WMG
THE UNIVERSITY OF WARWICK

AURRIGO

AESIN
AUTOMOTIVE ELECTRONICS INNOVATION

Simulation **Verification & Validation**

zenzic.io

ZENZIC
SELF-DRIVING REVOLUTION

Disclaimer

This report has been written by WMG, at the University of Warwick with contributions from AESIN and Aurrigo. Zenzic have compiled the report. Any views expressed in this report are not necessarily those of Zenzic.

The information contained herein is the property of these organisations and does not necessarily reflect the views or policies of the customer for whom this report was prepared. Whilst every effort has been made to ensure that the matter presented in this report is relevant, accurate and up-to-date, Zenzic and/or any of the authors of this report cannot accept any liability for any error or omission, or reliance on part or all of the content in case of incidents that may arise during trialling and testing. In addition, Zenzic and/or any of the authors of this report cannot accept any liability for any error or omission, or reliance on part or all of the content in another context.

When in hard copy, this publication is printed on paper that is FSC (Forest Stewardship Council) and TCF (Totally Chlorine Free) registered.

For further information on this report please contact the Zenzic team

Email: info@zenzic.io
Web: zenzic.io

Acknowledgements

- Principle Investigator – Dr. Siddartha Khastgir, Head of Verification and Validation, WMG
- Principle Author – Dr. Jason Xizhe Zhang, Simulation Lead, WMG
- Technical delivery – WMG, supported by AESIN and Aurrigo
- Technical oversight and approval – Zenzic

Special thanks also to the Advisory Group for guiding and shaping this work, particularly regarding the stakeholder engagement activities. The advisory group comprised: Department for Transport (DfT), Driver and Vehicle Standards Agency (DVSA), Vehicle Certification Agency (VCA), Institute of Digital Engineering (IDE), Connected Places Catapult, BSI, Oxfordshire County Council, Ordnance Survey, Oxbotica, IPG Automotive, Aurrigo, University of Leeds Institute for Transport Studies, Centre for Connected and Autonomous Vehicles (CCAV), Department for Business, Energy & Industrial Strategy (BEIS) and Nissan.

Contents

Executive summary

The safe introduction and adaptation of automated/assisted driving technologies requires extensive testing. Distance-based and real world-based testing approaches have been suggested to be challenging to fulfil such extensive testing requirements. Recent developments within the industry and academia have led towards a simulation-based and scenario-based testing approach. Under such approach, virtual environments are used alongside the physical environment, and purposely derived test scenarios are used instead of driving extensive distances to provide safety evidences of systems. The Zenzic Simulation Verification and Validation (V&V) project aimed at answering how simulation, together with the use of scenarios, can be used for the testing of automated driving technologies. Furthermore, it highlights the challenges associated with such approach and how the Zenzic V&V project addresses such challenges.

The content presented in this report can be broadly divided into three distinctive aspects:

1 From a **system testing** point of view – providing that the simulation environments are reliable and are comparable to the physical operational environment of the system, the question was addressed as to how they can be integrated into the system testing workflow. Furthermore, given the importance of the scenario for system development and testing, how to design and implement a workflow that utilises both simulations and scenarios was addressed. This aspect, therefore, introduces a novel scenario-based testing process, within which the scenario creation, scenario description format, scenario database, scenario retrieval, scenario execution, and scenario analysis are explained. Eight different scenario creation methods, using both data-driven and knowledge-driven approaches, are introduced. A novel, flexible, and readable scenario description format is also presented, together with its conversion to open standards for wider toolchain compatibility. A powerful and rich scenario database was also utilised during the project, for organising the related scenarios, and offering an efficient way to retrieve test scenarios. During execution, optimisation algorithms were implemented to explore the scenario parameter spaces and find the edge cases. Euro NCAP scenarios were used during the project to test a target System Under Test using the scenario-based testing workflow within a simulation environment.

2 A main assumption outlined in Aspect 1 is that simulation environments are comparable to the physical environment, which is fundamental to any simulation-based testing activities. However, across the industry and academia, limited evidence has been shown as to how simulation validation can be carried out. The second aspect, therefore, focuses on the simulation validation processes. The whole validation is divided into dynamic elements and static elements. Dynamic elements validation focuses on the validation of dynamic agents, ranging from macroscopic level, to vehicle level, to sub-system level, and to sensor level. The static elements validation, on the other hand, focuses on validating the scenery representation within the simulation. A novel approach to static element validation is presented within this report, which compares lidar scans of the real world with a digital twin to draw conclusions.

3 The third aspect documents the stakeholder engagement activities carried out within the V&V project. Within Aspect 3, key and relevant contributors are identified and experts' views on simulation V&V are summarised which highlight the importance of simulation V&V as well as

the challenges. The outcomes of the stakeholder engagement activities have fed into the *UK Connected and Automated Mobility Roadmap: Simulation to 2030*, which can be found through the Zenzic website – zenzic.io.

1 | Introduction

1.1 Background

The recent advancement in Automated Driving Systems (ADSs) and advanced driver assistance systems (ADASs) technologies is driven by the many benefits they offer, such as increased road safety (Fagnant and Kockelman, 2015), increased traffic throughput (Le Vine *et al.*, 2016), reduced emission levels (Fagnant and Kockelman, 2014), and decreased driver workload (Balfe, Sharples and Wilson, 2015). Along the development cycle of a system, a key and necessary stage is its safety assurance. Ensuring safe introduction and public trust are important factors for their acceptance and scalable adoption. However, the complexities associated with ADSs and ADASs, and their interactions with the environment pose great challenge for their safety evaluation. Traditional approaches for testing vehicles utilises a distance-based approach – i.e., 'X' miles driven without accident, usually the higher the X number the safer the system is. Such approach is no longer effective when applied to ADSs and ADASs due to two main reasons: 1) the number of miles required for ADSs and ADASs is too high, 2) the lack of consideration of a system's operational design domain (ODD). Kalra et. al suggested that for ADSs they would need to be driven 11 billion miles to demonstrate they are 20% better than human drivers (Kalra and Paddock, 2016), translating this number of miles into a time-scale would require a fleet of 100 vehicles driving 24 hours a day, 365 days a year at a speed of 25 miles per hour to operate for 500 years! In addition, for ADSs and ADASs, the ODD is another crucial factor in their safety assurance and safe deployment. ODD defines the scenery, the environment and the dynamic conditions within which the system is designed to operate ('Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification', 2020); being able to operate safely within a system's ODD is crucial. Now assuming a system is designed for inter-urban motorway use cases, if such system was tested for 11 billion miles (or 500 years – hypothetically) in a setting that only contains minor roads in a dessert, such effort would not be sufficient to claim the system can operate safely on motorways.

This has led to the hazard-based testing approach [7], which focuses on the quality of the 'miles' driven rather than the quantity, concentrating on how a system fails rather than how it works. One key enabler for achieving this is through the use of scenarios, where each scenario describes a set of scenery, environmental, and dynamic conditions within which the system will be tested [8], this results in a scenario-based testing approach. Furthermore, with the advancement in computational power in recent years, industry and academia have adopted a hybrid scenario execution approach containing both virtual and physical environments.

1.2 Motivation

Due to the novelty of the scenario-based evaluation approach and simulation-based testing, industry and academia have made a significant effort to bring up the maturity of such methodologies in recent years, both nationally (e.g., OmniCAV (Brackstone *et al.*, 2019)) and internationally (e.g., ASAM Simulation domain standards (*ASAM e.V.*)). To utilise the simulation environment as a testing environment, a key assumption is that the simulation and real world are comparable; this has led to the topic of simulation validation. Current work (ongoing) is taking place within the UNECE (United Nations Economic Commission for Europe) Validation

Method for Automated Driving Standards Group (VMAD SG2) on developing a credibility assessment framework for simulation and virtual testing. The Zenzic V&V work documented here is a collaboration between government and industry partners to further develop and contribute towards the scenario-based and simulation-based testing based on the current state-of-the-art within the industry. The key focuses for the V&V work are: 1) the introduction of the major components along a scenario-based evaluation process, 2) the implementation and integration of such an evaluation process with the rest of the Zenzic Interoperable Simulation ecosystem, 3) the establishment of a simulation validation approach for both the dynamic and static elements, 4) creation of the *UK Connected and Automated Mobility Roadmap: Simulation to 2030* in collaboration with Zenzic.

1.3 How to read this report

The document is organised as follows:

1. Section 2 will introduce the scenario-based evaluation framework at the conceptual level, together with the illustration of all the key components.
2. Section 3 will convert and expand the evaluation framework at the functional and implementation levels.
3. Section 4 will illustrate how, in collaboration with the Zenzic Phase 3 Interoperable Simulation project, the V&V implementation framework was integrated with the rest of the Zenzic Interoperable Simulation ecosystem, and how testing has been conducted utilising the framework.
4. Section 5 will introduce the simulation validation methodology, including both the static and dynamic elements within a simulation environment.
5. Section 6 will illustrate how the current results, and stakeholder engagement, have formed the roadmap.

2 | The V&V evaluation continuum

2.1 Terms and definitions used

Before diving deep into the scenario-based evaluation process, it is important to understand what a scenario is, the different types of scenarios, how they are relevant to this project, and the terminologies used.

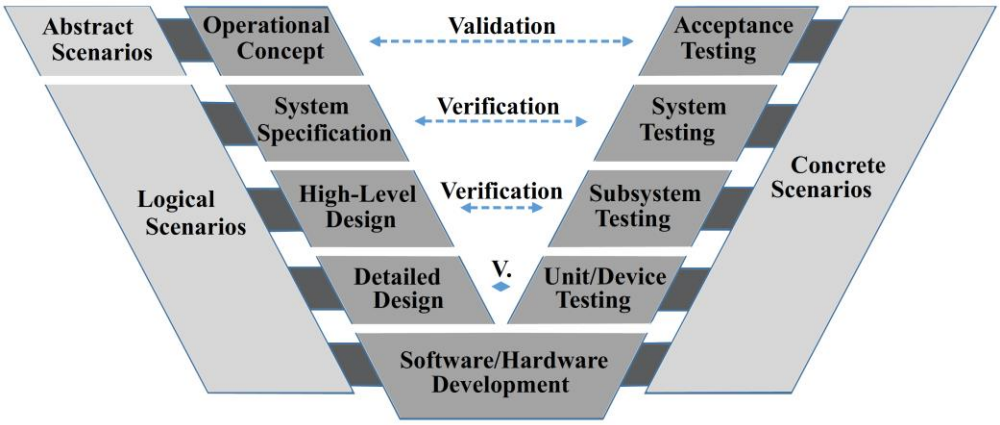
A commonly used definition for scenario was proposed by Ulbrich et al (Ulbrich *et al.*, 2015) as below:

'A scenario describes the temporal development between several scenes in a sequence of scenes. Every scenario starts with an initial scene. Actions & events as well as goals & values may be specified to characterize this temporal development in a scenario. Other than a scene, a scenario spans a certain amount of time.'

Within such definition, a scene was further described as an instantaneous snapshot of the scenario conditions including the scenery, dynamic elements, and environmental conditions.

Within a scenario-based development and testing process, purposely defined scenarios are used throughout the whole V development model - operational concept definitions, specification and design, development, and testing - as shown in Figure 2.1 (Bock *et al.*, 2019). Different types of end users sit at different locations along this V model; ADS developers might favour a common format in order to share across organisations and systems, raising the need for standardisation effort. Simulation test engineers might favour a highly detailed, fully parameterised, machine-readable format for their execution and analysis. Regulators and the public might favour a higher abstraction level and human readable natural language format. Due to the different abstraction levels along the V development cycle, different scenario levels have been introduced within industry and academia. Menzel et al (Menzel, Bagschik and Maurer, 2018) firstly introduced the three scenario abstraction levels: **functional scenario**, **logical scenario**, and **concrete scenario**. Functional scenario operates the scenario at a semantic level, the entities and their relation are described via a linguistic notation. Logical scenario targets the scenario content at a state space level, it represents the entities and their relations using parameter ranges. Concrete scenario also targets scenario content at a state space level, it represents the entities and their relations using concrete values. As an addition, Neurohr et al. (Neurohr *et al.*, 2021) recently extended the three levels of scenarios by a fourth level – **abstract scenario**. Abstract scenario sits in between the functional scenario and logical scenario, it utilises a formalised format to organise the scenario descriptions, usually adopting a natural language–based expression (Bock *et al.*, 2019)(Zhang, Khastgir and Jennings, 2020a).

Figure 2.1, V-model for system development



Similar classification of the scenario abstraction level is also published by Khastgir et al (Khastgir *et al.*, 2017), within which the terms **use case**, **test scenario**, and **test case** were introduced. A use case describes the system behaviour as a sequence of actions linking the result to a particular actor. A test scenario is a specific path through a use case, i.e., a specific sequence of actions. A test case is a set of test preconditions, inputs, and expected results, developed to drive the execution of a test item to meet test objectives, including correct implementation, error identification, checking quality, and other valued information.

2.2 Scenario-based evaluation continuum overview

Figure 2.2 illustrates the key components of a scenario-based evaluation continuum; it sub-divides the content into functional steps, main elements, and the related context. As the main elements, the whole process only needs **scenario**, **environment**, and **certification**. Every main element is then mapped to their corresponding functional steps at the top, and their related contexts are referred to at the bottom. The workflow is independent from the test execution environment, and is applicable for simulation execution, real-world execution as well as X-in-the-Loop (XiL) testing. The central information carried along this workflow is the scenario; relevant scenario content is created, processed, and assessed throughout the whole process, which forms the overall scenario-based V&V framework. The following sub-sections will further detail each of these main elements to provide more insight.

2.3 Scenario

The **scenario** within the main elements row in Figure 2.2 is located at the upstream of the workflow; from it the structured scenario artefacts, together with pass/fail criteria, are created. The scenario element includes three sub-processes: create, format and store.

The create sub-process represents the creation of scenario content; scenarios can be created using two different approaches – knowledge-driven and data-driven (Zhang *et al.*, 2021), (Zhang, Khastgir and Jennings, 2020a), (Menzel *et al.*, 2019). Furthermore, the scenario creation can be tailored towards different related contexts as shown in the bottom row, for example system engineering, safety, cyber security and in-service testing. The created scenarios at this stage do not need to be represented using any specific format, they can simply use any raw output data format from the generation pipeline. After the scenario generation, the format sub-process will then convert the raw scenario output content into a human and machine-readable format. Subsequently, the formalised scenario content will then be stored in a scenario database for storage, sharing, analysis and query purposes.

2.4 Environment

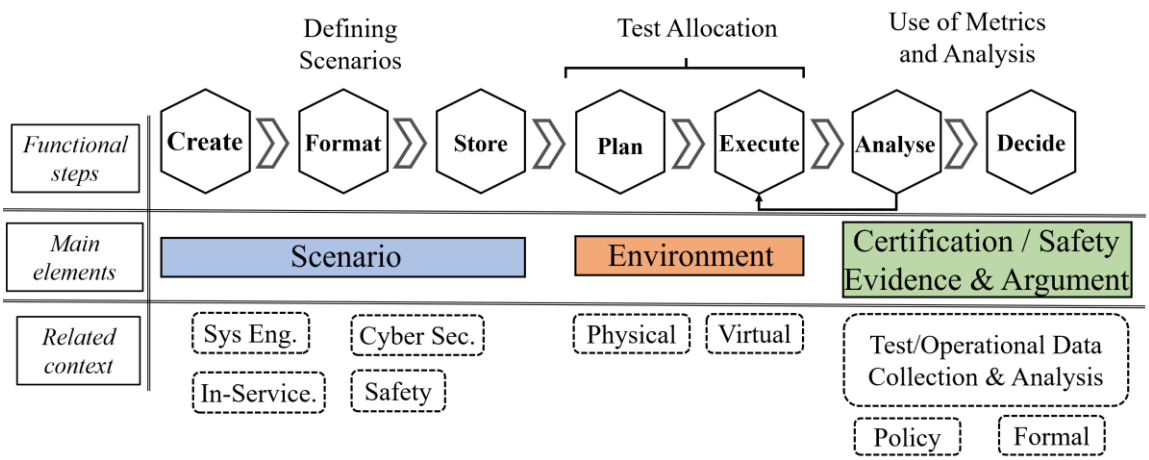
The **environment** within the main elements row in Figure 2.2 contains different options of the execution environment, such as real world, simulation or a hybrid of the two. Test allocation is a key step within the environment element. This step entails the allocation of test scenarios to be executed in different environments. Once the allocation or the test plan has been created, the next step is to execute and analyse the scenario.

2.5 Certification/safety evidence & argument

The **certification/safety evidence & argument** within the main elements row in Figure 2.2 contains analyse and decide. Analyse can be further divided into 3 separate stages:

1 Correct execution - whether the intended test case has been executed.

Figure 2.2, Elements within the scenario-based evaluation continuum



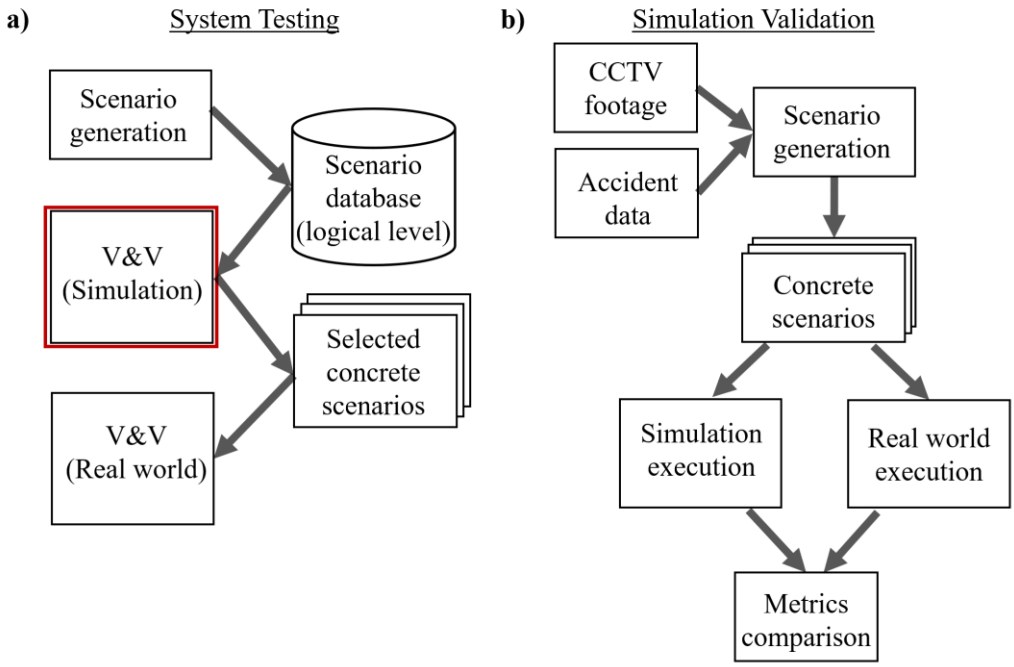
2 Pass/fail assessment – monitoring the execution of the scenario and assessing the runtime output against a set of pre-defined pass/fail criteria/metrics.

3 Scenario parameter space exploration – based on the current and past concrete parameters (e.g., speed, acceleration) and the pass/fail criteria, a test case generator, such as an optimisation algorithm, can be applied to introduce a new set of test case parameters with the aim of violating the scenario pass criteria.

The output from the test case generator will result in the creation of new test cases which can then be fed back into the execution module. This is indicated by the arrow going from analyse to execute in Figure 2.2, and it forms a closed feedback loop within this workflow. This allows the increase of scenario coverage, the decrease of the 'unknown unsafe' region and the addition of new test cases into the database. The final stage is the decide stage, based on whether the intended test cases have occurred, the assessment on the pass/fail criteria and whether the scenario coverage was achieved. This stage will determine the output of the whole V&V process.

2.6 Simulation validation vs system testing

Figure 2.3, Simulation and real-world based testing used in conjunction



Before going into the V&V evaluation continuum at the functional and implementation level, it is important to differentiate between simulation validation and system testing.

As illustrated, the execution of scenarios can be achieved in a virtual environment, the real world, or a hybrid of the two. As the quantity of scenarios increases, being able to decide how they are distributed across the available environment settings for their execution becomes crucial. Real-world execution ensures that the surrounding environment is consistent with the deployment environment, it can also ensure that identical systems (i.e. hardware and software) are tested between testing and deployment. However, real-world execution can be expensive, risky, and time consuming, for example, waiting for a specific weather condition to occur, or involving

hazardous conditions for the system under test (SUT). Simulation-based testing can be used in conjunction with more 'expensive' real world testing (Zhang *et al.*, 2021) to form a comprehensive testing strategy, as shown in Figure 2.3a. In this case it acts as an initial phase of the testing to identify a small set of high interest scenarios from a large number of input scenarios (which might be impossible to execute in the real world), this small set of high interest scenarios can then be executed in a real world environment. However, to enable such a workflow, or rather to utilise any simulation environment for testing, a fundamental assumption is that simulation and real-world environments are comparable or correlated. Without this assumption, simulation-based activities carry little meaning.

To validate the simulation environment, the process can be divided into two aspects – dynamic and static. The dynamic aspect includes all the scripted and non-scripted agents, as well as the SUT. Scripted agents are the individual agents whose behaviours are specified in detail; they interact with the SUT directly within a scenario. Non-scripted agents refers to the macroscopic traffic behaviour, only the high-level goals and characteristics are specified, how they navigate or behave is determined by an external traffic model. The SUT can be further divided into system level and sensor level components. This results into four levels of dynamic validation: macroscopic traffic level, vehicle level, system level, and sensor level. Based on such dynamic validation levels, different approaches can be used to validate them. For example, sensor level might require the validation of the graphical realism of the simulation, whereas vehicle level or macroscopic level might only require the comparison of the trajectories.

On the other hand, the static element validation concerns how accurate the environment simulator is; the term environment simulator refers to the simulated world, or 'scenery' based on the definition from BSI PAS 1883 ODD taxonomy ('Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification', 2020). For example, if there is a building within the scenery in the simulation world, how does it differ to the vehicle sensor compared with the same building in the physical world.

The high-level workflow for the simulation validation can be found in Figure 2.3b. It can be seen that similar scenario generation approaches as the system testing can be used to create scenario descriptions based on real-world data input. Since the goal of this workflow is not to test the system, only a small number of scenarios are needed for the simulation validation (with rich ODD elements and diverse behaviour characteristics). The same scenarios will be executed both in simulation and in the real world; subsequently their execution output will be compared to derive comparison metrics. Section 5 will outline more details on the simulation validation.

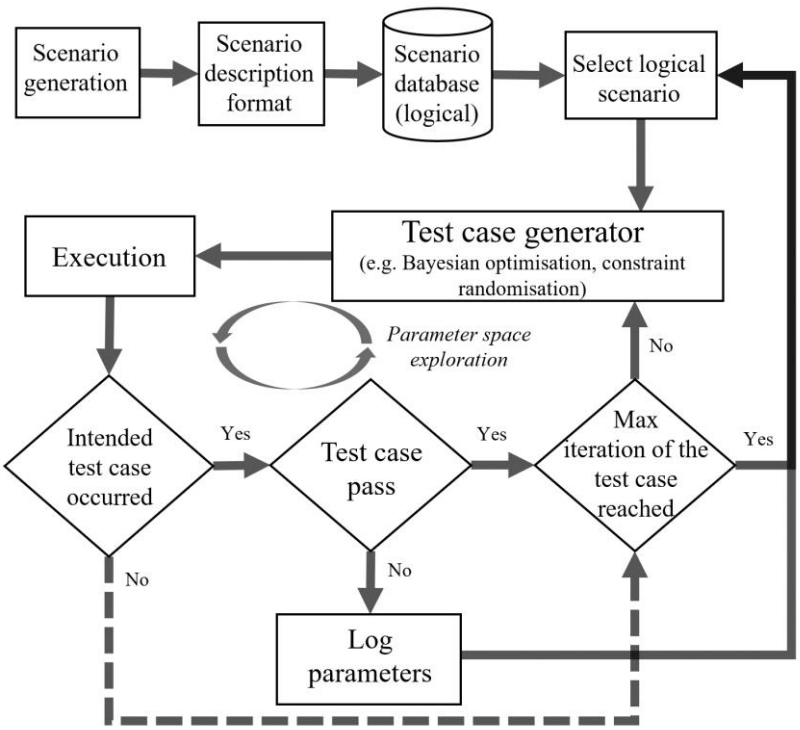
3 | The V&V framework at functional and implementation levels

This section will carry on from the V&V framework at the conceptual level as illustrated in Section 2 and dive deeper into the functional workflow and implementation architecture. Each of the elements within the functional steps in Figure 2.2 will be expanded, plus the decision logic involved will be illustrated.

3.1 Scenario-based evaluation at functional level

Building on the scenario-based evaluation continuum, each of the blocks can be expanded further to result in a workflow at the logical level. Figure 3.1 illustrates the functional level workflow of the scenario-based evaluation framework. Scenarios are generated and described using a human and machine-readable format at the logical scenario level, which are then stored in the Safety Pool™ scenario database (*Safety Pool Scenario Database*) ready for query for testing via API. Optionally, other desired executable formats such as ASAM OpenDRIVE (Dupuis, Hekele and Biehn, 2019) and ASAM OpenSCENARIO 1.x (ASAM OpenSCENARIO 1.1.1) can also be generated within the scenario description format box; they are then attached to the scenario database. A scenario selector is implemented for performing the API calls; it iterates within a specific scenario library and retrieves individual logical scenarios and the OpenX files attached to it. The test case generator is then used to generate concrete test case parameters. Upon execution, the test case data is processed and checked against three decision modules. The first one is whether the intended test case situation occurred. If yes, then the test case pass/fail criteria are checked. If no, then the test case run is checked against a pre-defined maximum iteration number of test cases. The test case pass/fail criteria module consists of multiple types

Figure 3.1, Scenario-based testing workflow at the functional level



of criteria sources. If a test case fails the criteria, then its parameter combination is recorded, and the current logical scenario testing is terminated. If a test case passes, then it will be checked against the maximum iteration limit. In the last step, if the maximum iteration is not reached, the current test case parameters together with the pass criteria will be fed into the test case generator where algorithms such as Bayesian optimisation (Gangopadhyay *et al.*, 2019) can be applied as a "concretiser" to introduce new parameters with the goal of driving the SUT to violate the pass criteria. The closed loop formed by the test case generator, the test execution, and the three test case checks enables the exploration of the parameter space set out within the logical scenarios, while increasing the test coverage and reducing the 'unknown unsafe' territories. The sub-sections below illustrate the details within each of the boxes in Figure 3.1.

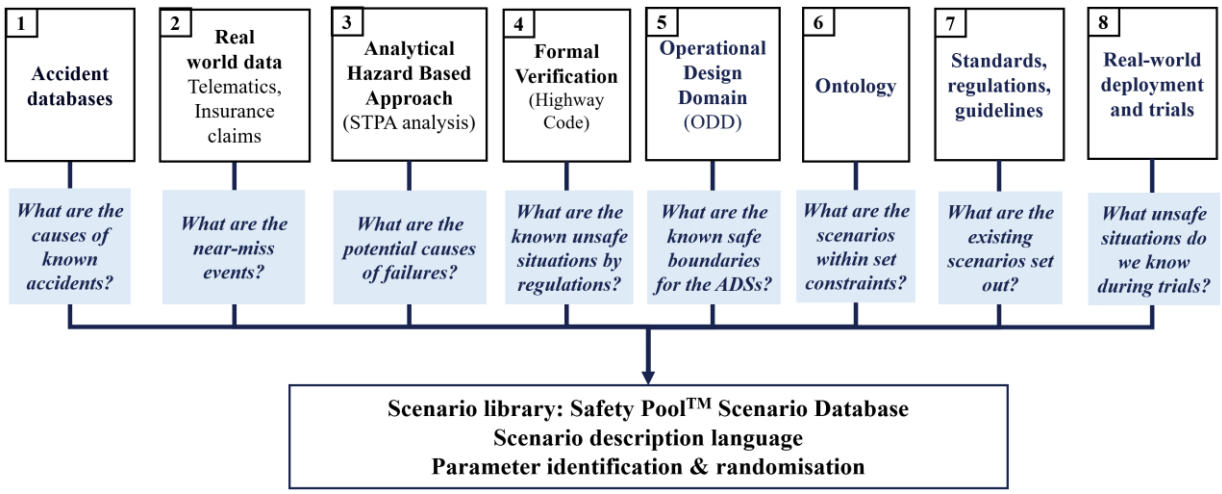
3.2 Scenario generation

As mentioned before, there are two different approaches for scenario generation: knowledge-driven and data-driven [7]. A knowledge-driven scenario generation approach utilises domain specific knowledge to identify hazardous events systematically and creates scenarios. A data driven approach utilises the available data to identify and classify occurring scenarios. Within the scenario generation work at the WMG V&V team, eight different methods for scenario generation have been used, as shown in Figure 3.2. Each approach aims at answering a specific question to ensure that the scenarios are rich and varied, and they cover real world use cases related to the questions being addressed. Upon the generation of scenarios, they feed directly into the Safety Pool™ scenario database for future maintenance and usage.

Option 1 Accident database – *what are the causes of known accidents?*

For option 1, both publicly available and private (if applicable) accident datasets are used as input to the scenario generation pipeline to create identical scenarios, or similar scenarios but not identical. One example is the UK STATS19 (*UK Department for Transport*, 2020) accident dataset, which was studied previously (Esenturk *et al.*, 2021) to identify accident hotspots and scenario parameters that contribute to the causation of accidents. The severity of the accidents recorded can be used as a filter such that characteristics of scenarios that lead to accidents of

Figure 3.2, Scenario generation methods at WMG



specific severity can be extracted. Analysing severe incidents provides useful data in trends that lead to dangerous scenarios.

Option 2 Insurance claim records – *what are the near-miss events?*

For option 2, insurance records can be utilised as another data source input to the scenario generation pipeline to re-create the recorded scenarios, or to generate scenarios in a similar category. Previously at WMG, anonymised insurance claim records provided by a project partner were used to identify trends and explore the parameter combinations in the near-miss events that led to the claims. Such scenarios can be utilised to study these near-miss situations and provide better awareness for the system and system developers.

Option 3 Analytical hazard based approach – *what are the potential causes of failure?*

Different from option 1 & 2, option 3 is a knowledge-driven scenario generation method; it utilises experts' knowledge on a specific domain, which in this case is the ADS internal system architecture, to generate scenarios. As an example, at the WMG V&V team, the Systems Theoretic Process Analysis (STPA) was performed on several automated driving stacks to identify potential system failure nodes and hazardous situations (Chen *et al.*, 2020). The analysis can then be converted into a set of logical scenarios together with their corresponding pass/fail criteria and input into the scenario database.

Option 4 Formal methods – *what are the known unsafe situations by regulations?*

Option 4 uses the formal analysis approach, utilising the highway code rules for scenario generation. Each of the highway code rules describes a hypothetical driving scenario with the corresponding behaviour, scenery, and environmental elements. Through a formalisation process (involving consultation with the relevant authorities), the traffic rules are converted into unambiguous, machine-readable logical statements. Upon formalisation, further analysis can be performed to identify the combination of scenario parameters that will result in a violation or unsafe situation as defined by the regulation.

Option 5 ODD – *what is the known safe boundary for the ADS?*

Option 5 uses either the system ODD, or the associated ODD elements along a designated operating route as the input for scenario generation. Based on such ODD input, the system's corresponding permitted behaviours can be filtered out from the behaviour library which contains all the manoeuvres as well as communicating behaviours. This results in a list of ODD elements, together with applicable behaviours. By further using an ontology model and a set of rules for constructing logically viable scenarios (e.g., overtake is not valid when the overtaking vehicle starts in front of the vehicle being overtaken), individual scenarios can be generated. By using such a method, a large number of scenarios can be generated across the whole pre-defined ODD space, ensuring a sufficient coverage.

Option 6 Ontology – *what are the scenarios within a set of constraints?*

Option 6 uses an ontology-based scenario generation approach. An ontology defines all the classes (e.g., Car, Road) within the domain. It also includes all the relationships (e.g., is on, faster than) between classes and all the pre-defined rules. An example rule can be 'if road A is connected to road B, then the width of road A must equal to the width of road B'; such rules will ensure the correct instantiation of a scenario. By using: 1) a well-developed ontology, with the associated classes, relationships, and rules; 2) a highly abstract scenario description of interest at the functional level or a set of conditions, the abstract information can be converted and detailed into a large number of logical scenarios that satisfy the pre-defined conditions.

Option 7 Standards, regulations and guidelines – *what are the existing scenarios that have been established?*

In addition to the above-mentioned scenario generation methods, existing scenarios already defined in standards, regulations or guidelines can also be utilised for the testing of an ADS, for example the scenarios set out in ISO22737 on Low-Speed Automated Driving (LSAD) (ISO, 2021), European New Car Assessment Programme (Euro NCAP) , and the Automated Lane Keeping Systems (ALKS) regulation (UNECE, 2020). ISO22737 has been developed for LSADs, the Euro NCAP provides a set of testing scenarios for the safety assurance of vehicles, and the ALKS regulation aims at establishing uniform provisions concerning the approval of vehicles with regard to the ALKS functions.

Option 8 Real-world deployment and trials – *what unsafe situations do we know have occurred during trials?*

Option 8 includes the scenarios that occur during real-world trials and deployments. Such scenarios might not have been considered pre-deployment, but are key learnings that can be fed back into the scenario generation process.

3.3 Scenario format

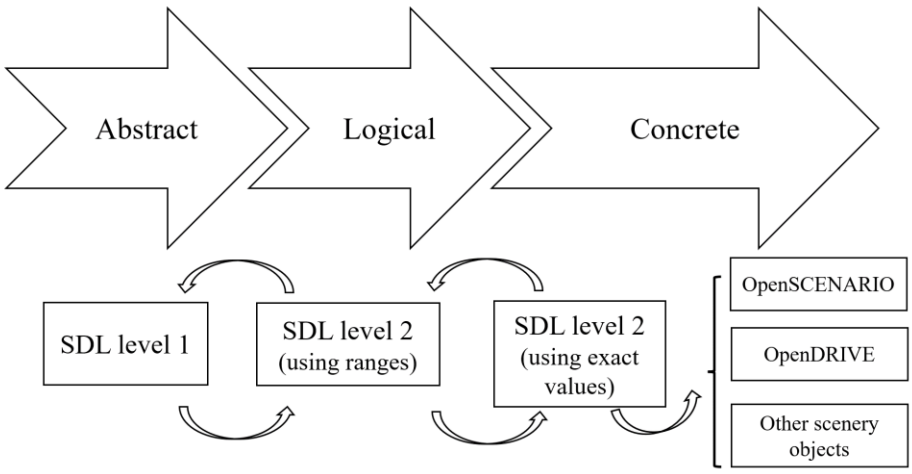
After generating the scenario content, an adequate scenario description format is used to represent the content and enable its sharing and execution. A two-level abstraction approach of scenario description format has been previously developed, published, and maintained by the WMG V&V team (Zhang, Khastgir and Jennings, 2020b). As illustrated earlier, the role of scenarios can be seen throughout the system development and testing cycle, therefore a diverse set of end users as well as competing requirements on the description format are analysed when forming the language concept. End users such as ADS developers, test engineers, regulators, and the public are considered; they sit at different positions along the V model.

Previously several types of end users along the V development cycle have been illustrated, and their requirements on the scenario itself apply at different abstraction levels. Although there are synergies among these requirements, some are also competing (such as executability vs high level abstraction), this resulted in the **two-level abstraction** approach for the Scenario Description Language (SDL). SDL level 1 at the abstract scenario level, is abstract and contains less information; its syntax closely resembles a spoken natural language format under a structured grammar. SDL level 2 sits at the logical and concrete scenario levels, it uses a formal

machine-readable format. By a detailing process, one can convert SDL level 1 into level 2; and by abstracting, the opposite can be achieved. In addition, a conversion toolchain has been developed at the WMG V&V team to convert between SDL level 2 and the ASAM OpenX formats (OpenSCENARIO 1.x and OpenDRIVE) for wider tool support, as shown in Figure 3.3. OpenSCENARIO1.x and OpenDRIVE are low level xml-based formats which can be executed across a variety of simulators. The drawbacks of them are that: 1) they are not human readable, and therefore cannot satisfy certain user requirements, 2) OpenSCENARIO 1.x covers the dynamic and environmental aspects, and OpenDRIVE covers the road network, this leaves part of the ODD elements uncovered under neither standards, such as trees and buildings. Although a user could add such additional ODD elements within the object description in OpenDRIVE, such descriptions, from a language point of view, could benefit from further development and alignment with ODD requirements. By enabling the conversion between the WMG SDL and OpenX format, a gap in the current scenario description formats can be fulfilled, as outlined by the 'Simulation Scenarios' study (Knabe, 2019). This gap resulted from a missing link between high level languages and low level formats (OpenSCENARIO1.x and OpenDRIVE).

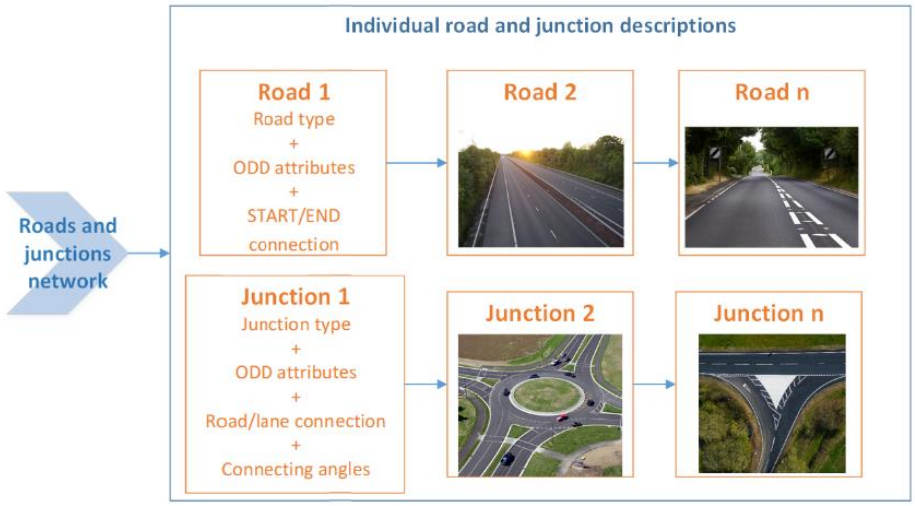
The domain model of the SDL covers the scenery, the environmental conditions, and behaviours of any non-ego agents; in here 'ego' refers to the SUT. Since ODD plays an important role in the safety assurance and safe deployment of ADSs, the domain elements that cover the scenery and environmental conditions are referenced to the BSI PAS 1883 on ODD taxonomy ('Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification', 2020). This interlink between ODDs and scenarios facilitates a common, coherent, and efficient testing and development process. For the behavioural elements of the SDL, they are divided into manoeuvres and agent type. Manoeuvres further divides into relative manoeuvres and absolute manoeuvres. Relative manoeuvres are the manoeuvres that require more than one entity to perform, for example 'Cut-In' is a relative manoeuvre which requires two entities. Absolute manoeuvres are the manoeuvres that a single entity can perform, an example of this is 'Drive' which does not indicate relations to others. Agent type, at a high level, includes vehicles, pedestrians, and animals.

Figure 3.3, Two-level scenario description language mapped to the scenario abstraction levels



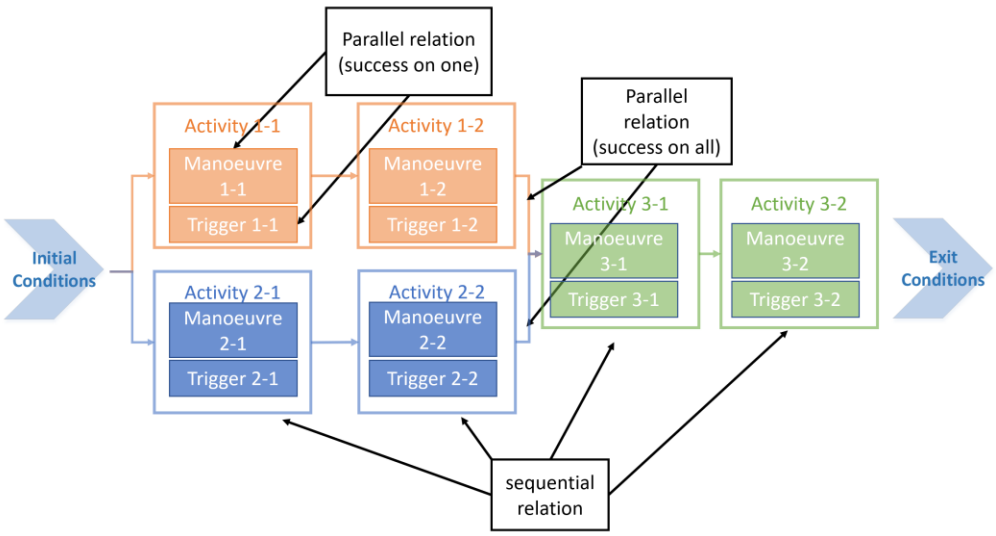
For the scenery aspects, the SDL considers any scenery settings as a roads-and-junctions network. Each road or junction is described individually using their types and the associated ODD attributes. In addition, for each junction, the connecting roads and lanes as well as connecting angles are also required. This can then be referenced to the individual road description and allow the composition of the entire scenery. Figure 3.4 provides an example visualisation of the SDL scenery construct.

Figure 3.4, A simple illustration of the SDL scenery composition



For the behaviour description, the overall structure contains two parts: initialisation phase and manoeuvres phases. The initialisation phase sets out the initial road and lane position for each dynamic entity; the relative heading angle and relative position between them can also be defined. For the manoeuvre phases, a behaviour tree style description format is utilised. Each dynamic entity contains multiple activity phases in a sequential relation; when two dynamic entities are performing activities at the same time the activity phases between them will be in a parallel relation. Each activity phase consists of the actual manoeuvre activity and a trigger condition. Figure 3.5 illustrates the logic of an SDL behaviour description which consists of three

Figure 3.5, An example illustration of the SDL dynamic description

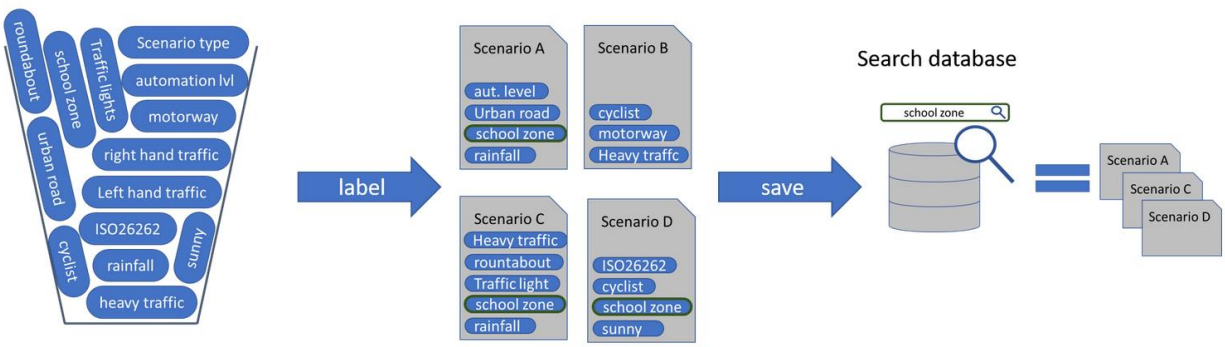


dynamic entities. The first two dynamic entities each have two activity phases and these two dynamic entities are performing actions in a synchronised fashion. The third dynamic entity starts to perform activities after the first two dynamic entities have stopped, it lasts for the remainder of the scenario until the exit conditions are met. As for the environment conditions, the SDL allows the specification of all the ODD-related environmental parameters within the description.

3.4 Scenario storage

After creating the scenarios in the WMG SDL format, together with the OpenX format, the next stage is to store them in a scenario database such that they can be used by individuals and organisations to exchange, host, query and analyse. Within the Zenzic V&V project, as well as many other national and international projects, the Safety Pool™ (*Safety Pool Database*) scenario database has been used to provide such functions. It was initiated by WMG and Deepen AI, and it utilises the ASAM OpenLABEL (ASAM OpenLABEL1.0) standard to organise and maintain scenarios within the database. The Safety Pool™ database mechanism uses a JSON schema with an associated ontology to assign ODD and behaviour tags to each individual scenario, as illustrated in Figure 3.6. In Figure 3.6 (ASAM OpenLABEL1.0), the bucket of labels corresponds to the ODD and behaviour-based ontology, the 'label' process corresponds to the assignment of these elements to individual scenarios. Once the scenarios are attached with their labels, they are then imported into the scenario database. With the help of such labels, users can navigate and filter a large number of scenarios with little effort. In addition, the Safety Pool™ database also provides the capability to host test route/track information, and to attach the associated ODD elements to the route; this allows users to effectively find the suitable test locations that match the required scenery description in the SDL. Within the Zenzic V&V project, the database API was used to retrieve scenarios in an automated way.

Figure 3.6, Illustration of a scenario tagging process using ASAM OpenLABEL



3.5 Test case generator and execution

Test case generator

Once the scenarios have been populated into the database, a scenario selector retrieves the relevant scenarios iteratively via API calls to the database; the associated SDL description as well as the OpenSCENARIO and OpenDRIVE files can be returned. Since the SDL level 2 sits at the logical scenario level where the parameters are defined in value ranges, and the

OpenSCENARIO format describes scenario parameters using concrete values, the parameters within the returned OpenSCENARIO file are set to be the mid-value within the ranges defined in the SDL.

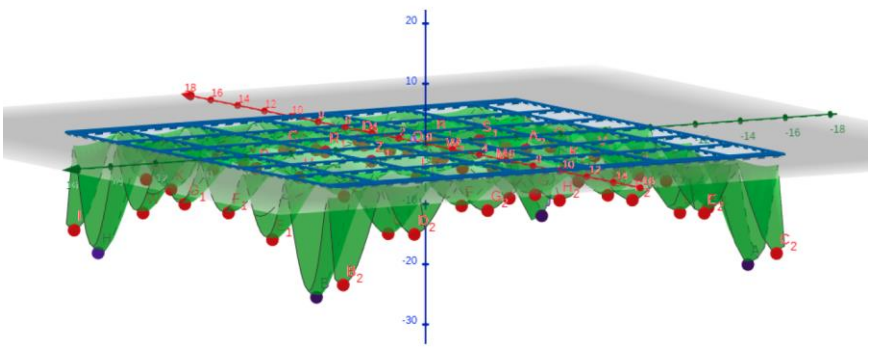
The test case generator can be divided into two different settings: the initial iteration, and subsequent iterations. During the initial iteration, if the SDL level 2 is used for the execution, a user-defined generation algorithm can be used to pick the starting values for the scenario parameters within the defined ranges. If the OpenSCENARIO file is used for the initial execution instead, the initial values are already given, which are the mid-values. For the subsequent iterations, optimisation algorithms can be implemented to intelligently select the parameter combinations for the next test case based on the analysis of the execution data.

For designing the V&V framework in general, the type of optimisation algorithm is not restricted, it serves the purpose of exploring the parameter space set out within the scenario and identifying edge-cases. Within the V&V framework used in this project, Bayesian optimisation (Gangopadhyay *et al.*, 2019) has been used to fulfil such functionality. For any given concrete scenarios, the optimiser learns parameter values by observing the scenario output; the newly identified parameter values drive the SUT to violate its pass criteria.

The high-level operations of the Bayesian optimisation algorithm can be divided into two aspects: 1) the identification of meaningful combinations of test parameters, 2) the identification of multiple sets of parameter combinations for non-convex optimisation problems. To illustrate the difference between these two aspects, Figure 3.7 (Gangopadhyay *et al.*, 2019) visualises the optimisation problem the algorithm is trying to solve. Each of the red dots represents a meaningful combination of the test parameters, described in aspect 1. However, across the parameter space, there are multiple red dots representing different clusters of interesting regions. Aspect 2's task is to identify these local regions (indicated by the blue grid at the top) to allow aspect 1 to operate and find the best combination of parameters within each region.

Hypothetically, an example of aspect 1 optimisation could be that wet road conditions cause more collisions, therefore wet road scenarios form an interesting region and within this region there is a significant combination of parameters. However, as shown in the figure above, there could be multiple feasible regions of local optima that can all lead to failure of the system. For example, there might exist another interesting region where the road is icy, for which the

Figure 3.7, Illustration of the optimisation parameter space



scenario parameters could be very different from the wet road scenario cluster. The second aspect is developed to explore all the possible sets of parameter combinations, not only to identify one set of failure conditions.

Execution

The execution can be done in different environments – virtual, real-world, or a hybrid (XiL). As illustrated in the previous section, as the environment moves from virtual to real-world, the resource demand (time, cost etc.) will increase, and the number of executions will decrease. At WMG, several simulation tools have been integrated with the V&V pipeline either in the context of industry collaborations or as internal capability development. The Zenzic Phase 3 Interoperable Simulation project has developed a distributed simulation architecture for execution, which includes the SUT and simulation software that the V&V pipeline interacts with. The various components involved across both projects are all placed at different nodes and connected via a common network. This will be elaborated on further in the next section. Further building on the simulation execution, scenarios created and stored so far can also be executed in the real world. For example, the OmniCAV project (Brackstone *et al.*, 2019) - funded by Innovate UK and the Centre for Connected and Autonomous Vehicles (CCAV) – has executed several scenarios in a real world environment. In addition, scenarios can also be executed within mixed environments such as the 3xD autonomous vehicle testing simulator at WMG, where physical vehicles are able to drive within a virtual environment, as shown in Figure 3.8.

Figure 3.8, The 3xD simulator at WMG

3.6 Test case analysis

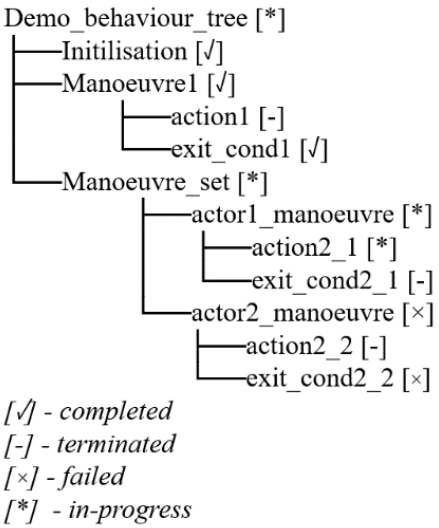
Upon the generation of the test cases and also the execution of them, the test case analysis can be performed. Both the runtime analysis and post-execution analysis can be performed using the V&V framework introduced in Figure 3.1. The analysis stage is further divided into three parts, covering different angles of verification in order to narrow down the test case selection to a useful and manageable output.

1 Has the intended test case occurred?

The first step of the test case analysis is assessing whether the intended test case has taken place. In SDL a behaviour tree can be constructed providing the means for monitoring and assessment of the test case progress. SDL and similar languages that are executable within simulators will contain a specific set of initialisation conditions, manoeuvre sequences, trigger conditions and exit conditions, all of which can be monitored during the execution of the scenario using progress monitoring. The completion of the intended test case will only occur if all of these aforementioned conditions are satisfied in the intended order. A behaviour tree allows for a visualisation of these conditions such that the executed scenario can be assessed against its intended counterpart.

Figure 3.9 (Zhang *et al.*, 2021) illustrates an example behaviour tree that consists of both parallel and sequential activities. On the main branch, it has Initialisation, Manoeuvre1 and Manoeuvre_set in a sequence. Within Manoeuvre1, it has action1 and exit_cond1 in parallel relation with a success on one criteria. This means that whichever node within the parallel relation finishes this tree branch will succeed. Within Manoeuvre_set branch, it contains the manoeuvres for both actor 1 and actor 2, along with their exit conditions. By using such a behaviour tree implementation, it provides the ability to assess which node has been completed, terminated, failed or is in-progress. In the example, Initialisation is completed and Manoeuvre1 is completed as well by the exit_cond1 being satisfied. However, within Manoeuvre_set the

Figure 3.9, Example behaviour tree progress status of a test case



actor2_manoeuvre is failed due to its exit condition failed. Such information generated by the behaviour tree implementation is used to assess whether the intended scenario has occurred.

2 Has the test case passed/failed?

If the previous assessment result is yes, the test case data will then be passed to the pass/fail assessment function. However, if the assessment result is no, the test case will be checked against a pre-defined iteration limit.

For the pass/fail assessment, several different types of criteria can be used including: STPA-related, Highway Code derived, ODD-related or a set of generic criteria. Each of the STPA scenarios has its own specifically tailored pass criteria, however such criteria need to be converted into a machine-readable format and made accessible to the ADS. In a previous publication from WMG (Chen *et al.*, 2020), the details of STPA-based scenario generation and evaluation are illustrated.

The ODD-based evaluation method uses the ODD specification of the ADS to form a safe operating boundary and evaluates the simulation ground truth data against the boundary during runtime. At any given point, the ADS could be inside or outside of its ODD, and this can then be used to assess its ability to maintain within its ODD. The ground truth data can be retrieved from the simulator during runtime; such ground truth can be then filtered to only contain the attributes listed in the ODD taxonomy, and subsequently converted into a common intermediate format. In addition, the ODD specification can be parsed and converted into the same intermediate format. An ODD assessment module then compares the two intermediate formats derived from the ground truth data and ODD specification, and outputs whether the system is inside or outside its ODD The Highway Code-converted Digital Highway Code (DHC) is developed to serve as an oracle and evaluate the ADS's ability to obey the regulations. The DHC model contains ODD elements as well as the behaviour elements, with each individual Highway Code rule being analysed and converted into a machine-readable format. An ontology framework is used to represent the domain model and the rules, and the ontology reasoner is used to perform runtime assessment of the ADS. In addition to the STPA-based, ODD-based and DHC-based test case pass/fail assessment, a set of generic assessment criteria can also be used. For example, a fixed timeout is used to terminate test cases if needed. Collision criteria are also implemented to fail test cases whenever a collision of the system is detected. Other criteria such as lane keeping, average speed limit, and maximum speed limit could be included.

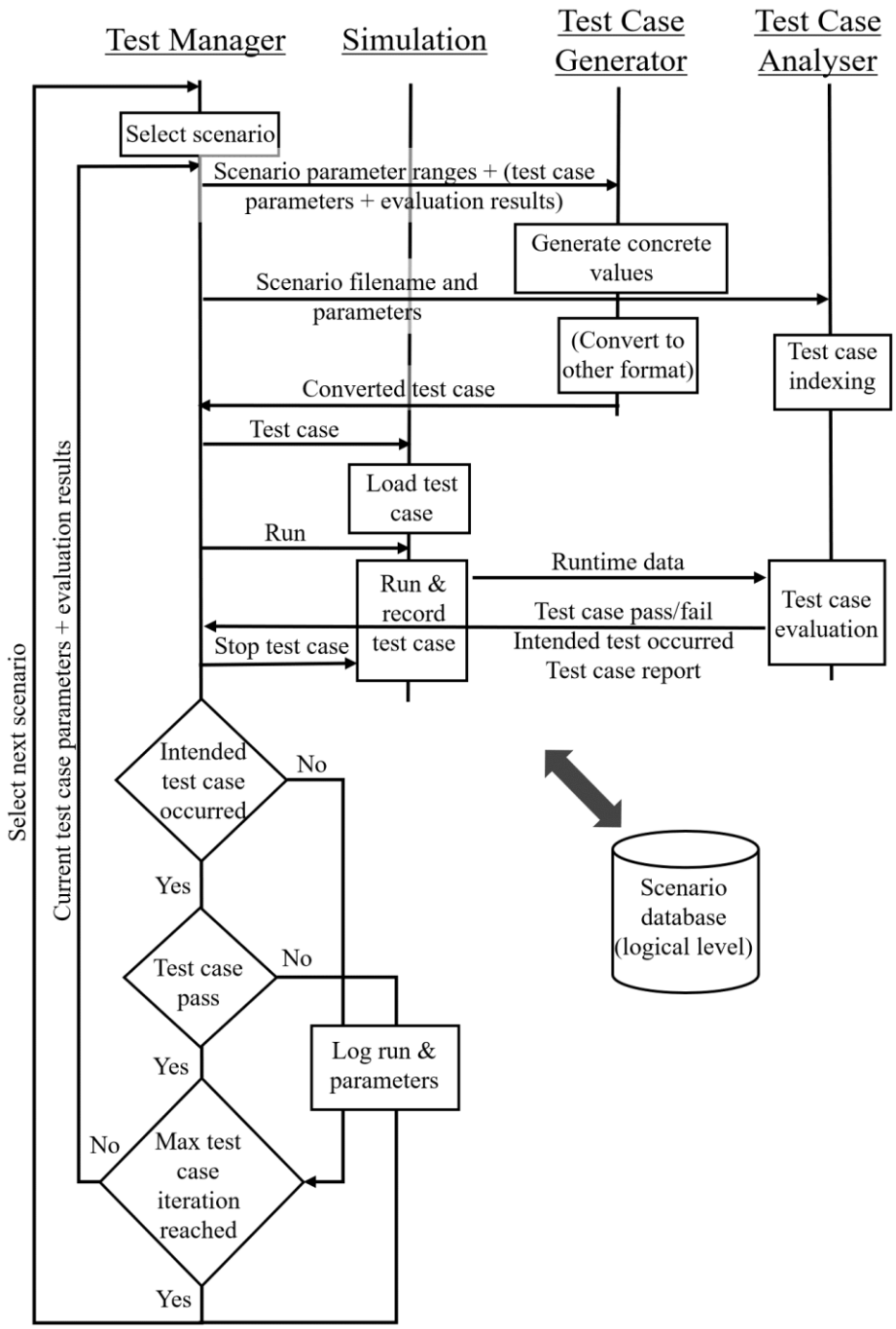
3 Has the maximum iteration of the test case been reached?

Based on the pass/fail assessment, if the result is passed, the workflow is then sent to check whether the maximum iteration for the test case has been reached. If the result is failed, then the current test case parameter combination will be logged, and the testing of the current scenario will be terminated. The maximum iteration is set to stop a large number of loops for the test case generation within the same logical scenario.

3.7 Scenario-based evaluation at implementation level

This section illustrates the implementation architecture of the V&V workflow. As shown in Figure 3.10 (Zhang *et al.*, 2021), the necessary functions are modularised into four parts: Test Manager, Simulation, Test Case Generator, and Test Case Analyser. The Test Manager is in charge of orchestrating the whole workflow. All the communications are established between other modules and the Test Manager. Simulation contains the environment simulator, the traffic simulator and the SUT. The Test Case Generator is for generating concrete test case parameters and optionally converts the scenario into other formats (rather than the WMG SDL, OpenX) prior to execution. The Test Case Analyser contains test case indexing in order to obtain the scenario parameter variables, and test case evaluation for checking the three decision boxes mentioned in the previous section.

Figure 3.10, Implementation architecture of the V&V workflow



The whole workflow is initiated by the Test Manager which is integrated with the scenario database for selecting the logical scenario and passing the scenario information to the Test Case Generator. A concrete test case is then obtained either by using the average values of the parameter value ranges in SDL, or using the already defined values in the OpenSCENARIO 1.x file. Optionally, conversion into other desired scenario formats can be performed before sending the prepared test case back to the Test Manager. Meanwhile, the Test Manager also sends the scenario information to the Test Case Analyser for indexing. Upon receiving the generated test case, Test Manager sends the converted test case to Simulation, and it then sends a run signal to start the simulation. During runtime, live data is communicated between the Simulation and

Test case evaluation module. The Test case evaluation will then produce the evaluation results and send back to the Test Manager. The Test Manager then stops the simulation run, and checks the evaluation results against the three decision boxes. Based on the outcome, the Test Manager will:

1. Select a new logical scenario to test,
2. Command to generate new test cases under the same logical scenario.

Such process will continue until all the intended logical scenarios have been tested. Please note that this implementation architecture is the workflow currently used within WMG; for the Zenzic V&V project, it has been modified slightly to integrate with the simulation execution framework developed as part of the Zenzic Phase 3 Interoperable Simulation project.

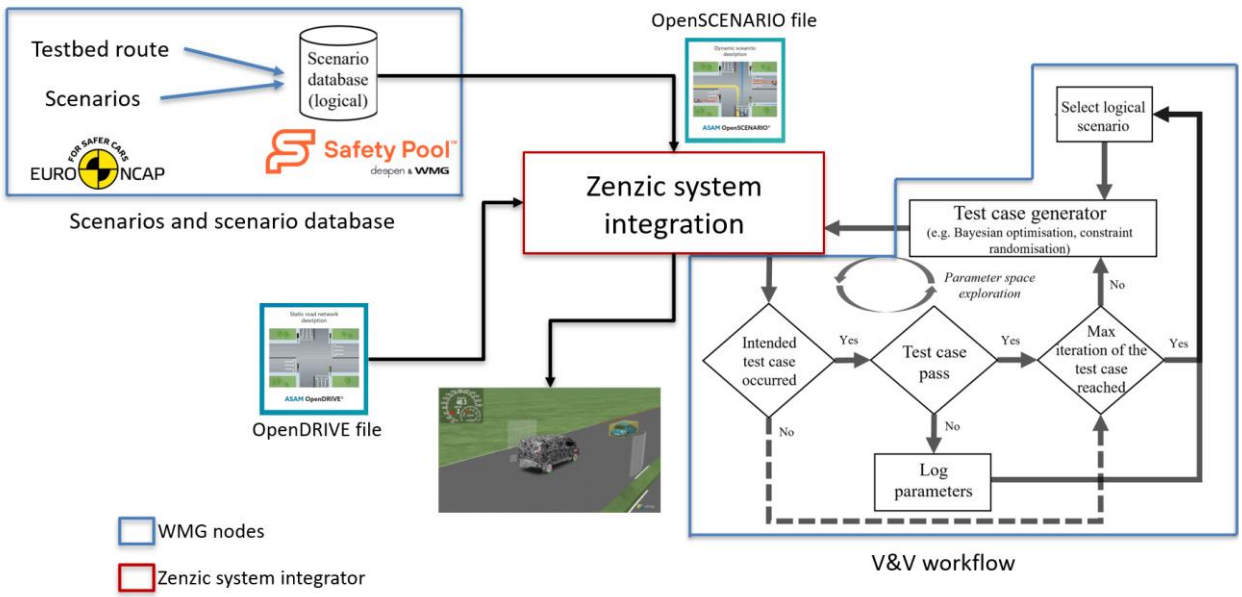
4 | Framework integration

4.1 Integration with the Zenzic Interoperable Simulation ecosystem

To integrate the V&V implementation framework with the Zenzic Interoperable Simulation ecosystem, the elements of the functional workflow diagram (Figure 3.1) were divided into three individual blocks. As shown in Figure 4.1, the Zenzic system integration layer sits at the centre of the Zenzic V&V integration work. The integration utilises a common network to enable communication between different operational nodes that are connected to this network. At the top left, it connects to the Safety Pool™ scenario database via Safety Pool's API. In addition, it also connects to the SUT node, and the simulation execution node. The majority part of the WMG V&V node is seen at the bottom right, which sends the scenario ID to the Zenzic system integration layer and receives execution output from it.

In this case, based on the original WMG V&V framework, the following modifications were made: 1) the database API call was made by the Zenzic system integration layer rather than the WMG V&V module, 2) the simulation execution and the SUT were not hosted within the V&V framework, but were connected as a separate node to the Zenzic system integration layer. The rest of the V&V framework largely remained the same. The following sub-sections illustrate the scenario preparation and retrieval process, the construction of the data transmission schema between the Zenzic system integration layer and the V&V module, and the use cases performed during the Zenzic V&V testing process.

Figure 4.1, Zenzic V&V integration functional diagram



4.2 Scenario preparation and retrieval

The preparation of scenarios used in the Zenzic V&V project contains several stages: 1) construction of the SDL scenarios, 2) conversion from SDL to OpenSCENARIO/OpenDRIVE, 3) custom modifications to the OpenSCENARIO file to fit the UTAC testbed OpenDRIVE specification, 4) confirmation of suitable parameters for updating with the Zenzic system integration layer and the execution node, 5) upload onto Safety Pool™ database and generation of the corresponding scenario ID, 6) enabling API calls from the Zenzic system integration layer to the database for scenario retrieval.

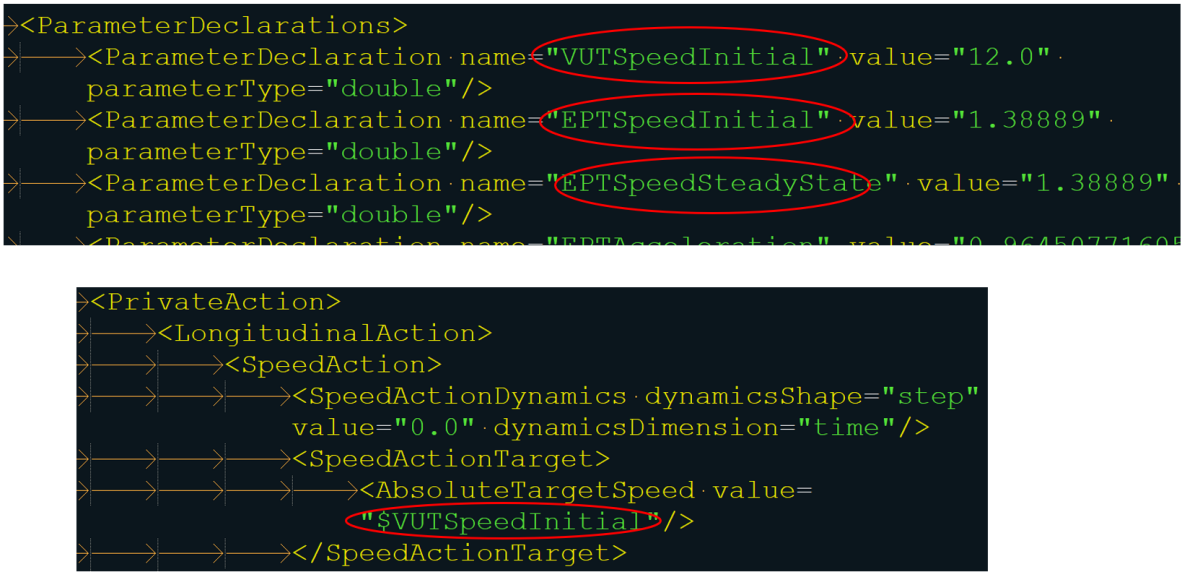
Since the scenarios used in this V&V project were agreed to be Euro NCAP (Euro NCAP, 2017) scenarios (which will be described in detail in Section 4.4), the creation of SDL scenarios involved manual translation, based on the tables and diagrams found in the relevant Euro NCAP documentations (NCAP, 2020)(EuroNCAP, 2017), to SDL level 1 and level 2.

Upon the generation, a toolchain is available to convert from SDL level 2 description into the equivalent OpenSCENARIO 1.1 and OpenDRIVE 1.6 format. The building blocks of this conversion toolchain include an SDL parser based on the pre-defined grammar of the SDL level 2. The parsed information is then sent to a generator based on the OpenSCENARIO/OpenDRIVE schema, and an established mapping between the SDL elements and OpenX elements; the generator then outputs the corresponding OpenSCENARIO and OpenDRIVE files. Please note that since the scenery description within the SDL is location-independent, and its features are based on the criteria from the relevant documentation, the generated OpenDRIVE file is also location-independent and is generic. The term 'generic' indicates that the toolchain will constantly generate the corresponding OpenDRIVE file purely based on the SDL description of the scenery; the road and lane IDs used within the file are generated by a default setting within the toolchain.

For the Zenzic V&V project, the virtual 'Mile Straight' of the UTAC testbed was used. To adapt the converted OpenSCENARIO file to the UTAC-based OpenDRIVE file, a manual modification was made on each of the OpenSCENARIO files to select the suitable road and lane IDs for the entity actions. All the Euro NCAP scenarios take place on a straight single road, therefore the 'Mile Straight' within the UTAC testbed was chosen as the scenario initialisation and testing location.

As described in Section 3.5, during the V&V process, the scenario parameters need to be constantly updated, while the scenario storyline is kept the same. The SDL level 2 description allows for easy modification of the scenario parameters, with the user able to simply replace the range values with different concrete values during each iteration. However, in the Zenzic V&V process, the executed files were chosen to be OpenSCENARIO and OpenDRIVE; to easily update the corresponding scenario parameters, they are written inside the 'parameter declaration' section of the OpenSCENARIO file. As seen in Figure 4.2, the parameter declaration section, all the parameters that need to be updated are gathered in here, together with their default value and parameter type. Then within the scenario storyboard, only the variable names are referenced. During the Zenzic V&V process, only the values within the parameter declaration section were updated while the rest of the file remained the same.

Figure 4.2, Illustration of parameter declaration within the OpenSCENARIO file



The last step for the scenario preparation includes the upload of the scenario to the Safety Pool™ database and gathering of the corresponding reference ID, this ID is then used during the actual V&V activities to be sent to the system integration layer for each logical scenario. The system integration layer will then query the OpenSCENARIO file from the Safety Pool™ database using such scenario ID.

4.3 Simulation data and test case parameters

During the integration between the V&V framework and the Zenzic system integration layer, the data format used for communication was Google Protocol Buffers (Protobuf in short) (*Protocol Buffers*). It is a language-neutral, platform-neutral, extensible mechanism for serialising structured data; it is smaller, faster, and simpler than the xml format. In order to use the Protobuf format, the process is divided into two steps: 1) the definition of the message template, including all the possible scenario parameters together with their data type, 2) sending the actual message during the V&V activities.

Two different message elements have been defined for the Zenzic V&V integration: the message from the V&V module to the Zenzic system integration layer, and the message from the system integration layer back to the V&V module.

For the message sent from the V&V module to the integration layer, the required message contents are simMode, scenarioID, testRunIteration, and updated parameters. simMode contains three enumerators, which are 'first run', 'continue', and 'stop'. 'first run' does not require inputs of the scenario parameters as they are based on the default values in the OpenSCENARIO file, 'continue' indicates these iterations will contain corresponding parameter updates, and 'stop' is used after the last iteration to terminate the whole process. The scenarioID is a unique identifier the database assigned to each scenario, the testRunIteration is used to indicate the current iteration number, and the updated parameters contain the parameter names as found in the

parameter declaration section of the OpenSCENARIO file, together with the updated parameter values. Please note that the updated parameter section can contain any number of updates between one and the number of parameters in the parameter declaration section.

For the message sent from the integration layer back to the V&V module, the testGroundTruthRecord of each execution is recorded and sent to the V&V module for post-run analysis. The information included within this ground truth record is scenarioID, testRunIteration, and timeStampedGroundTruth. The time stamped ground truth data further contains ego ground truth, environment ground truth, scripted vehicle ground truth, and VRU (vulnerable road user) ground truth data. In here the ground truth data is obtained directly from the simulation node. The ego ground truth contains the ego vehicle's road lane position, lane type, junction type, left/right lane marking types, collision information, position in the world coordinates, velocity, orientation, acceleration, gear, throttle, brake, steering. The environment ground truth contains cloudiness, precipitation intensity, wind intensity, light intensity, fog level, and the road wetness. The scripted vehicle ground truth contains the agent's name, road and lane position, velocity, acceleration, and the world coordinates. The VRU ground truth contains the VRU agent name, the road and lane position, and the world coordinates.

4.4 Euro NCAP AEB use case

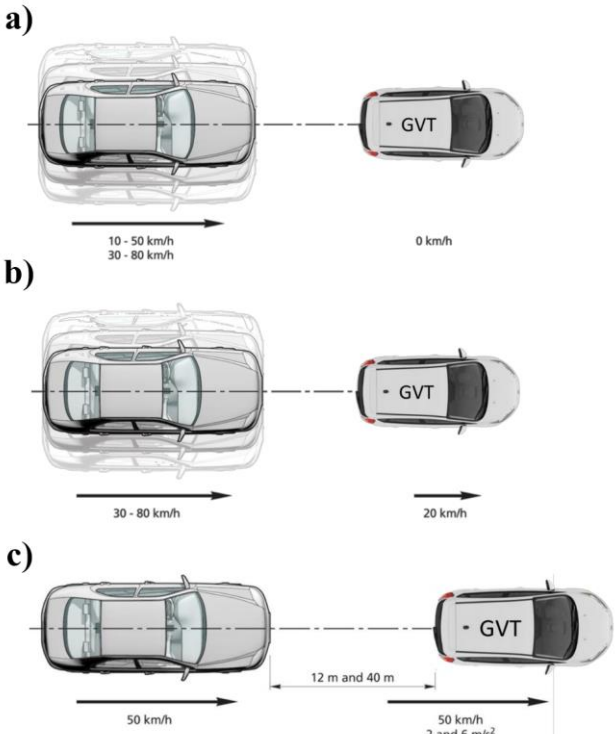
The SUT tested within the Zenzic V&V project was an autonomous emergency braking (AEB) system. Across different partners involved in the Zenzic V&V work, the Euro NCAP scenarios were chosen. They include Car-to-Car scenarios, as well as AEB VRU scenarios. Please note that although the Euro NCAP use case was chosen, the V&V framework is independent to any specific use case. To adapt the framework to different use cases, one would need to use the corresponding scenario creation methods to obtain the related scenarios, and input the corresponding assessment criteria.

Taking the traffic rule compliance use case as an example, at the scenario creation stage specific test scenarios need to be generated which correspond to the individual rule or a combination of rules. Each rule specifies what the driver *can* or *cannot* do when encountering a situation; from this the assessment criteria can be obtained. For instance, one of the overtaking related rules (Rule 162) from the UK Highway Code specifies that during overtaking, the overtaking vehicle needs to ensure that there is a 'suitable gap' to the vehicle being overtaken. Such 'suitable gap' can become part of the metrics for the assessment of the scenario.

Euro NCAP Car-to-Car scenarios

Figure 4.3 (EuroNCAP, 2017) displays the three Car-to-Car Euro NCAP scenarios tested during the V&V activities; they are Car-to-Car Rear Stationary (CCRs in Figure 4.3a), Car-to-Car Rear Moving (CCRm, in Figure 4.3b), and Car-to-Car Rear Braking (CCRb, in Figure 4.3c). In all three scenarios, the Global Vehicle Target (GVT) as indicated in Figure 4.3 is the scripted vehicle, and the SUT is the rear following vehicle. In CCRs, the GVT is stationary and the SUT is moving

Figure 4.3, Euro NCAP car-to-car scenarios

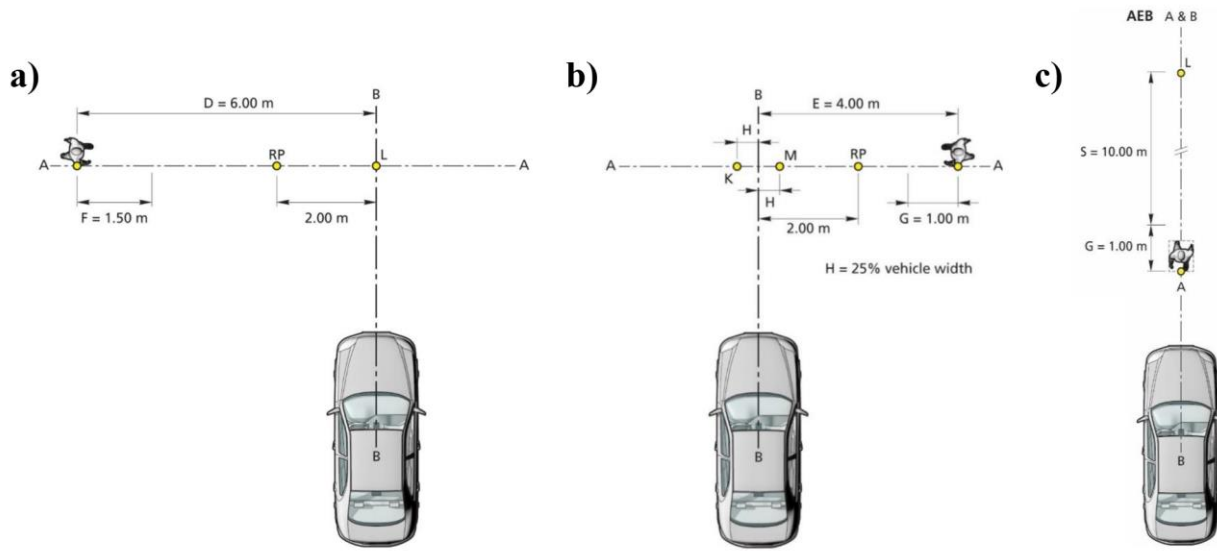


towards it at two sets of speed ranges. In the CCRm, the GVT is moving at a slow speed while the SUT is moving at a faster speed following the GVT. In CCRb, the GVT is performing a braking action while the SUT follows behind.

AEB VRU scenarios

Figure 4.4 (NCAP, 2020) displays the three Euro NCAP AEB VRU scenarios used for the testing, they are Car-to-Pedestrian Farside Adult (CPFA, in Figure 4.4a), Car-to-Pedestrian Nearside

Figure 4.4, Euro NCAP AEB VRU scenarios



Adult (CPNA, in Figure 4.4b), and Car-to-Pedestrian Longitudinal Adult (CPLA, in Figure 4.4c). In CPFA, an adult pedestrian is walking towards the trajectory of the SUT from the farside. In CPNA, an adult pedestrian is moving toward the SUT from the nearside. And in the CPLA, an adult pedestrian is walking toward the SUT along the SUT's trajectory from the front.

During both the car-to-car scenarios and the VRU scenarios, the V&V framework effectively converged towards a parameter combination which encourages hazardous situations (i.e., collision in this case). For example, during the initial iteration of the CPFA scenario, the pedestrian would reach the opposite side of the road even before the SUT reached the pedestrian's trajectory. Since the role of the optimisation module is to identify hazardous situations, within the subsequent iterations, the SUT initial speed, the pedestrian target speed, and the initial distance between the vehicle and the pedestrian were updated by the algorithm. It can be seen in Figure 4.6, the initial SUT speed was constantly updated to higher values to encourage early intersection between SUT and pedestrian. In the meantime, the pedestrian target speed was updated to lower values, and the initial distance between the pedestrian and the SUT was also decreased, thus encouraging a potential collision to occur. Similar performance of the optimisation algorithm was observed across all the test scenarios. Figure 4.5 illustrates the same parameter updates but using test case layouts; the arrows indicate the speed and the length of them indicate the magnitude. Please note that these three parameters are not the only parameters that were updated during the testing, but they are the most visually representative.

Figure 4.6 Examples of parameter optimisation

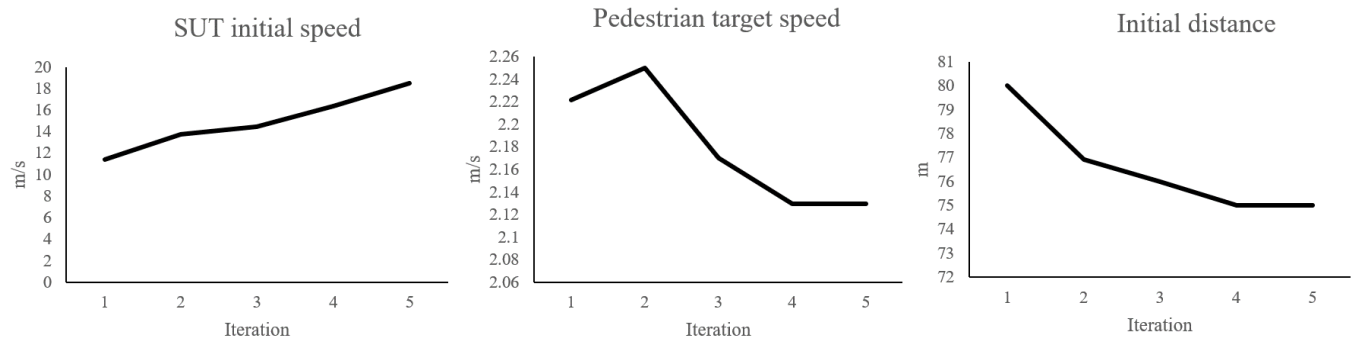
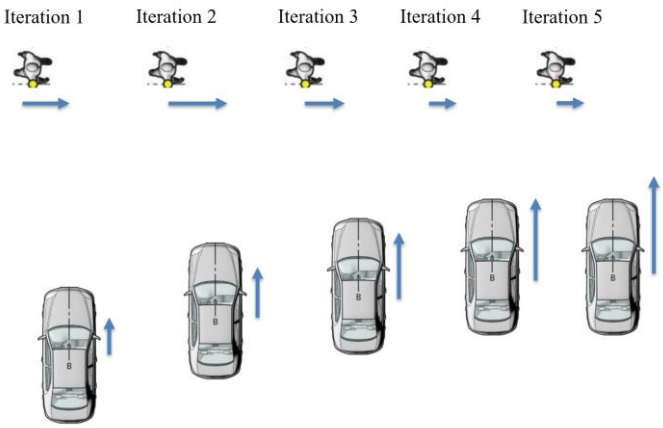


Figure 4.5 Visualisations of the test case parameter update

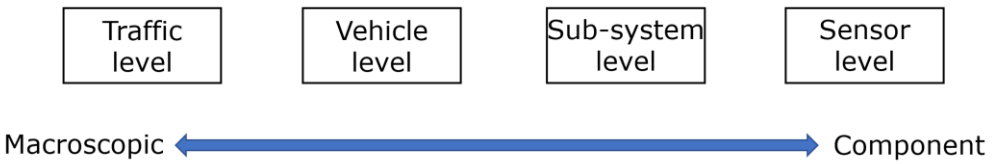


5 | Simulation validation methodology

5.1 Dynamic elements validation method

When referring to the dynamic elements, there are four levels as shown in Figure 5.1.

Figure 5.1, Different levels of dynamic elements



At the most left side, the dynamic elements can be represented from the macroscopic traffic level. At this traffic level, an un-accountable number of dynamic agents travel between a starting point and a destination point through intelligently calculated routes. The behaviours of the dynamic agents within the traffic are not explicitly controlled, but rather left for the traffic model to control, which creates the 'smart' actors. Traffic properties are part of the ODD for an ADS (Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification, 2020), it is therefore important to validate their performance against real world traffic. An example use case could be – if the ODD of an ADS can handle highly congested motorway conditions, such an ODD claim will need to be validated by setting the traffic behaviour to match the characteristics of highly congested traffic (i.e., high density, low speed, large variety of vehicle types, etc).

The second to the left is the vehicle level dynamic behaviour, this level targets at individual vehicles rather than the macroscopic traffic. Vehicle level behaviours are commonly described in scenario description languages such as the WMG SDL (Zhang, Khastgir and Jennings, 2020b), and ASAM OpenSCENARIO (ASAM OpenSCENARIO 1.1.1). At this level, the trajectories, speed, acceleration, and their temporal developments for a target vehicle will be explicitly stated. Please note that it is not of concern as to how the vehicle's internal systems achieve such behaviour at this level.

The next level down represents the system level dynamic behaviour, in which the high-level system components of an ADS are represented. Mapped to the WMG SDL concept, usages of this level for validation are normally associated with the SUT's internal behaviours. For example, STPA-based scenarios (Chen *et al.*, 2020) have the specific internal element section with their SDL description to describe the SUT's internal behaviours, as well as what a test engineer should perform on the SUT's internal system for each scenario (e.g., manually delay certain signal from sending/receiving, therefore test the SUT's backup system). The system level architecture of an ADS can be broadly divided into four parts: sensing, perception, planning and control. The sensing aspect will be explained later in this section at the sensor level dynamic validation. Sensing converts the outside environment into machine readable format (i.e., a camera is converting the shape and colour of objects into RGB values which the computer can interpret and render). Perception will further take such converted information to process and generate

ground truth (i.e., a human detection algorithm will detect the presence of a human based on a list of RGB values). The planning algorithm will then process all the ground truth and generate strategies for navigations, and the control part will take the actions to implement the strategy. The validation at the system level will be discussed in more detail later in this section.

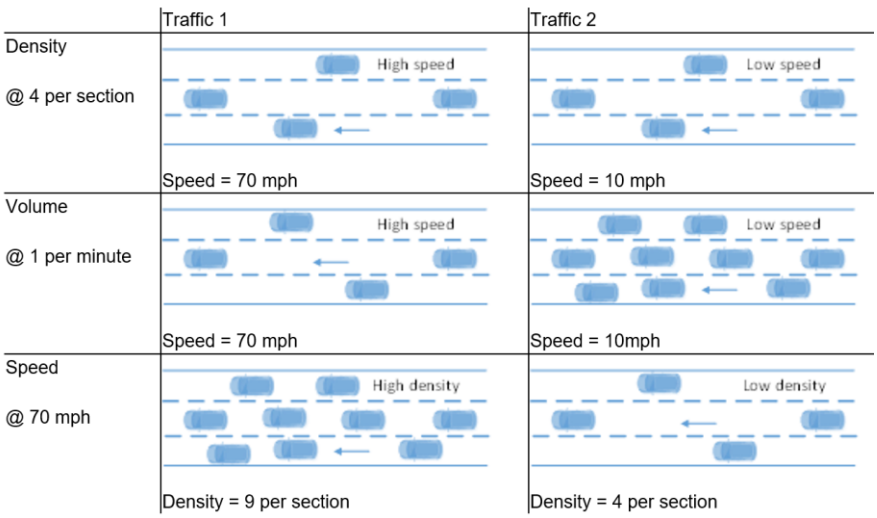
The last section of the validation is at the sensor level. Sensors are normally included in the sensing part of an ADS system, which converts the environmental features into a machine-readable format. It is crucial to have capable sensors within an ADS as their output can directly impact all the subsequent processes. Within the industry, the development of sensor technologies is usually handled by individual sensor suppliers rather than the system or vehicle developer. However, ensuring adequate and compatible sensors or sensor models within the system is important. Their validation approach will be covered later in this section.

Approaches for different levels of validation

Macroscopic traffic level

To describe the macroscopic traffic properties at a high level, there are three basic parameters that define traffic, which are **density**, **volume** and **speed**. In addition, the traffic agent composition (e.g., 10% truck, 5% motorcyclists) is also a key factor. Traffic elements form part of the dynamic elements of an ODD ('Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification', 2020) and therefore are crucial parts of the ADS safety assurance. **Density** defines how many vehicles on average are expected within the observed road section, measured as vehicles per distance. **Volume** defines the rate of vehicle appearance/disappearance at the observed road section, measured as vehicles per duration of time. And **speed** defines how fast the vehicles will travel, measured as distance per time. One can derive the third parameter by knowing the other two parameters, **since volume = density × speed**. However, if only one parameter is known, the traffic will not be uniquely instantiated. Figure 5.2 illustrates cases where different traffic shares the same value when only one parameter is specified. For the first case, only the density is specified which is at four vehicles per section; both traffic 1 and traffic 2 can satisfy such a requirement, and the traffic speed

Figure 5.2, Illustration of traffic situations when only one parameter is specified



cannot be fixed to instantiate a unique traffic model. Similarly, case two only specifies the traffic volume which is at one per minute; this also cannot instantiate an unique traffic model. At case three, although the traffic speed is specified, this allows variations in the traffic density, and consequently the traffic model is not uniquely instantiated. Therefore, at the traffic level, by utilising these three traffic characteristics one could establish the baseline for a traffic level validation process. Furthermore, if a traffic model spans across a large area, sub-divisions of the traffic can be validated individually.

Vehicle level

Vehicle level comparisons can be divided into ego behaviours and non-ego behaviours. The impacting factors on ego behaviours at the vehicle level can propagate down into the system and sensor levels of simulation. However, to examine the ego behaviour, vehicle level comparison against the real world recorded data can still be performed to provide an overall benchmark for a vehicle level validation. The non-ego behaviours, on the other hand, are key for the correct representation and execution of the intended scenarios. Within a concrete test case, the non-ego behaviours are desired to be deterministic such that the same test case can be reproducible. In SDL level 2 (at the logical scenario level), the non-ego vehicle behaviour is represented using manoeuvres, speed, acceleration, relative positions etc. In lower level languages such as OpenSCENARIO 1.x ('ASAM OpenSCENARIO 1.1.1'), individual trajectories of the non-ego actors can also be defined. Therefore, the accuracy of simulation execution can be assessed using both the trajectory information and the higher-level manoeuvres. As shown in Figure 5.3, at the most detailed level, by comparing the deviation between real world non-ego trajectories with the simulation result, the assessment can be performed. Moving one level up, the trajectory can be represented using lateral and longitudinal behaviours. The next level up combines the lateral and longitudinal actions into combined manoeuvres, for example 'turn left' or 'change lane right' are both results of lateral and longitudinal actions. At the top level, behaviour can be expressed in turns of missions, for example 'follow lead vehicle' or 'keep safe distance'; they cannot be simply expressed as particular manoeuvres, and they can be complicated as well as simple dependant on the lead vehicle behaviour in this case.

Based on the trajectory comparison option, a recent study (Tenbrock *et al.*, 2021) compared the behavioural information between physical data and virtual result. As shown in Figure 5.4 (Tenbrock *et al.*, 2021), the individual vehicle trajectory data are extracted from various traffic, and converted into OpenSCENARIO 1.1 format, which was then executed in the esmini and Carla simulators. By performing the comparison, the deviation between the trajectories exhibited by

Figure 5.3, Different abstraction levels of vehicle level behaviour

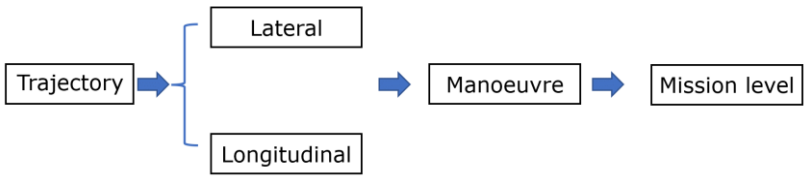
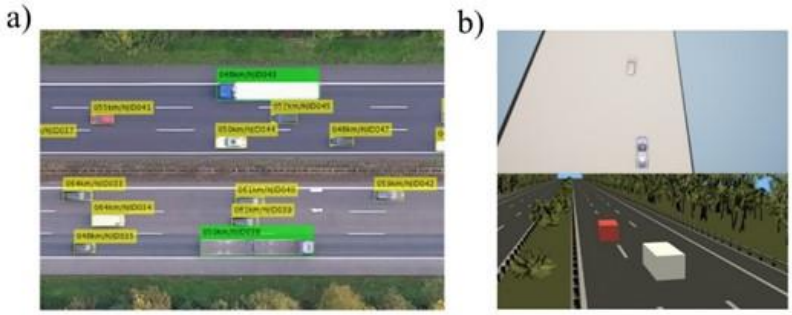


Figure 5.4, a) information extraction from physical data, b) re-creation of the scenarios in a virtual environment



the scripted agent using OpenSCENARIO and the raw data can be quantified. Furthermore, other vehicle level data such as velocity and acceleration can also be compared between the virtual and real-world data. As shown in Figure 5.5 (Tenbrock *et al.*, 2021), the velocity data is plotted over time between the real-world data input and the behaviour extracted from the esmini simulator.

System level

An ADS contains perception, planning and control as the high-level core system components; please note here the sensing part is treated as the step before reaching the ADS core functions so that the sensor model can be isolated. As shown in Figure 5.6 (Tas *et al.*, 2016), the perception, planning (cognitive decision) and control (action) form the cognitive system. The interface between the cognitive system and the outer environment is achieved through sensing. Each part within the ADS functions can be further broken down into sub-system level components; Figure 5.7 illustrates an example information flow between different modules (Tas *et al.*, 2016). The sensing layer includes all the sensors which includes (but are not limited to) GPS, IMU, vehicle network, camera, radar, lidar. The sensor layer captures the outside environment and converts it into a machine-readable format for the perception layer. The perception layer is in charge of recognising the captured environment information and identifying the vehicle state as well as understanding the scene around the vehicle. This recognition and understanding can be established from RGB format data (from the camera) to recognise objects (e.g., pedestrians) within the RGB value arrays. Since the input to the perception layer contains multiple sensor outputs, the final stage of the perception layer will be to fuse the sensor

Figure 5.5, Comparison of vehicle velocity between simulation and real-world

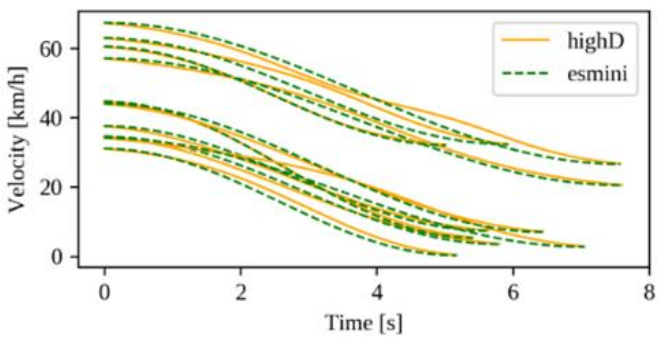
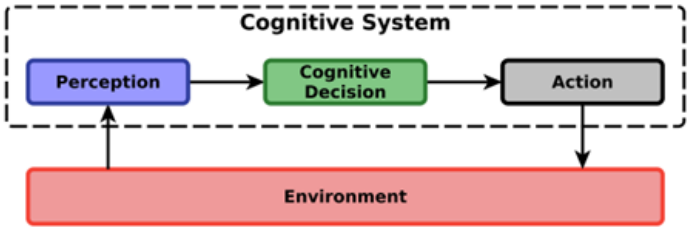


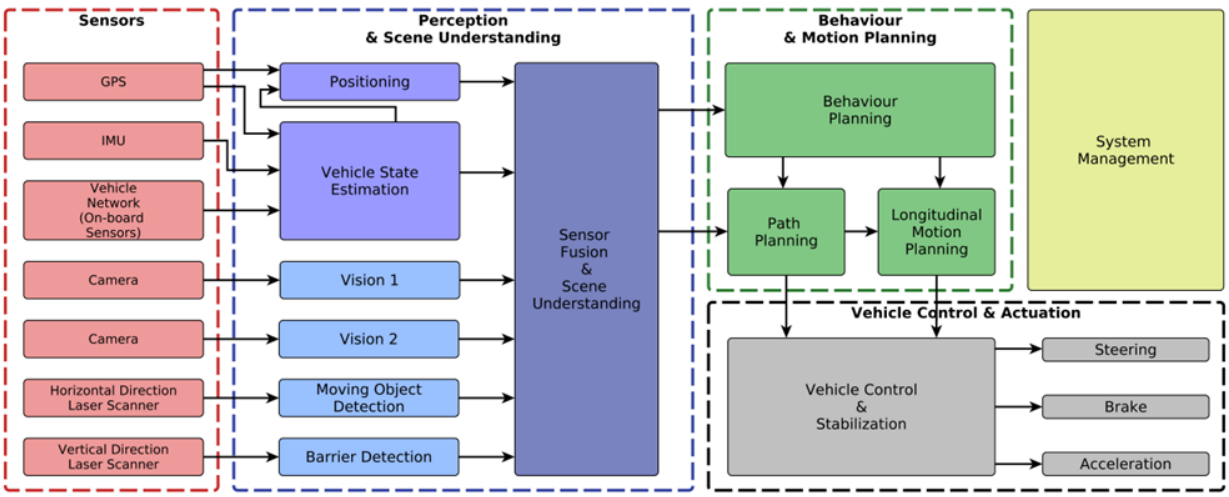
Figure 5.6, A cognitive system interacting with its environment



information together to form a unified understanding of the surrounding. The fused information will then be passed onto the behaviour and motion planning, and vehicle control and actuation layers. Planning will calculate the most suitable path to achieving the objective of the navigation, however it will not control the system to move at this stage. The control and actuation module will then take the path information to work out the actuation values (such as steering, brake, throttle) for the system to smoothly move towards the target.

At the system level, the evaluation can be made directly on the output of the sensor model; identical sensor data shall be used as input to both the physical and digital twin versions of the ADS. By comparing their outcome, the representativeness between the virtual and physical models can be established. To examine the system level similarities, there are two approaches: 1) directly compare the system level output, 2) compare the vehicle level behaviours since it is a function of system level behaviours. In approach 1, the internal data at any stage can be compared directly between the two environments. For example at the vehicle level, the throttle or steering values can be compared. At the internal level, the outcome from various sub-systems can also be compared, such as the calculated path from the path planning module. For approach 2, the behaviour at the vehicle level will be used for the validation. As mentioned earlier, the vehicle level behaviour is a function of the system and sub-system level behaviours, since they propagate upwards and affect the vehicle behaviour. Therefore, comparing how the vehicle behaves between the simulation and physical worlds will indicate the system level validity.

Figure 5.7, An example information flow diagram between different modules



Sensor level

Sensor level validation focuses on the sensing module; referenced to Figure 5.7 it will be comparing between the environment input and the sensor module output across different environments. Since here the key is to identify the delta between virtual and real-world sensor models, the environment information is not a variable in here. Furthermore, sensor distortion or degradation modifications can also be applied to the virtual sensor model, and assessed together in an integrated unit against the real world sensor with degradation or distortion.

5.2 Static elements validation method

Background

The static element validation method used in this project focuses on evaluating the virtual representation of the 3D objects as compared to the real-world. The method proposes to use a lidar system to devise similarity metrics between the real-world based 3D objects and a virtual digital twin. Although a lidar system is used to illustrate the methodology, such comparison can be generalised to all 3D objects to allow validation using other sensors such as cameras or radar. The overall workflow includes three stages: (1) collect the real-world data, (2) synthesize the virtual map and virtual testing environment, (3) perform the same data collection within the virtual environment, (4) develop a method for comparing the virtual world lidar scans and real-world lidar scans.

During the development phase of this method, that uses 3D object comparison to validate the virtual environment, four main criteria have been established:

1 **Suitability for the data type**

A given static element object within the environment is created from a lidar environment scan. This assumes that it is large in scale and complex, unlikely to be convex and will be unable to be described through a single shape. Which is generally different to the type of model created through using CAD systems.

2 **Automation**

The method should be able to function without manual interference or prior knowledge of the model. This will allow for the comparison to be done on many pairs quickly and will allow validation to completed without human bias.

3 **Shape preserving**

Ideally the method should maintain the shapes' original structure without deformation. This is to avoid loss of information and fidelity for the comparison.

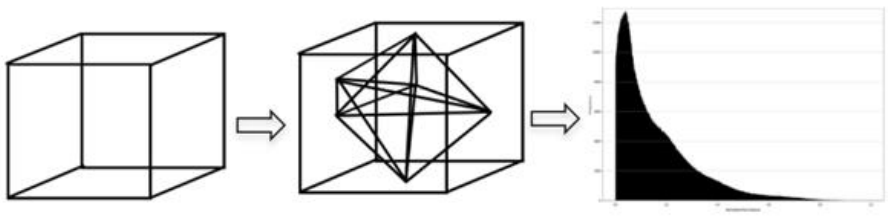
4 **Practicality**

This mainly includes complexity, runtime, and robustness, meaning it will be able to handle large and complex objects quickly and without breaking even if the data is not exactly as expected.

To illustrate the methodology proposed in the Zenzic V&V project, it is important to review the current approaches documented in the literature for comparing 3D objects. The most direct comparison method for lidar and point cloud data is to make a one-to-one comparison using the mean distance between corresponding points as demonstrated in (Zhou *et al.*, 2020). The comparison based on this method is either done on a point-by-point basis, or by using a method that takes a least squares average plane (of all the points within a region) around a target point and finds the shortest distance between this plane and the original point. The biggest disadvantage of this method is it requires manual rotational alignment of the two clouds to allow corresponding points to be compared. Other methods (Huang and You, 2012)(Novotni and Klein, 2001) tried to tackle the alignment issue and allowed for automatic alignment between the objects by identifying key points as references, however they do not account for any discrepancies in the point cloud data, as a small discrepancy in the alignment will cause a large effect in the comparison score. Similarly, another method used an underlying surface created using a 3D Fisher vector known as DPDist (Urbach, Ben-Shabat and Lindenbaum, 2020), but such approach uses neural networks which requires many sample models to train. The method in (Gotoda, 2003) suggests turning a 3D object into many 2D silhouettes from different perspectives; the most similar silhouette pairs are identified between the two objects and the dissimilarity is calculated and summed over all pairs. This method would work best with convex objects that have distinct silhouettes, whilst the point cloud maps for autonomous driving usage will have lots of detail that will be obscured by the silhouette process (obstructed by buildings etc). The method described in (Muliukha, Lukashin and Ilyashenko, 2019) uses a comparison of the distribution of pairwise distances between a sample of points on the surface of the object. The distribution for each shape is found by taking a sample of points on the surface of the object, finding the pairwise distances between them and storing those in a histogram as shown in Figure 5.8. The method is also indifferent to the alignment of the object as the distribution is inherent to the shape and it is shape preserving. The method runtime can be controlled by the number of points sampled and is unaffected by the size of the object.

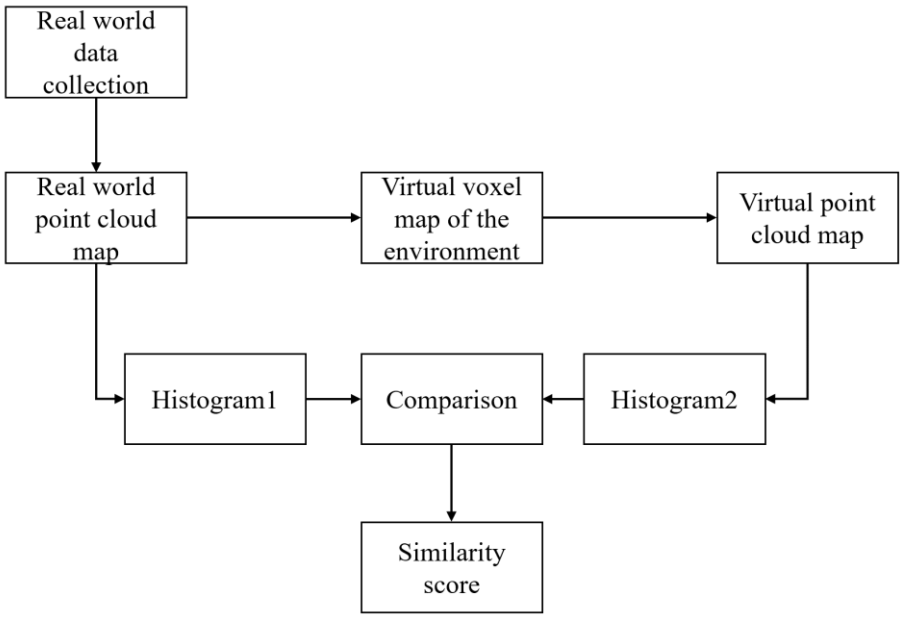
Proposed methodology

Figure 5.8, Demonstration of cube to histogram conversion



The high-level workflow for the static element validation is summarised in Figure 5.9. As can be seen the process starts with a data collection in the real world using a lidar sensor; a point cloud map can then be generated based on the mapping data. A virtual voxel map can be created based on the point cloud data, which will enable a virtual vehicle to perform a virtual scan within

Figure 5.9, Static elements validation workflow



this voxel map, creating a virtual point cloud map. A comparison of the point cloud characteristics can then be performed to produce a similarity score between the virtual point cloud map and the real-world point cloud map. The difference between the two can be used to assess the fidelity of the simulation environment. Based on this high-level workflow, the sub-sections below will describe each step in more detail.

The real-world data collection can be achieved by using a 360-degree lidar mounted on a vehicle, while the vehicle travels along an area of interest to collect point cloud data. The point cloud data is then transformed into a voxel environment map. The voxelisation can be performed using a 3D cubic grid of specified dimensions; if a point cloud data is present inside a cubic grid then it is filled in as a voxel, which is a solid cube that can be thought of as a three-dimensional pixel. Then a simulated lidar can be placed on a virtual vehicle which travels the same route through the voxel environment as the vehicle did through the real world, thus producing a simulated lidar scan. These two sets of scan data will be the input to the data analysis box.

Adapted from the method documented in (Muliukha, Lukashin and Ilyashenko, 2019), the proposed comparison method works by taking a sample of points on the surface of each object. The pairwise Euclidean distances for all these points are then calculated and stored in a 2-dimensional matrix. These distances are normalised with respect to the maximum distance of each object and converted into a histogram with a fixed number of evenly spaced bins. Instead of storing frequency in the histogram, probability density is used instead. Once the two histograms are created, these are discrete probability distributions that represent the shapes. These distributions can be compared using the sum of the absolute differences between the columns of the histograms.

The main impacting factor of this method is the element of inconsistency created through the random sampling. This can be mitigated by taking a large enough sample of points which will

give a more stable distribution of distances for the object. Another key factor that affects the consistency of the method is the number of bins created for the histograms, as having a greater number of smaller bins will allow for a higher fidelity comparison as smaller differences in distance will be accounted for, this however does also lead to a greater effect on the score from the random sampling. Since the algorithm has $O(n^2)O(n^2)$ complexity with respect to the number of sample points, it is important to find the highest number of sampled points that still has an acceptable runtime.

5.3 TRL level assessment for the simulation

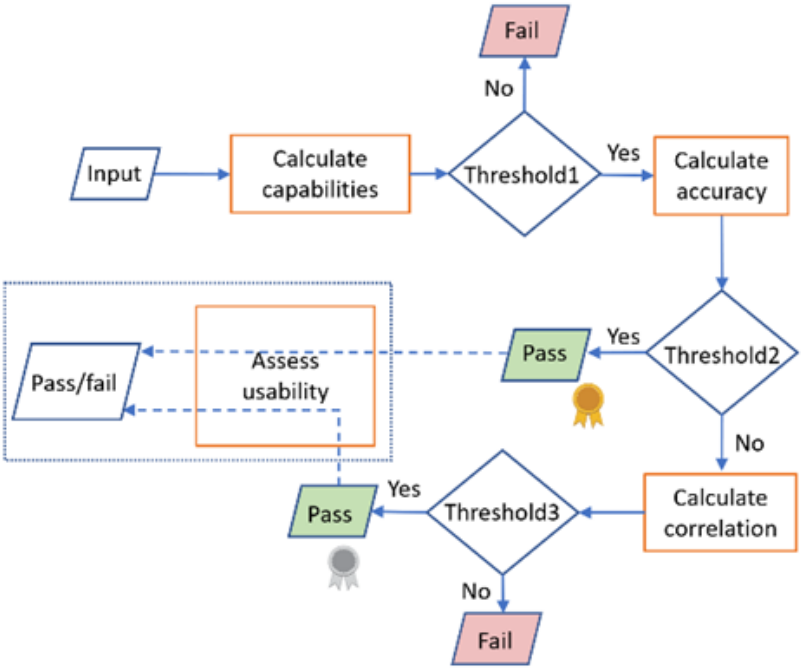
This report so far has introduced both the dynamic and static elements within a simulation environment, together with their validation approach. This section will propose a Technology Readiness Level (TRL) assessment framework where the validation comparison metrics can be used as input to generate the corresponding overall qualification of the virtual testing tools.

Within the United Nations Economic Commission for Europe (UNECE) Validation Method for Automated Driving Subject Group (VMAD SG2) on developing a credibility assessment framework for simulation and virtual testing, four properties of a virtual testing platform have been identified for carrying out the credibility assessments, which are: **Capability**, **Accuracy**, **Correctness** and **Usability**. Based on this foundation, further quantifiable definitions for each of the properties above can be developed and introduced. **Capability** indicates what the virtual testing tools can do/offer; a direct measurable property would be the overall coverage of the physical environment testing data required. In this case, a 100% coverage would mean that the virtual testing platform can provide data output for all the required functionalities. **Accuracy** can be further divided into two aspects: 1) how well the virtual testing platform is performing regarding the reproducibility (i.e., how repeatable are the tests?) and 2) how well the virtual testing output correlates to the physical testing output. Both aspects of the accuracy can be quantified and assessed against a user-defined threshold. Correctness in the quantifiable definition represents the correlation between the virtual data and physical test data, i.e., whether the virtual data and algorithms can constantly output data that share strong correlation with the physical data. It differs from the accuracy since the correctness indicates correlation rather than the direct comparison of absolute values; a constant negative correlation will still result in a reliable performance given that the necessary conversion is made. The usability, on the other hand, is not directly quantifiable, as it represents the training and experiences needed to perform the virtual testing. This is rather a qualitative measure than a quantitative measure. In the proposed TRL assessment framework, usability will not be included, however it can be an extension to the scheme and form the wider qualification scheme.

The flow in Figure 5.10 illustrates the assessment framework based on the output information from virtual and physical environments. First the capabilities of the virtual testing tool are calculated, compared with threshold 1, and if the threshold is not reached then the virtual testing tool will be failed. If passed, the data will then be used to calculate the accuracy. If the accuracy reaches threshold 2, the virtual testing tool will pass the quantitative validation with gold standard. If the accuracy threshold is not reached, the correlation between the virtual and physical data will then be calculated and compared with threshold 3; if strong correlation can be established then the virtual testing tool will pass the quantitative validation with silver standard.

If there is a weak correlation then the virtual testing tool will be disqualified. In a final stage, a user can choose to assess the usability also, and the previous pass result can then be assessed against the usability requirements to determine the final overall qualification.

Figure 5.10, TRL assessment framework



6 | Stakeholder engagement

The stakeholder engagement activities undertaken as part of this project have shaped the creation of the *UK Connected and Automated Mobility Roadmap: Simulation to 2030.* An interactive version of this roadmap, and accompanying report, can be found through Zenzic's website - zenzic.io.

6.1 Introduction

Background

The stakeholder engagement activity provided the opportunity to gather broader views on ADS simulation, including potential simulation business demand, and facilities needed for ADS simulation verification. A group of leading industry experts were identified as stakeholder representatives. 1-to-1 interviews were conducted during September and October 2021, and one multi-stakeholder meeting was conducted in September 2021.

Structure to this study

This study is presented in two parts:

- Section 6.2 provides information about the contributors and the questions asked
- Section 6.3 provides the contributors' responses and analysis

6.2 Stakeholders

This section provides a summary of the contributors.

Type / category

Contributors	
Organisation size / type	Number
SME	9
LE	2
Public	3
University	2

Contributors	
Role	Number
Developer	6
Supplier	4
Regulator	2
Researcher	3
Consultant	1

Experience level

The contributors had a variety of experience levels with ADS research, development, and assurance.

Contributors	
Personal ADS experience level	Number
<1 year	0
1-2 year	5
3-5 year	7
>5 year	4

Simulation questionnaire

Topic	Question
Development	What is your vision for the use of simulation in highly automated vehicle development?
	What about IP protection if sharing models, particularly supply chain?
Assurance	What is your vision for the use of simulation in highly automated vehicle assurance?

	Use of in-service data for continuous improvement of models?
	How is confidence in simulation results achieved?
Services	Who undertakes simulation, and who monitors/reviews simulation?
	What is the role of national digital twin providing operational data for e.g. routing, asset management?
	Current and anticipated industry needs for 3rd party simulation services
Industry needs	The potential for interoperable simulation to deliver value/benefits for the industry
	Industry expectations for simulation customer experience

6.3 Gathered input views

This chapter provides a collation and analysis of contributors' responses.

Coverage and justification

1 Simulation for development

ADS test and verification requires significant mileage testing, which cannot be covered by physical testing alone within sensible development timeframes. Simulation is needed to efficiently cover all test and verification work needed. Specific test cases arise which cannot be recreated in physical testing. Edge cases emerge which are not discovered through physical testing. Automated simulation is needed to investigate these test cases with sufficient parameter variation coverage and to improve simulation efficiency.

Simulation is needed to test all layers of ADS controls, in particular to examine the perception performance and the accuracy of object identification and placement. Human-machine interactions need to be investigated in a simulated environment before physical testing, to understand potential behaviour safely or assess operation in line with user expectations, which may vary, therefore other road users are included in simulation.

2 Simulation for assurance

There is a contrast between the existing assurance approach, which prescribes vehicle functionality in precise conditions, compared with ADS operation, where the amount of variation can no longer be prescribed. Assurance bodies need to consider the variability, and assess overall vehicle safety, rather than assess the tolerance to a prescribed function. Assurance auditing of the overall vehicle safety needs to assess the coverage, requiring a sufficient selection of scenarios to cover the ODD. The high number of scenarios and test cases involved can only be assessed efficiently by simulation, supporting physical testing. The manufacturer and operator

(as the ODD "owners") should propose the set of scenarios for audit, and the selection of simulation and physical testing for these scenarios. Too much reliance on simulation, where physical testing would have succeeded, will not be accepted. In particular, the simulation results must demonstrate where scenarios within the ODD cannot be sustained.

If in-service Software Over-The-Air (SOTA) updates will be permitted, which alter the ADS behaviour (not currently permitted), the assurance or re-certification activity will need to be supported by simulation. In-service gathered data and simulation could be used for performance prediction, which may lead to modified ODD tolerances and in turn updates to the ADS behaviour. It should be noted that successful simulation also demonstrates confidence to potential operators ahead of any deployment. As part of that, independent incident investigation may depend on simulation to understand events and find appropriate recommendations.

Accuracy and verification

1 Fidelity

Current levels of simulation fidelity provide sufficient resolution to flag expected issues, however work is required to improve fidelity to ensure trusted testing for other applications. Simulation environments for path planning and vehicle control are more advanced, whereas the current low fidelity of sensor modelling makes simulation for perception rather immature. Different tasks require different levels of fidelity, and therefore a variety of simulation tools are used across the toolchain. Modelling (the task of creating physical models of environments or components, and software control models) should be differentiated from simulation (the task of setting up and executing a time-dependent, or in some cases frequency-dependent, analysis). Key aspects of ADS controls can be imported as software models, but the simulation setup is most relevant, in particular the execution time step to be synchronised with target software run cycles and the exchange of data between models.

2 Sensor modelling

Sensor modelling is an immature area and a major research aspect for simulation, since the capability of current sensor models is limited in terms of the physics that are represented. Physics-based models are important because many edge cases depend on physical phenomena, which cannot be captured by empirical models. Sensor mounting introduces errors from ground truth measurement, and resulting inaccuracies have to be represented in models. Inaccuracies can also be amplified by piece-to-piece variation. Environmental factors should be represented in the modelled environment, and the sensor model should react appropriately to environmental factors on all physical dimensions. Other noise factors which distort sensor signals need to be modelled to adequately evaluate perception performance. Sensor model development and sensor model verification are burgeoning research areas.

3 Verification

Final results for ADS behaviour are the critical aspect for assurance, but these rely on verification of constituent elements (each sub model and execution) of the simulation.

As outlined in the Fidelity subsection-1 Fidelity and model verification (accuracy of physics representation and control implementation) should be differentiated from simulation verification (accuracy of computation). A broad range of test cases will be needed to build confidence, especially to investigate the ADS performance limits. Verification datasets generally contain comparative real-world data; many more Field Operational Trials (FOTs) will be needed, with data collected from full ADS stacks, to build up sufficient evidence. Logged recordings of ADSs in real-world operation can be replayed in simulation, and simulation models improved. Motorsport practices (where measured data is used to improve simulation models, and human drivers provide qualitative feedback about the recreation of physics in simulation) follow a similar format.

An interesting approach to environment model verification and sensor model verification, is to use a physical measurement of the environment to construct an initial digital twin, then simulate the environment containing an ADS that will construct a digital twin of the simulated environment it encounters, then replace the initial digital twin with the new digital twin from the ADS, and repeat. Further repetition using re-constructed digital twins will witness degradation of the simulated environment accuracy, and thus the effectiveness of the simulation can be evaluated. Various levels of physical and virtual representation exist on the continuum from simulation to physical testing. Further studies will be needed to build up evidence of the correlation between real-world testing, controlled environments, laboratory tests, X-in-the-Loop, and desktop simulation. Aspects of human-machine interaction on this continuum are also unknown. Edge cases and identified issues can be explored in parallel between simulation and controlled physical testing. Common criteria for acceptable and sufficient correlation between physical test and simulation results are yet to be defined and require more investigation. These criteria will also require definition of potential tolerances, along with distribution analysis, to find correlation with predicted outcomes.

4 Validated scenarios

One source of scenarios is automated simulation, including other road users. This method can be more likely to discover edge cases, but more careful validation is needed. The test cases based on scenarios need to recreate the possible environmental conditions, ADS conditions, and other road users, as closely as possible to trigger the same ADS behaviour as in the real world. Construction of simple scenarios must be done first to prove correct behaviour before complexity is increased (i.e., open roads first, followed by other static actors such as parked vehicles, followed by one other dynamic actor, followed by multi-vehicle traffic, etc.). Confidence in scenarios extrapolated from complex simulation is only possible after demonstrating correlation of simple scenarios. Descriptions of the expected ADS behaviour need to refer to a publicly accessible digital definition of an "ideal driver", which does not currently exist. This would be the input for deterministic ADS controls and be the verification baseline for adaptive ADS controls. The Highway Code provides a general guideline for the expected ADS behaviour. However, digitising the Highway Code to be deterministic and quantifiable is challenging, and further work will be needed.

5 Validated tools and methodology

Manufacturers and operators need to convince assurance bodies that simulation results are not only verified, but that simulation tools produce provably consistent results (i.e., tool validation), and that the simulation tools are used correctly to do so (i.e., method validation). However, assurance bodies are very unlikely to undertake simulation tool validation activities, so a common approach will need to be defined to provide evidence that the simulation tools and methods used are fit-for-purpose. These definitions will also aid repeated simulation when any investigations are required in the case of unexpected events. Alongside tool validation and method validation definitions, common approaches should be defined for data management, which will facilitate the traceability required for investigations.

The definition of common approaches needs to include requirements to demonstrate tools and methods are fit-for-purpose. These requirements should identify appropriate features but avoid specifics of applications (such as unique scenarios), which may potentially introduce anti-competitive preferences (similar to defeat device). It is unclear whether any definition of common test cases will help tool and method validation, and this will need to be investigated. Some developers may seek a defined list of accepted simulation tools from assurance bodies, as a means to ensuring acceptable results. However, assurance bodies typically work with manufacturers to review evidence for product assurance, rather than with tool developers to assure tools. The exception to this may be direct dialogue between assurance bodies and leading tool developers to improve understanding of each other's expectations and plans, to avoid repeat discussions with each manufacturer using their tools. (Dialogue between assurance bodies and ADS suppliers follows a similar format).

While physical modelling tools often require significant investment in licences, hardware, and training, such that manufacturers do not change tools unless absolutely necessary, ADS simulation and toolchains are still developing. Therefore, developers are likely to continue to employ multiple different tools with different strengths, such that single tool validation would not be effective at this stage. Work by ASAM to define potential common approaches, such as OpenSCENARIO, OpenCRG, OpenDRIVE, OpenLABEL, OpenODD, and OSI, will help with interchangeability of models and tools in future. It should be noted that simulation tools cannot be verified for functional safety, rather simulation results can be verified for accuracy, since so many variables are involved in ADS simulation. It is the ADS behaviour represented in simulation that must be verified for functional safety. Further work in defining common approaches, and potentially reviewing evidence from manufacturers and operators, will depend on the support of an expert group.

Users

1 Approach

As outlined in the1 Fidelity subsection, a variety of simulation tools are used across the toolchain. Many developers use in-house created tools, maintained by an in-house simulation organisation, specifically designed for selected tasks such as perception (using their preferred sensor suite) or path planning (based on their required behavioural competences). Many developers also prefer in-house simulation tools as this further protects IP.

Third party tools offer the potential to investigate the entire ADS stack (and several tool vendors are building frameworks to evaluate ADS stacks), or individual layers, but the immaturity of the tools and the divergent technical solutions, mean they currently require extensive tailoring. Third party tool providers will need to enable import of other third-party models into their simulation environment. There is a particular opportunity to provide sensor models, where sensor suppliers are unable to provide physical models of sufficient fidelity. It can be difficult for suppliers to share detailed representative sensor models, since these often reflect IP of the sensor itself. Alternatively, mimicking sensors with generic sensor models introduces potentially large errors. Common approaches to tool validation, method validation, and data management, could help to facilitate simulation activities by third parties. The involvement of an expert group in defining common approaches, and potentially reviewing evidence from manufacturers and operators, will support assurance bodies in the near-term (est. until 2030), but thereafter may facilitate independent simulation by third parties, with appropriate evaluation in place.

2 Simulation responsibility / review responsibility

Presentation of safety assurance evidence from simulation is the responsibility of manufacturers, backed up by verification of the results (through physical test comparison), tools, methods, and data management. Review of the results and wider verification evidence is the responsibility of assurance bodies. Assurance bodies will need to build up in-house skills to review simulation results (potentially witnessing simulation execution) and verification evidence of the correct use of simulation. They will also need to understand further actions to be undertaken when evidence presented from simulation does not pass audit, if the simulation should be improved and re-run or if additional physical tests are required.

In the near-term (est. until 2030), assurance bodies are very unlikely to undertake simulation, either independent simulation or re-run manufacturers' simulation. In the long-term (est. 2030 onwards), assurance bodies may build up in-house simulation execution skills to undertake simulation, which may require gaining more access to manufacturers' IP. This may well be influenced by the decision to use standard test cases as part of the assurance process. In the near-term (est. until 2030), assurance bodies are likely to be supported by expert groups and third parties' capabilities. Further stakeholders, such as operators and insurers, may seek access to the outcomes of assurance bodies' audit of simulation evidence, to confirm manufacturers' data or insured risk levels.

3 Simulation services business need

Outsourcing simulation activity is not currently commonplace, potentially only to very few highly trusted partners, due to developers' need to protect IP. In general, current third-party simulation services present difficulties for IP protection, and efforts to mask IP, such as conducting grey-box simulation, do not provide sufficient analysis to offer value to developers. Simulation, using protected third-party models, such as sensor models, supports a black-box approach. This offers IP protection, but the limited visibility means alternative verification evidence is required from, for example, the model provider. Co-simulation using multiple simulator tools in parallel, potentially with cloud-based computing, is also useful to protect IP, particularly to avoid distributing ADS control software.

'Interoperable simulation' needs clearer definition. Stakeholders consulted expressed that distributed multi-site co-simulation is unlikely to attract significant commercial usage. In general, medium and large OEMs currently prefer to execute simulation in-house while small developers have budgetary challenges with outsourcing simulation work. Interchangeable simulation will deliver more immediate value to ADS developers since improvements to third party models can then be integrated with other latest models.

Facilities for simulation verification and digital twins

Developers of vehicles that depend upon an environment model (i.e., to support localisation) currently need to scan the ODD to generate a static model. Subsequent scans may be needed to maintain static models up to date, which becomes a costly exercise. Several different developers may need to scan the same location, leading to opportunities for efficiencies. There is also significant value in the static model, or 'digital twin', to other potential users, such as road transport management authorities. Therefore, a publicly produced and maintained digital twin could become an important national asset. The most common facilities used by sub system and component developers are sensor labs. These are also occasionally used by ADS developers to undertake sensor characterisation. The most common facility used by simulation teams (from any stakeholder) is high performance computing facilities or cloud.

Facilities for certification

If assurance bodies' expectations for simulation require additional computing capabilities, manufacturers are likely to expect assurance bodies to ensure fit-for-purpose computing facilities are accessible (similar to arrangements for conventional vehicle tests) Other sectors, such as aerospace and defence, have already addressed similar challenges around test facilities for assurance, and developers believe there are opportunities for the automotive/ADS sector to take onboard prior cross-sector learning.

7 | References

ASAM e.V. Available at: https://www.asam.net/.

'ASAM OpenLABEL1.0'. Available at: https://www.asam.net/standards/detail/openlabel/.

'ASAM OpenSCENARIO 1.1.1'. Available at: https://www.asam.net/standards/detail/openscenario/.

Balfe, N., Sharples, S. and Wilson, J. R. (2015) 'Impact of automation : Measurement of performance , workload and behaviour in a complex control environment', *Applied Ergonomics*, 47, pp. 52–64.

Bock, F. *et al.* (2019) 'Advantageous usage of textual domain-specific languages for scenario-driven development of automated driving functions', *SysCon 2019 - 13th Annual IEEE International Systems Conference, Proceedings*.

Brackstone, M. *et al.* (2019) 'OmniCAV: A Simulation and Modelling System that enables CAVs for AI', *IEEE Intelligent Transportation Systems Conference (ITSC)*, 2020, pp. 2190–2195. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8917103.

Chen, S. *et al.* (2020) 'Identifying Accident Causes of Driver-Vehicle Interactions Using System Theoretic Process Analysis (STPA)', *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020-Octob, pp. 3247–3253. doi: 10.1109/SMC42975.2020.9282848.

Dupuis, M., Hekele, E. and Biehn, A. (2019) 'OpenDRIVE® Format Specification, Rev. 1.5', *OpenDRIVE*, (M), pp. 1–133. Available at: www.vires.com.

Esenturk, E. *et al.* (2021) 'Analyzing Real-world Accidents for Test Scenario Generation for Automated Vehicles', in *IEEE Intelligent Vehicles Symposium 2021*.

Euro NCAP (2017) *Euro NCAP 2025 Roadmap: In pursuit of Vision Zero*, *Euro NCAP Technical Paper*.

EuroNCAP (2017) 'EUROPEAN NEW CAR ASSESSMENT PROGRAMME (Euro NCAP) - Speed Assist Systems - Test Protocol', (March).

Fagnant, D. J. and Kockelman, K. (2015) 'Preparing a nation for autonomous vehicles : opportunities , barriers and policy recommendations', *Transportation Research Part A*, 77, pp. 167–181.

Fagnant, D. J. and Kockelman, K. M. (2014) 'The travel and environmental implications of shared autonomous vehicles , using agent-based model scenarios', *Transportation Research Part C: Emerging Technologies*, 40, pp. 1–13.

Gangopadhyay, B. *et al.* (2019) 'Identification of Test Cases for Automated Driving Systems Using Bayesian Optimization', *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, pp. 1961–1967. doi: 10.1109/ITSC.2019.8917103.

Gotoda, H. (2003) '3D shape comparison using multiview images', *NII Journal*, pp. 19–25.

Huang, J. and You, S. (2012) *Point cloud matching based on 3D self-similarity*, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. doi:

10.1109/CVPRW.2012.6238913.

ISO (2021) *Intelligent transport systems — Low-Speed Automated Driving (LSAD) Systems for Predefined routes — Performance requirements, system requirements and performance test procedures - ISO 22737*.

Kalra, N. and Paddock, S. M. (2016) 'Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?', *Transp. Res. Part A Policy Pract*, 94, pp. 182–193.

Khastgir, S. *et al.* (2017) 'Test Scenario Generation for Driving Simulators Using Constrained Randomization Technique', *SAE Technical Papers*, 2017-March(March). doi: 10.4271/2017-01-1672.

Knabe, E. (2019) 'SimS. Simulation Scenarios. Public report'. Available at: www.vinnova.se/ffi.

Menzel, T. *et al.* (2019) 'From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment', *IEEE Intelligent Vehicles Symposium, Proceedings*, 2019-June, pp. 2383–2390.

Menzel, T., Bagschik, G. and Maurer, M. (2018) 'Scenarios for development, test and validation of automated vehicles', *arXiv*, 2018-June(Iv), pp. 1821–1827.

Muliukha, V., Lukashin, A. and Ilyashenko, A. (2019) 'An Intelligent Method for Comparing Shapes of Three-Dimensional Objects', in *2019 25th Conference of Open Innovations Association (FRUCT)*, pp. 234–240. doi: 10.23919/FRUCT48121.2019.8981528.

NCAP (2020) 'Euro NCAP Test Protocol-AEB VRU systems', *November*, (June). Available at: https://cdn.euroncap.com/media/58226/euro-ncap-aeb-vru-test-protocol-v303.pdf.

Neurohr, C. *et al.* (2021) 'Criticality Analysis for the Verification and Validation of Automated Vehicles', *IEEE Access*, 9(i). doi: 10.1109/ACCESS.2021.3053159.

Novotni, M. and Klein, R. (2001) 'A geometric approach to 3D object comparison', in *Proceedings International Conference on Shape Modeling and Applications*, pp. 167–175. doi: 10.1109/SMA.2001.923387.

'Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) – Specification' (2020) *The British Standards Institution, BSI PAS 1883*.

Protocol Buffers. Available at: https://developers.google.com/protocol-buffers.

Safety Pool Database Access. Available at: www.safetypooldb.ai.

Safety Pool Scenario Database. Available at: https://www.safetypool.ai/.

Tas, O. S. *et al.* (2016) 'Functional system architectures towards fully automated driving', *IEEE Intelligent Vehicles Symposium, Proceedings*, 2016-Augus(Iv), pp. 304–309. doi: 10.1109/IVS.2016.7535402.

Tenbrock, A. *et al.* (2021) 'The ConScenD Dataset: Concrete Scenarios from the highD Dataset According to ALKS Regulation UNECE R157 in OpenX'. Available at: http://arxiv.org/abs/2103.09772.

UK Department for Transport (2020) 'Road Safety Data - STATS19'.

Ulbrich, S. *et al.* (2015) 'Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2015-Octob, pp. 982–988.

UNECE (2020) 'R157', 1958(March 1958).

Urbach, D., Ben-Shabat, Y. and Lindenbaum, M. (2020) 'DPDist: Comparing Point Clouds Using Deep Point Cloud Distance BT - Computer Vision – ECCV 2020', in Vedaldi, A. et al. (eds). Cham: Springer International Publishing, pp. 545–560.

Le Vine, S. *et al.* (2016) 'Automated cars : Queue discharge at signalized intersections with " Assured-Clear-Distance-Ahead " driving strategies', *Transportation Research Part C: Emerging Technologies*, 62, pp. 35–54.

Zhang, X. *et al.* (2021) 'Test Framework for Automatic Test Case Generation and Execution Aimed at Developing Trustworthy AVs from Both Verifiability and Certifiability Aspects', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2021-Septe, pp. 312–319. doi: 10.1109/ITSC48978.2021.9564542.

Zhang, X., Khastgir, S. and Jennings, P. (2020a) 'Scenario Description Language for Automated Driving Systems: A Two Level Abstraction Approach', in *Proc. of the 2020 IEEE International Conference on Systems, Man and Cybernetics (SMC)*.

Zhang, X., Khastgir, S. and Jennings, P. (2020b) 'Scenario Description Language for Automated Driving Systems: A Two Level Abstraction Approach', *IEEE International Conference on Systems, Man and Cybernetics (SMC)*.

Zhou, T. *et al.* (2020) 'COMPARATIVE EVALUATION OF DERIVED IMAGE AND LIDAR POINT CLOUDS FROM UAV-BASED MOBILE MAPPING SYSTEMS', *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2, pp. 169–175. doi: 10.5194/isprs-archives-XLIII-B2-2020-169-2020.

ZENZIC

SELF-DRIVING REVOLUTION

To find out more, please contact: info@zenzic.io