Accepted article for the 6th International ATZ Conference Automated Driving 2020, Wiesbaden, Germany

Highly Parameterizable and Generic Perception Sensor Model Architecture

A Modular Approach for Simulation Based Safety Validation of Automated Driving

Clemens Linnhoff, Philipp Rosenberger, Martin Friedrich Holder, Nicodemo Cianciaruso, and Hermann Winner

Institute of Automotive Engineering, Technical University of Darmstadt, Germany, {forename.surname}@tu-darmstadt.de,

Abstract. Scenario-based virtual testing is seen as a key element to bring the overall safety validation effort for automated driving functions to an economically feasible level. In this work, a generic and modular architecture for simulation of automotive perception sensors is introduced, as part of the overall virtual testing pipeline. It is based on the functional decomposition of real world perception sensors. All interfaces between the individual modules of the model architecture are oriented on internationally recognized standards and therefore facilitate a high degree of interchangeability. In addition, a wrapper framework handles all outer communication and enables a profile-based parameterization of the model, where every profile reflects a specific set of parameters tailored to the specifications and use case of the end user.

Keywords: Perception Sensor Simulation, Safety Validation, Automated Driving

1 Introduction

In recent years, scenario-based testing has been pulled into focus, forced by research projects like *PEGASUS* [1] and *ENABLE-S3* [2]. Current projects like *SET Level 4 to 5* [3] and *VVMethods* [4] follow this approach, while focusing on simulation of the automated driving system and its environment. Here, perception sensor simulation has been identified as a major challenge of the overall simulation approach, as described in former work by the authors [5]. In this regard, functional decomposition is ideally suited to gain better insight into the information processing and loss during reception and processing by a perception sensor system, as it is highly sophisticated to reach valid perception sensor simulation [6,7,8]. It is a widely used method of choice for (safety) validation in general to reduce the exploding scenario space, when test automation is designed and scenarios are varied [9,10,11].

Consequently, interfaces must be defined between the identified functional blocks. In case of perception sensor systems and their simulation, two initiatives are establishing standards, one is the ISO 23150 initiative for real hardware [12] and the other is the Open Simulation Interface [13] (OSI) project within the ASAM e. V. [14]. In addition, there are initiatives to order and classify perception sensor simulation approaches by in-

and outputs and by their simulation method, as e.g. in recent work of the authors [15]. The paper at hand will follow up on this publication and bring in the last findings from work of the authors within the currently running research project *SET Level 4to5*. Furthermore, the publication at hand is following up a recently published article by the authors describing a modular framework for lidar sensor system simulation [16] by generalizing the approach and extending the parameterizability towards a highly parameterizable and generic perception sensor model architecture that is usable for lidar, radar, camera and ultrasonic sensor system simulation.

In the next section, definitions for interfaces, simulation approaches and model architectures are collected and extended, where needed. Afterwards, the existing modular approach and the state of the art for perception sensor system simulation are summarized, extended, and generalized, as promised. The generic approach is accompanied by a concept for parameterization, as described in a separate section. Finally, the benefits of the approach are concluded and an outlook to further work is presented.

2 Defining Sensor Models

What do you mean by 'Sensor Model'? What is the output? What is the modeling approach? – These are the questions everyone has to deal with, when coming into contact with virtual validation of automated driving functions. As a first step to preempt misunderstandings regarding different types of models, the authors previously defined the terms sensor and sensor system, to distinguish between the front-end of a sensor and the appended signal processing chain [15]. This separation aims to clarify the question of the model output, but still lacks the annotation of input and modeling approach. Therefore, a new naming convention for simulation models of automotive perception sensors is established within the mentioned running research projects. It does not directly contain the modeling approach, but describes the sensor technology and the model in- and output: First the input of the model is named, so it is *<input>-based*. Secondly follows the technology of the sensor, e.g. radar or lidar, completed by the output of the model. In general terms, this convention would read: *<input>-based <technology> <output> model*.

The introduced convention fits very well into the mentioned OSI/ISO standardization activities where appropriate interfaces are prepared. OSI in its current version 3.2.0 [17] defines a variety of the possible model inputs and outputs. The input class for a sensor model in OSI is `osi3::SensorView` that can contain two types of sensor model inputs: A global ground truth (GT) object list (OL) of all static and (possibly) moving objects and the output of a ray casting/tracing engine applied on the scene graph while considering atmospheric attenuation, materials and shapes via e.g. Bidirectional Scattering Distribution Functions [18] (BSDF), as proposed by [19]. The latter is dependent on the sensor technology and can either be a list of reflections for radar and lidar or an image for camera. Following the newly introduced naming scheme, OSI-compliant radar or lidar models can either be *object-based* or *reflection-based* and a camera model can be *object-based* as well as *image-based*.

OSI also defines two possibilities for the output of a sensor model. The first one being objects and the second one being detections within the `osi3::FeatureData` class. At

this point, the terms "detection" and "feature" are ambiguously defined. OSI in Version v.3.2.0 deviates from a second standard that can be consulted. The ISO/DIS 23150 draft international standard on data communication of road vehicles defines the interfaces between sensors and data fusion units for automated driving functions in real world systems [12]. While in OSI a detection is part of `osi3::FeatureData`, the ISO defines both as separate entities. According to ISO/DIS 23150 a detection is defined as a sensor technology specific entity in the sensors coordinate system that is based on a single measurement. A feature on the other hand is defined in the vehicle coordinate system, while still being technology specific and mostly based on a single measurement.

In related work from Philipp et al., features are defined as extractable elements of the surrounding like static and dynamic objects [11], based on the taxonomy from Ulbrich et al. [20]. However the detection level is not tackled there, so the publication cannot be used for distinguishing both terms. Nevertheless, it hints towards possible misunderstandings regarding the terms as they are ambiguously defined in the sensor modeling community. To solve the confusion regarding the terms "detection" and "feature", the authors contribute to align OSI with the ISO/DIS 23150 as part of the already existing working group within OSI.

In this work, the terms from ISO/DIS 23150 are used for the output definition of the models, as OSI will most likely adopt ISO definitions. So, a detection is the output of a single sensor in its own reference frame and in spherical coordinates. For the output of a sensor fusion unit, the term feature is used, as data are located in the reference frame of a notional sensor, in OSI called "virtual sensor", (commonly defined as middle of the rear axle of the host vehicle aligned with vehicle's center axis) in Cartesian coordinates. In consequence, a radar model, receiving a GT OL as an input and outputting a radar-specific OL would be defined as an *object-based radar object model*. A lidar model on the other hand, with a preceding ray casting/tracing and lidar detections (a.k.a. lidar point cloud (PCL)) as an output would read *reflection-based lidar detection model*.

To complete the model description, the modeling approach is prefixed. The terms "stochastic", "physical" or "phenomenological" are frequently used in the sensor modeling field. Based on the individual mathematical representation of the sensor measurement principle in the model it is for example described as *stochastic object-based radar object model*. Definitions and a more detailed discussion on the distinction of terms can be found in [15,21]

3 Modular Approach for Perception Sensor System Simulation

Following these definitions we propose a sensor system model architecture that serves the defined interfaces. The architecture is developed in close regard to the current state of the art in sensor system simulation models.

3.1 State of the Art of Perception Sensor System Simulation

To use simulation models of perception sensors, usually a simulation tool is needed to provide the ground truth in every simulation time step. Multiple tools are available (commercial or open-source, e.g. [22,23,24,25,26,27,28]) that in addition already provide

parameterizable models to either generate synthetic lidar PCL data, radar detection lists, camera images or OLs with different levels of fidelity [29]. However, by serving only one of these output interfaces at once, they can be used by different use cases separately, but do not scale in levels of fidelity. These models are self-contained black box models, where individual parts of signal processing are not interchangeable. Therefore a module of the signal processing chain itself cannot be the system under test.

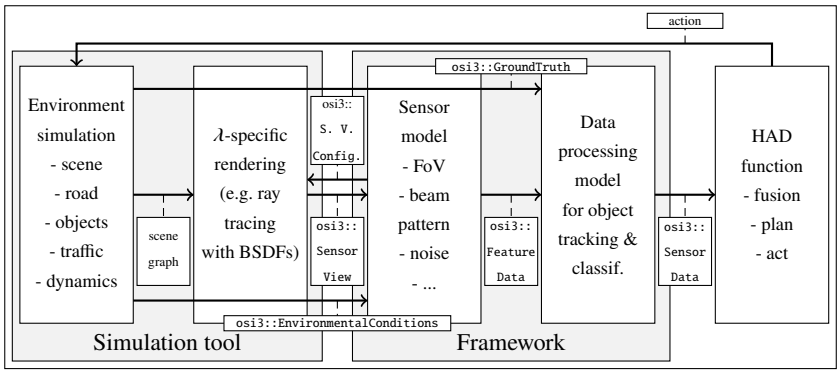
Different physical rendering methods are in use to generate synthetic detections of active perception sensors, like ray casting/tracing methods or projection methods like "z-buffer" as described and compared in previous work of the authors [8]. For simulation on OL level, a stochastic or phenomenological approach can be sufficient, especially in early development phases of higly automated driving (HAD) planning functions, to generate desired uncertainties of existence (e.g. FP- / FN-rates), states (e.g. bounding box dimensions / location), and classes, as they are described by Dietmayer [30].

Nevertheless, scenarios can be found, where stochastic object-based models are easy to falsify by comparison to real world sensor data, as e.g. shown by Aust [31, pp. 17-22]. He describes an intersection scenario, where a vehicle carrying a lidar sensor system approaches an already waiting car from behind. The actual sensor will gain reflections only from the rear of the car in front and the data processing will initialize the corresponding object with either almost no length or a default value for it. A stochastic-only object-based approach will most likely fail in generating the length of the car in front with required fidelity in this case. Only with the availability of detections in an intermediate simulation-step, as in a reflection-based model or with the incorporation of knowledge about detection generation within the front-end in an object-based approach [32], phenomena like sensor perspective dependent dimension uncertainty can be reflected.

To make use of all the different approaches to sensor system modeling, to efficiently reuse already existing model parts and to get on top of the validation problem of parameter space explosion, a modular approach derived from functional decomposition is necessary to achieve perception simulation with valid fidelity. Hanke et al. introduce a modular simulation architecture to facilitate iterative development [33]. Even if they focus on stochastic object-based modeling, they integrate sub-modules to separately reflect individual sensor characteristics. Thieling et al. propose a modular simulation architecture for perception sensor modeling [21], as well. They focus on modularity and scalability to support different sensor technologies and XiL testing approaches from SiL to ViL with growing feasibility of the models and state that "sensor models have to be based on a modular and generalized design which meet the needs of different sensor types (e.g., camera, lidar, etc.)" [21, p. 5]. Even if the publication here focuses on feasible validation and high granularity of the simulation, the trend within the community towards modularity and its benefits is evident.

Therefore, the authors have recently proposed a sequential sensor system architecture at the example of a high fidelity lidar system model [16]. It tackles the current limitations of the state of the art in modularity, interchangeability, distributability, and testability and separates the actual business logic of the simulation model from the outer communication. Additionally, a standard-compliant framework is introduced, which handles all external and internal exchange protocols. Into this framework, sequential strategies [34, pp. 315ff]

Fig. 1: Generic state of the art simulation approach and interfaces for reflection-/object-based perception sensor system simulation [16, p. 8]



can be embedded, which contain the actual business logic of the model. The architecture is demonstrated on the example of a reflection-based lidar detection/object model, as shown in Fig. 1. The framework is packaged as a Functional Mock-up Unit [35, p. 6] (FMU) and exchanges OSI data via the Functional Mock-up Interface [35] (FMI) standard. The business logic of the sensor system model is separated into two strategies for the sensor model and the subsequent data processing for object tracking and classification. The decomposed modules of the complete system incl. the simulation tool and the downstream HAD function are shown in Fig. 1.

The sensor model strategy receives super-sampled detection data, generated by ray casting/tracing, from the simulation tool via the model framework. Proceeding from these detections, a high fidelity PCL is generated considering effects like beam pattern, beam divergence, noise behavior etc. This PCL is passed to the data processing strategy via the `osi3::FeatureData` interface. This strategy then uses GT information to map the high fidelity PCL to the objects from GT and to augment the states of the objects. Thus, a higher fidelity OL is created by simulating the behavior of object detection, tracking and classification algorithms. Simulating the data processing can safe computational resources, but mainly replaces the actual algorithms that are not accessible in most cases. The modular structure still enables the use of individual data processing units e.g. from the sensor manufacturer itself, when available.

In the following section the existing sequential approach of the reflection-based lidar system model is extended to a generic modular architecture for multiple sensor technologies and different model inputs and outputs.

3.2 Modular Approach To Defining a Generic Modular Sensor System Model Architecture

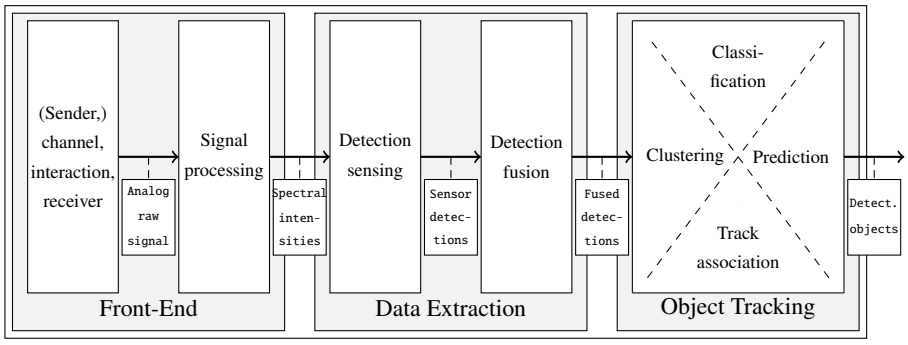
Because of the separation of the outer communication exchange protocols from the actual business logic in the lidar system model architecture of the authors, the generic approach can use the exact same wrapping framework. As the architecture is designed to work with different model types of different sensor technologies, it does not contain any model or sensor specific parts. The modules, of cause, are tailored to a specific sensor technology.

A high-level decomposition yields three main generic parts of any perception sensor system, as shown in Fig. 2. The first part is the *front-end* of the sensor, which contains the transmission unit in case of an active sensor, the propagation channel with all interactions in the scene, and the reception unit as a first module. It outputs an analog raw signal to the signal processing module, which contains for example analog-digital conversion or FFTs in case of radar and yields a low-level signal of spectral intensities. This signal can either be in a time spectrum in a frequency spectrum and can also be multidimensional. It forms the input to the second part: *data extraction*.

The first module of data extraction is the detection sensing. It contains a number of substeps for the different sensor technologies, e.g. thresholding, interpolation or edge/corner recognition to extract detections from the sampled signal. The generic architecture allows sensor fusion on detection level, so the next module is the detection fusion. Here for example the point clouds of two different lidar sensors can be fused. Up to this point, all simulations and calculations where carried out in the sensor coordinate frame. The output of the fusion module however is represented in the vehicle frame. So even if there is no actual sensor fusion, because only one sensor is simulated, the detection fusion module still transforms the detections to the vehicle coordinate system. As already discussed, according to the ISO/DIS 23150 the detections are then called features and are the input for the next part: *object tracking*.

The object tracking part uses features from the current time step k as well as tracked objects and object candidates from the previous time step $(k-1)$ that implicitly contains the whole history of the perceived OL. Even if separation is not completely feasible in all object tracking approaches, the vast majority is composed of the following four subsystems. The first module is the clustering step. It this step features can be clustered and object hypotheses can be formed. These can either be used as an input for track association or clustering and track association can be directly combined in one module. Interlocked with the track association is the classification module, which contributes to the finally outputted detected OL. The prediction step contains one or multiple motion models for the track association of the objects in the next time step.

Fig. 2: Generic functional decomposition of current automotive perception sensor systems for object detection



Depending on the specific model later, some modules can be kept short or skipped entirely. It can also occur, that modules cannot be considered separately, but only as a combined unit. For example in some lidar systems that use analog magnitude comparators analog-digital conversion is done simultaneously to detection sensing. In that case, the boundary between front-end and data extraction becomes somewhat fuzzy. Also, modeling a module from this decomposition does not imply modeling the actual signal processing done in the associated module in a real sensor system. It merely means to model the behavior and the effect of this module on the simulated domain, e.g. detections, features or objects. In Tab. 1 several examples of module outputs for different modeling approaches and sensor technologies are presented.

Module	**Object-based**	**Reflection-based**		**Image-based**
		Lidar	**Radar**	**Camera**
Detection Sensing	BB Vertices	PCL	Radar Detection	Camera Detection
Detection Fusion	Fused BB Vertices	Fused PCL	Fused Detections	Fused Detections (e.g. Stereo)
Clustering	Clustered BB Vertices	Point Cluster (L-Shapes)	Detection Cluster	Clustered Shapes
Track Association	Detected Object			
Classification	Classified Object			
Object Tracking	Tracked Object			

Table 1: Output of modules for different model types (BB: Bounding Box, PCL: Point Cloud)

For an object-based model the vertices of the GT BB can be utilized for the detection sensing calculation by calculating object occlusion with rendering methods. The detection sensing module therefore outputs BB vertices, that are visible to the sensor. In case of multiple sensors, the vertices can be fused, transformed to the vehicle coordinate system, and clustered to account for separability effects in certain sensor systems. The following track association forms object hypothesis and outputs detected objects that can be classified and tracked. In reflection-based models, the first step is to generate the detection level of the respective sensor, e.g. a PCL for lidar, or detections for radar and camera, from the ray casting/tracing results that form the input of the model. Then the detections are fused and clustered as described before. From the track association onward the structure of the module interfaces is exactly the same for all modeling approaches and sensor technologies. This only applies to the interfaces. The content of the modules and of their outputs can of course differ.

4 Parameterization Profiles

Each simulation model is accompanied by a list of parameters. They can either be hard coded into the model, which makes sense, if they are not expected to be changed, or parameters can be defined not right inside the model code, but externally. The

parameterization of the proposed sensor model architecture towards a specific sensor configuration or algorithm behavior is performed with a so called parameterization profile. This opens up the possibility to develop a rather generic model and tailor it to a specific system by parameterization. Another benefit of the latter approach is that model development and parameterization are separated and can be conducted by separate people, separate teams, or separate companies. Defining the parameters externally also enables an automated parameter space exploration and facilitates modular validation methods.

In the original lidar simulation framework architecture of the authors, a possibility to parameterize individual strategies is envisaged [16, p. 6]: The framework is implemented in a way that strategies only rely on the OSI data structure and a set of execution parameters. A not further specified setup class was defined for the purpose of passing parameters to strategies. This setup class is now replaced by a profile approach. A profile contains a set of parameters that defines a specific sensor system and its modeling characteristics. The profiles are separated into structures for each strategy, so the decomposition of the business logic coincides with the structure of the parameter profile. In addition, one structure for general model parameters is defined for the main purpose of setting the sensor view configuration. This configuration is passed to the simulation tool via OSI to parameterize the ground truth input it provides to the model. The structure in Tab. 2 arises from these deliberations.

Parameters
– Model
– Front-end
– Detection sensing
– Detection fusion
– Clustering
– Track association
– Object classification
– Object Tracking

Table 2: Parameter decomposition

The profile structure including the underlying parameters is fixed for one model, the values of the parameters however are defined in individual profiles. Each profile is defined in a separate header file and loaded into the framework once at the start of the simulation. Which profile to use in the individual simulation use case can be set in the model description of the model FMU. Therefore, the end user can choose which profile to use. The parameter values are then passed to the respective strategies by the framework.

To give some more insight how a profile could look like, Tab. 3 contains an exemplary parameter profile for a sensor system comprised of multiple lidar sensors with point cloud fusion and tracking simulation. The model is reflection-based and thus needs to set the sensor view configuration for ray casting/tracing.

Model	**Clustering**
– Rays per beam – Required GT field of view – Number of sensors – Mounting positions	– Reference point for L-shapes – Object separability
Front-end	**Track association**
– Beam pattern – Beam divergence – Maximum range – Range resolution – Wave length – Cycle time	– Association margins
Detection sensing	**Object classification**
– Detection threshold – Interpolation methods	– Object classes – Classification criteria
Detection fusion	**Object tracking**
– IDs of sensors to be fused – Fusion margins	– Default object dimensions – Existence probability threshold – Track birth/death cycle thresholds

Table 3: Example parameterization profile for a reflection-based lidar object model

5 Conclusion and Outlook

In this paper a technology-independent modular model architecture for automotive perception sensors was introduced. The structure is based on the functional decomposition of real world sensors and can facilitate radar, lidar as well as camera sensor models. All interfaces between the individual modules of the model architecture are for the most part already part of internationally recognized standards, or are proposed to be included in those standards. The framework, which wraps the individual decomposed modules, handles all outer communication and enables a profile-based parameterization of the model. Profiles contain parameters for the modules and therefore separate model development from the parameterization effort. This targets towards feasible validation of the model-parameter-combination and enables the end user to tailor a generic sensor system model to the specific system and use case.

In future research the proposed architecture will be used in the development of both object-based as well as reflection-based lidar and radar models. These models will be contributed to the mentioned research projects *SET Level 4 to 5* and *VVMethods* and used for testing of the complete automated driving pipeline. They will also be used to validate individual components like a tracking function as a system under test. This is only possible with a modular approach, because it enables interchanging modules, like the object tracking, between different models. It also allows comparisons between different modeling approaches as well as the application of modular validation methods.

References

1. German Aerospace Center. *PEGASUS Research Project: Securing Automated Driving efficiently*. http://www.pegasus-projekt.info/en/about-PEGASUS, 2017. Accessed: 2020/09/01.
2. AVL List GmbH. *Enable-S3: European Initiative to Enable Validation for Highly Automated Safe and Secure Systems*. https://www.enable-s3.eu/about-project/, 2017. Accessed: 2020/09/01.
3. Deutsches Zentrum für Luft- und Raumfahrt e. V. (DLR). *SET Level 4to5 - Simulationsbasiertes Entwickeln und Testen von Level 4 und 5 Systemen*. https://sl4to5.de/, 2020. Accessed: 2020/09/01.
4. TÜV Rheinland Consulting GmbH. *VVMethoden – Verifikations- und Validierungsmethoden automatisierter Fahrzeuge Level 4 und 5*. http://www.tuvpt.de/index.php?id=vvmethoden, 2020. Accessed: 2020/08/21.
5. Martin F. Holder, Philipp Rosenberger, Hermann Winner, Thomas Dhondt, Vamsi Prakash Makkapati, Michael Maier, Helmut Schreiber, Zoltan Magosi, Zora Slavik, Oliver Bringmann, and Wolfgang Rosenstiel. *Measurements revealing Challenges in Radar Sensor Modeling for Virtual Validation of Autonomous Driving*. In *2018 IEEE International Conference on Intelligent Transportation Systems (ITSC 2018)*, pages 2616–2622. IEEE, 2018.
6. Martin F. Holder, Zora Slavik, and Thomas D'hondt. *Radar Signal Processing Chain for Sensor Model Development*. In Andrea Leitner, Daniel Watzenig, and Javier Ibanez-Guzman, editors, *Validation and Verification of Automated Systems*, pages 119–133. Springer, Cham, 2019.
7. Philipp Rosenberger, Martin F. Holder, Marc René Zofka, Tobias Fleck, Thomas D'hondt, Benjamin Wassermann, and Juraj Prstek. *Functional Decomposition of Lidar Sensor Systems for Model Development*. In Andrea Leitner, Daniel Watzenig, and Javier Ibanez-Guzman, editors, *Validation and Verification of Automated Systems*, pages 135–149. Springer, Cham, 2019.
8. Philipp Rosenberger, Martin F. Holder, Sebastian Huch, Hermann Winner, Tobias Fleck, Marc René Zofka, J. Marius Zöllner, Thomas D'Hondt, and Benjamin Wassermann. *Benchmarking and Functional Decomposition of Automotive Lidar Sensor Models*. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, Paris, France, 2019.
9. Christian Amersbach and Hermann Winner. *Functional Decomposition - A Contribution to Overcome the Parameter Space Explosion during Validation of Highly Automated Driving*. In *26th International Technical Conference and exhibition on the Enhanced Safety of Vehicles (ESV)*, Eindhoven, the Netherlands, 2019.
10. Björn Klamann, Moritz Lippert, Christian Amersbach, and Hermann Winner. *Defining Pass-/Fail-Criteria for Particular Tests of Automated Driving Functions*. In *IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 169–174, Auckland, New Zealand, 2019.
11. Robin Philipp, Fabian Schuldt, and Falk Howar. *Functional Decomposition of Automated Driving Systems for the Classification and Evaluation of Perceptual Threats*. In *13. Uni-DAS e.V. Workshop Fahrerassistenz und automatisiertes Fahren 2020*, 2020.
12. International Organization for Standardization. *ISO/DIS 23150: Road vehicles - Data communication between sensors and data fusion unit for automated driving functions - Logical interface*, 2020.
13. Timo Hanke, Nils Hirsenkorn, Carlo van Driesten, Pilar Garcia Ramos, Mark Schiementz, Sebastian Schneider, and Erwin Biebl. *Open Simulation Interface: A generic interface for the environment perception of automated driving functions in virtual scenarios.* https://www.hot.ei.tum.de/forschung/automotive-veroeffentlichungen/, 2017. Accessed: 2020/09/01.
14. Carlo van Driesten and Thomas Schaller. *Overall Approach to Standardize AD Sensor Interfaces: Simulation and Real Vehicle*. In Torsten Bertram, editor, *Fahrerassistenzsysteme 2018*, pages 47–55, Wiesbaden, 2019. Springer Fachmedien Wiesbaden.

15. Philipp Rosenberger, Jan Timo Wendler, Martin F. Holder, Clemens Linnhoff, Moritz Berghöfer, Hermann Winner, and Markus Maurer. *Towards a Generally Accepted Validation Methodology for Sensor Models - Challenges, Metrics, and First Results*. In *2019 Graz Symposium Virtual Vehicle (GSVF)*, Graz, Austria, 2019.
16. Philipp Rosenberger, Martin F. Holder, Nicodemo Cianciaruso, Philip Aust, Jonas Franz Tamm-Morschel, Clemens Linnhoff, and Hermann Winner. *Sequential Lidar Sensor System Simulation - A Modular Approach for Simulation Based Safety Validation of Automated Driving*. *Automotive and Engine Technology (AAET)*, 2020.
17. *Open Simulation Interface v3.2.0 - OSI "Editorial Eaton"*. https://github.com/OpenSimulationInterface/open-simulation-interface/releases/tag/v3.2.0, 2020. Accessed: 2020/08/20.
18. Fred E. Nicodemus, Joseph C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. *Geometrical considerations and nomenclature for reflectance*. Final Report National Bureau of Standards, 2010. Published: Final Report National Bureau of Standards.
19. Steven G. Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, and Martin Stich. *OptiX: a general purpose ray tracing engine*. *ACM Transactions on Graphics*, 29(4):1–13, 2010.
20. Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. *Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving*. In *2015 IEEE International Conference on Intelligent Transportation Systems (ITSC 2015)*, volume 18, pages 982–988. IEEE, 2015.
21. Jörn Thieling and Jürgen Rosmann. Scalable sensor models and simulation methods for seamless transitions within system development: From first digital prototype to final real system. *IEEE Systems Journal*, pages 1–10, 2020.
22. TWT GmbH. *TRONIS Sensors*. https://www.tronis.de/features, 2020. Accessed: 2020/02/12.
23. TASS Internatinal. *PreScan Sensors*. https://tass.plm.automation.siemens.com/prescansensors, 2020. Accessed: 2020/09/01.
24. Jean-Claude Kedzia, Philippe de Souza, and Dominique Gruyer. Advanced radar sensors modeling for driving assistance systems testing. In *2016 10th European Conference on Antennas and Propagation (EuCAP)*, pages 1–2, 2016.
25. dSpace GmbH. *Probabilistic Sensor Models for ADAS/AD Simulations*. https://www.dspace.com/en/ltd/home/products/systems/simulationmodels/simulation_models_use_cases/probabilisticsensormodels.cfm, 2020. Accessed: 2020/09/01.
26. VIRES Simulationstechnologie GmbH. VTD - VIRES Virtual Test Drive. https://vires.mscsoftware.com/solutions/sensors/, 2020. Accessed: 2020/9/01.
27. IPG Automotive GmbH. *Virtual testing of ADAS*. https://ipg-automotive.com/areas-of-application/adas-automated-driving/virtual-testing-of-adas/, 2020. Accessed: 2020/09/01.
28. CARLA Team. *CARLA - Open-source simulator for autonomous driving research.* https://carla.org/, 2020. Accessed: 2020/08/21.
29. Martin Herrmann and Helmut Schön. *Efficient Sensor Development Using Raw Signal Interfaces*. In Torsten Bertram, editor, *Fahrerassistenzsysteme 2018*, pages 30–39, Wiesbaden, 2019. Springer Fachmedien Wiesbaden.
30. Klaus Dietmayer. *Predicting of Machine Perception for Automated Driving*. In Markus Maurer, J. Christian Gerdes, Barbara Lenz, and Hermann Winner, editors, *Autonomous Driving: Technical, Legal and Social Aspects*, pages 407–424. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.
31. Philip Aust. *Entwicklung eines lidartypischen Objektlisten-Sensormodells*. M. Sc. Thesis, Technical University of Darmstadt, Darmstadt, Germany, 2019.

32. Yifei Jiao. *Implementation of a generic perception sensor simulation model for object list output*. M. Sc. Thesis, Technical University of Darmstadt, Darmstadt, Germany, 2020.
33. Timo Hanke, Nils Hirsenkorn, Bernhard Dehlink, Andreas Rauch, Ralph Rasshofer, and Erwin Biebl. *Generic architecture for simulation of ADAS sensors*. In *2015 16th International Radar Symposium (IRS)*, pages 125–130, Dresden, 2015. IEEE.
34. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
35. MODELICA Association, Project FMI. *Functional Mock-up Interface for Co-Simulation*. https://svn.modelica.org/fmi/branches/public/specifications/v1.0/FMI_for_CoSimulation_v1.0.1.pdf, 2017. Accessed: 2020/09/01.

Acknowledgments

This work received funding from the projects *SET Level 4to5* and *VVM*, as part of the PEGASUS project family, promoted by the German Federal Ministry for Economic Affairs and Energy based on a decision of the Deutsche Bundestag.