [Online]. Available: https://doi.org/10.1080/15472450.2021.1959328

[270] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. (2019, March) nuscenes: A multimodal dataset for autonomous driving. [Online]. Available: https://arxiv.org/pdf/1903.11027.pdf

[271] IPG, 2019. [Online]. Available: https://ipg-automotive.com/products-services/test-systems/vil-systems/

[272] T. Schmitt, 2019. [Online]. Available: https://ipg-automotive.com/fileadmin/user_upload/content/Content_special_sizes/IPG_Webseite/Events/TECH_WEEKS_2020/Presentations_pdf/IPG_Automotive_TECH_WEEKS_Renault.pdf

[273] IDIADA, 2020. [Online]. Available: https://www.applusidiada.com/global/en/what-we-do/service-sheet/Vehicle-in-the-loop-testing

[274] S. GLASER, V. JUDALET, D. GRUYER, O. ORFILA, and S. PECHBERTI, "Virtual reality hmd for the test of adas system," in Second International Symposium on Future Active Safety Technology toward zero-traffic-accident, September,2013, Nagoya, JAPAN. JSAE, 2013, pp. 22–26.

[275] M. H. Julien Chaplier, Thomas Nguyen That and G. Gallée, "Toward a standard: Roadxml, the road network database format," DSC, pp. 211–220, 2010, https://www.ifsttar.fr/fileadmin/user_upload/editions/inrets/Actes/Actes_INRETS_A126.pdf.

[276] E. Dagan, O. Mano, G. P. Stein, and A. Shashua, "Forward collision warning with a single camera," in IEEE Intelligent Vehicles Symposium, 2004. IEEE, 2004, pp. 37–42.

[277] J. Decker, K. Saad, D. Rey, S. M. Canta, and R. A. Kipp, "Physics-based, real-time mimo radar simulation for autonomous driving," Springer Fachmedien Wiesbaden, September 2021.

[278] M. Hadj-Bachir, P. de Souza, P. Nordqvist, and N. Roy, "Modelling of lidar sensor disturbances by solid airborne particles," SIA SIMULATION NUMERIQUES, April 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02000567

[279] M. Treiber and A. Kesting, Traffic Flow Dynamics: Data, Models and Simulation. Springer-Verlag Berlin Heidelberg, 2013.

[280] J. Laval, C. Toth, and Y. Zhou, "A parsimonious model for the formation of oscillations in car-following models," Transportation Research Part B, Methodology, vol. 70, pp. 228–238, 2014.

[281] Z. He, L. Zheng, and W. Guan, "A simple nonparametric car-following model driven by field data," Transportation Research Part B, Methodology, vol. 80, p. 185–201, 2015.

[282] V. Zeidler, H. S. Buck, L. Kautzsch, P. Vortisch, and C. M. Weyland, "Simulation of autonomous vehicles based on wiedemann's car following model in ptv vissim," in Proceedings of the 2019 98th Annual Meeting of the Transportation Research Board (TRB), Washington, DC, USA, 2019, pp. 13–17.

[283] G. Wang, J. H. Z. Li, and L. Li, "Harmonious lane changing via deep reinforcement learning," IEEE Transaction on Intelligent Transportation System, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9325948.

[284] M. Guériau, R. Billot, N.-E. El Faouzi, J. Monteil, F. Armetta, and S. Hassas, "How to assess the benefits of connected vehicles? a simulation framework for the design of cooperative traffic management strategies," Transportation research part C: emerging technologies, vol. 67, pp. 266–279, 2016.

[285] H. Yu, R. Jiang, Z. He, Z. Zheng, L. Li, R. Liua, and X. Chen, "Automated vehicle-involved traffic flow studies: A survey of assumptions, models, speculations, and perspectives," Transportation Research Part C, vol. 127, April 2021.

[286] H. U. Ahmed, Y. Huang, and P. Lu, "A review of car-following models and modeling tools for human and autonomous-ready driving behaviors in micro-simulation," Smart Cities, vol. 4, p. 314–335, 2021. [Online]. Available: https://doi.org/10.3390/smartcities4010019

[287] G. F. Newell, "A simplified car-following theory: a lower order model," Transportation Research Part B: Methodological, vol. 36, no. 3, pp. 195–205, 2002.

[288] C. Lia, X. Jianga, W. Wanga, Q. Chenga, and Y. Shenb, "A simplified car-following model based on the artificial potential field," Procedia Engineering, vol. 137, p. 13–20, 2016. [Online]. Available: https://core.ac.uk/download/pdf/82041059.pdf

[289] M. Zhu, X. Wang, and Y. Wang, "Human-like autonomous car-following model with deep reinforcement learning," Transportation research part C: emerging technologies, vol. 97, pp. 348–368, 2018.

[290] M. Treiber, A. Hennecke, and D. Helbing. (2008, February) Congested traffic states in empirical observations and microscopic simulations. II. Institute of Theoretical Physics, University of Stuttgart, Pfaffenwaldring 57, D-70550 Stuttgart, Germany. [Online]. Available: https://arxiv.org/pdf/cond-mat/0002177.pdf

[291] S. Shalev-Shwartz, S. Shammah, and A. Shashua. (2018) On a formal model of safe and scalable self-driving cars. Mobileye 2017. [Online]. Available: https://arxiv.org/pdf/1708.06374.pdf

[292] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama, "City-scale traffic animation using statistical learning and metamodel-based optimization," Physical Review E, vol. 51, pp. 1035–1042, February 1995. [Online]. Available: https://courses.physics.ucsd.edu/2014/Spring/physics221a/REFERENCES/bando.pdf

[293] K. Bevrani, E. Chung, and P. Teo, "A space-based car-following model: Development and application for managed motorway system safety evaluation," Future Transp., vol. 1, p. 443–465, 2021.

[294] J. Sun, Z. Zheng, and J. Sun, "Stability analysis methods and their applicability to car-following models in conventional and connected environments," Transportation Research Part B: Methodological, vol. 109, p. 212–237, January 2018.

[295] S. Feng, Y. Zhang, S. Li, Z. Cao, H. Liu, and L. Li, "String stability for vehicular platoon control: Definitions and analysis methods," Annual Reviews in Control, vol. 47, pp. 81–97, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1367578819300240

[296] T. Toledo, "Driving behaviour: Models and challenges," Transport Reviews, vol. 27, pp. 65–84, January 2007.

[297] J. J. Olstam and A. Tapani, "Comparison of car-following models," Swedish National Road and Transport Research Institute Linkoping, vol. 960, 2004.

[298] S. Moridpour, M. Sarvi, and G. Rose, "Lane changing models: a critical review," Transportation letters, vol. 2, pp. 157–173, 2010.

[299] Z. Zheng, "Recent developments and research needs in modeling lane changing," Transportation Research Part B: Methodological, vol. 60, pp. 16–32, February 2014.

[300] J.-C. Bornard, "Développement d'un modèle du conducteur automobile: De la modélisation cognitive à la simulation numérique," PhD. Thesis, Bordeaux, 2012.

[301] T. Bellet, J.-C. Bornard, B. Richard, and S. Laverdure, "Use of a cognitive simulation model of the driver to support the virtual human centred design (v-hcd) of adas and automated vehicles," in Graz Symposium Virtual Vehicle: Artifical Intelligence Meets Model-Centric Design, 2018, 2018.

[302] X. Wang, Y. Zhang, and J. Jiao, "A state dependent mandatory lane-changing model for urban arterials with hidden markov model method," International Journal of Transportation Science and Technology, vol. 8, pp. 219–230, June 2019.

[303] X. Li and J.-Q. Sun, "Studies of vehicle lane-changing to avoid pedestrians with cellular automata," Physica A: Statistical Mechanics and its Applications, vol. 438, pp. 251–271, November 2015.

[304] D.-F. Xie, Z.-Z. Fang, B. Jia, and Z. He, "A data-driven lane-changing model based on deep learning," Transportation Research Part C: Emerging Technologies, vol. 106, pp. 41–60, September 2019.

[305] E. Balal, R. L. Cheu, and T. Sarkodie-Gyan, "A binary decision model for discretionary lane changing move based on fuzzy inference system," Transportation Research Part C: Emerging Technologies, vol. 67, pp. 47–61, June 2016.

[306] B. Vanholme, D. Gruyer, B. Lusetti, S. Glaser, and S. Mammar, "Highly automated driving on highways based on legal safety," IEEE Transaction on Intelligent Transportation System, vol. 14, pp. 333–347, March 2013.

[307] R. Balakrishna, C. Antoniou, M. Ben-Akiva, H. N. Koutsopoulos, and Y. Wen, "Calibration of microscopic traffic simulation models," Transportation Research Record, vol. 1999, no. 1, pp. 198–207, 2007. [Online]. Available: https://doi.org/10.3141/1999-21

[308] P. Maheshwary, K. Bhattacharyya, B. Maitra, and M. Boltze, "A methodology for calibration of traffic micro-simulator for urban heterogeneous traffic operations," Journal of Traffic and Transportation Engineering (English Edition), vol. 7, no. 4, pp. 507–519, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2095756418300035

[309] L. Jie, Z. Fangfang, H. van Zuylen, and L. Shoufeng, "Calibration of a micro simulation program for a chinese city," Procedia - Social and Behavioral Sciences, vol. 20, pp. 263–272, 2011, the State of the Art in the European Quantitative Oriented Transportation and Logistics Research – 14th Euro Working Group on Transportation & 26th Mini Euro Conference & 1st European Scientific Conference on Air Transport. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877042811014121

[310] F. Cunto and F. F. Saccomanno, "Calibration and validation of simulated vehicle safety performance at signalized intersections," Accident Analysis & Prevention, vol. 40, no. 3, pp. 1171–1179, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001457508000055

[311] M. Yu and W. (David) Fan, "Calibration of microscopic traffic simulation models using metaheuristic algorithms," International Journal of Transportation Science and Technology, vol. 6, no. 1, pp. 63–77, 2017, connected and Automated Vehicles: Effects on Traffic, Mobility and Urban Design. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2046043017300369

[312] M. Treiber and A. Kesting, "An open-source microscopic traffic simulator," IEEE Intelligent Transportation Systems Magazine, vol. 2, no. 3, pp. 6–13, 2010.

[313] P. A. Lopez, M. Behrisch, L. Bieker-Walz, and al, "Microscopic traffic simulation using sumo," in 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, Hawaii, USA, November 4-7, 2018.

[314] R. Klefstad, Y. Zhang, M. Lai, R. Jayakrishnan, and R. Lavanya, "A distributed, scalable, and synchronized framework for large-scale microscopic traffic simulation," in Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005., 2005, pp. 813–818.

[315] K. Ramamohanarao, H. Xie, L. Kulik, S. Karunasekera, E. Tanin, R. Zhang, and E. B. Khunayn, "Smarts: Scalable microscopic adaptive road traffic simulator," ACM Transaction on Intelligent System Technology, January 2016. [Online]. Available: https://people.eng.unimelb.edu.au/etanin/tist17.pdf

[316] S. Zatmeh-Kanj and T. Toledo, "Car following and microscopic traffic simulation under distracted driving," Transportation Research Record, vol. 2675, pp. 643–656, 2021.

[317] Q. Chao, H. Bi, W. Li, T. Mao, Z. Wang, M. C. Lin, and Z. Deng, "A survey on visual traffic simulation: Models, evaluations, and applications in autonomous driving," COMPUTER GRAPHICS forum, vol. 38, p. 1–22, 2019. [Online]. Available: https://doi.org/10.3390/smartcities4010019

[318] G. MARKKULA, "Driver behavior models for evaluating automotive active safety from neural dynamics to vehicle dynamics," THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN MACHINE AND VEHICLE SYSTEMS, CHALMERS UNIVERSITY OF TECHNOLOGY, Department of Applied Mechanics, Goteborg, Sweden, 2015.

[319] O. BENDERIUS, "Driver modeling: Data collection, model analysis, and optimization," Thesis for the degree of Licentiate of Engineering in Machine and Vehicle Systems, CHALMERS UNIVERSITY OF TECHNOLOGY, Department of Applied Mechanics, Goteborg, Sweden, 2012.

[320] S. Swarup, "Adequacy: What makes a simulation good enough?" in SpringSim-ANSS, 2019 April 29-May 2, Tucson, AZ, USA. Society for Modeling & Simulation International (SCS), April 2019. [Online]. Available: https://nssac.bii.virginia.edu/~swarup/papers/swarup_springsim2019.pdf

[321] A. Bracquemond, “Status of the moove project,” in Automated Vehicle Symposium 2019 (AVS2019), Orlando, 2019.

[322] A. Bracquemond and G. Thiolon, “Moove project : Recognition of road scenes by the data collected at the output of the sensors of the autonomous vehicle,” in 28th Aachen Colloquium Automobile and Engine Technology 2019., 2019.