Test and validation of perception-based ADAS: modern solutions to traditional challenges

Raul Sena Ferreira

Continental Engineering Services, Toulouse, France, raul.ferreira@continental.com

Abstract - *Perception-based advanced driver-assistance systems (ADAS) are widely applied in modern vehicles to assist drivers by increasing the vehicle's safe operation. This perception-based component usually has the advantage of perceiving the environment with high accuracy through modern deep-learning (DL) models embedded in the vehicle. However, testing and validating DL-based perception functions for safety-critical autonomous systems is a crucial task. The reason is that accurate DL models applied in computer vision tasks still fail in scenarios where humans perform well. Besides, the huge space of combinations between pixels and other sensor values is still an open problem. It brings important challenges such as scenario generation relevance, scenario coverage, resource-hungry graphical processing, the stochastic nature of the DL model outputs, and so on. Besides, we still have other challenges such as decreasing the amount of manual work in tasks for preparing the tests and evaluations (e.g., translating from object domain design specifications and requirements to generated concrete scenarios). Therefore, we present a framework for testing and validation of perception-based ADAS that implements solutions for the aforementioned challenges. Moreover, this framework is based on ongoing European R&D projects such as SUNRISE, and automotive industry expertise.*

Keywords: *perception, testing and validation, simulation, deep learning, ADAS.*

Introduction

State-of-the-art DL models are widely applied to several domains such as public policy (Ferreira, et al., 2017), power utility (Dal Pont, et al., 2019), and also in safety-critical tasks such as surgical robots (Marcus, et al., 2024) and self-driving cars (Ganesan, et al., 2024). However, perception-based DL models tend to have problems such as ghost detections (Bogdoll, Nitsche, and Zöllner, 2022), misclassifications (Ferreira, et al., 2023), or even being blind to the presence of new objects (Sabokrou, et al., 2018). Hence, testing and validating these perception functions is important, especially if we want to find corner cases that lead to hazards at runtime. Moreover, as explained by Hoss, Scholtes, and Eckstein (2022), the realization of safety-oriented perception testing remains an open issue since challenges concerning the three testing axes: (1) test criteria and metrics, (2) test scenarios, and (3) reference data, and their interdependencies currently do not appear to be sufficiently solved in the literature.

In the literature, several works are dedicated to simulation but due to the complexity of each part of a simulation chain, mostly of these works focus on specific parts of the simulation such as finding safety violations on control (Li, et al., 2020), system specifications (Zapridou, Bartocci, and Katsaros, 2020), error space exploration (Singh, et al., 2021), path planning (Arcaini, et al., 2021), and driving coverage-based scenes generation (Hu, et al., 2021). Despite the scarcity of complete solutions, we still find recent works focusing on perception-based ADAS, such as presented by Putter, et al. (2023). The authors proposed a validation and safety assessment strategy for a perception-based crash severity prediction function. They apply ISO 26262 and ISO/PAS 21448 standards (Kirovskii and Gorelov, 2019) and test it in a data-driven scenario-based testing approach.

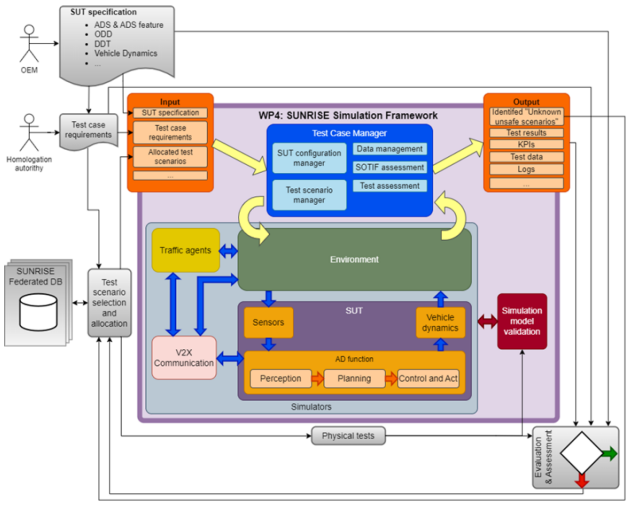


Figure 1: SUNRISE safety assurance framework.

The difference between our framework and related works is that we developed a complete framework from ODD parsing up to the deployment simulation. This approach is supported by recent testing object-based environment perception for safe automated driving studies (Hoss, Scholtes, and Eckstein, 2022) and based on the ongoing European

Horizon 2020 project called SUNRISE (safety assurance framework for connected and automated mobility systems) †, illustrated in Figure 1. Our framework extends the purpose of the SUNRISE project since we tackle the safety assurance of automated vehicles of Levels 4 and 5 instead of 2. Such extension increases the difficulties of implementation because the environmental perception of such AVs is subject to uncertainties (Hoss, Scholtes, and Eckstein, 2022).

This work is organized as follows: Section “Methodology”, explains the concepts applied in our approach and the implementation’s architecture. Afterward, we detail the three main aspects of the architecture of our approach and introduce its main functionalities. In Section “Results”, we show the first results and the insights provided by them. In Section “Conclusion”, we present our final considerations.

Methodology

To harmonize the virtual validation framework implementation, collaborative work among the SUNRISE project enabled the definition of the main subsystems to cover all the aspects of the virtual validation pipeline. In Figure 1, we illustrate the reference methodology developed on the SUNRISE safety assurance framework. An illustration of each part of the final implemented architecture is shown in Figure 2. We use the Carla simulator (Dosovitskiy, et al., 2017) since it is open-source and is widely adopted in the industry and academia. All interfaces are connected by ROS 2 for easy integration and sharing among different simulators and systems.

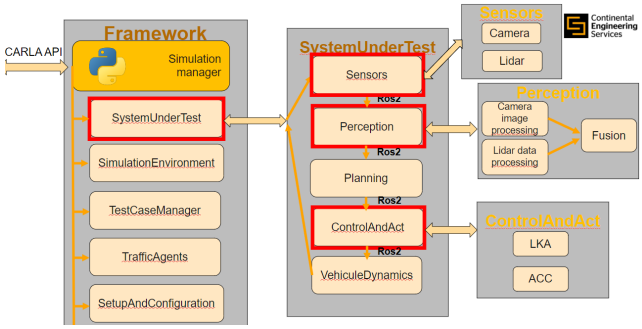


Figure 2: Logical implementation of our framework.

The framework is connected to the CARLA simulator as an external module. The simulation manager controls the communication between the other subsystems (e.g., the system under test, simulation environment, etc). The main subsystems are presented as follows:

System under Test: It contains all the elements existent in the ADAS. It includes sensors (e.g., camera and lidar), perception functions (e.g., camera and lidar processing, fusion algorithms), planning algorithms, control and act strategies (e.g., lane keep assistant, and active cruise control), and vehicle dynamics.

† https://ccam-sunrise-project.eu/

Simulation environment: It creates the environment in which the ADAS will be tested. It contains a set of environmental conditions and is linked to the traffic agents subsystem.

Test case manager: It manages inputs (e.g., ODD and test scenarios), and outputs of the simulation framework. It is also responsible for parsing predefined test requirements to test scenarios in the simulation framework.

Traffic agents: It provides control of dynamic elements surrounding the ego vehicle in the simulation. This part implements behavioral and control models of the traffic participants that can belong to different description levels.

Setup and configuration: It contains all necessary configurations, libraries, and internal parameters to perform the simulation.

Besides, we have also a dedicated module for evaluation and analysis of the results of the simulation. However, we omitted details of this part since it’s known to be a huge part of a simulation platform and it is outside of the scope of this paper.

In the next section, we give general details about the main parts of our framework: scenario generation, coverage, and evaluation.

Testing and validation of perception-based ADAS

In this section, we briefly explain how we generate and evaluate the coverage of scenarios taking into consideration the huge combination of pixel space, traffic participants parameterization, and other continuous parameters existent in a simulation.

Scenario Generation

Scenario simulation is an essential part of testing since not all DL errors lead to hazards, and most of the time, the physics of such autonomous systems need to be considered when searching for hazards (Ferreira, et al., 2022). In this part, we apply a library called SIMOOD (Ferreira, et al., 2022) that is dedicated to testing perception functions built with DL that are integrated within the CARLA simulator.

SIMOOD applies image perturbations during simulation instead of using it over static datasets as usually done in the literature (Ferreira, et al., 2021; Hendrycks and Dietterich, 2019). Since each image perturbation has several parameters, the number of possible simulations to find a hazard grows exponentially and the time to simulate even a tiny part of these simulations grows quickly as well.

Integrating SIMOOD into our framework helps to tackle the two problems mentioned above when generating graphical scenarios for tests. It uses a two-step approach. First, it performs a unit testing of the ML model using a genetic algorithm (Mirjalili, 2019) on the same dataset applied to train the ML model. It determines which combination of image perturbations with their respective intensity levels increases the number of incorrect predictions. Second, the selected perturbations are applied to the simulation to

verify if they lead to hazards in the system. That is, SIMOOD takes a set of existing scenarios injects image perturbations selected by a genetic-algorithm-on-data approach, and posteriorly tests them on simulations, proving itself capable of turning safe scenarios into unsafe ones, which can be analyzed to improve the system's safety. Since SIMOOD tries to find corner cases only in the space of pixels, we still need to perform the coverage of the other parts of the simulation, as explained below.

Scenario Coverage and evaluation

Scenario generation coverage: Effective testing of ADAS requires a wide range of scenarios with different weather conditions, traffic densities, road types, and other environmental factors. Manual scenario creation is time-consuming and prone to oversight. Therefore, we use genetic algorithms (GAs) (Yang, 2014) to automate the generation of diverse and challenging driving scenarios in the CARLA simulator. A simplified example is shown below:

- *Scenario Representation*: Each driving scenario is represented as a chromosome in the GA, where each gene corresponds to a specific scenario parameter. This allows the algorithm to explore a vast combinatorial space of potential scenarios.
  - Weather conditions: sunny, rainy, foggy
  - Time of day: morning, noon, night
  - Traffic density: low, medium, high
  - Road type: urban, highway, rural
  - Pedestrian density: low, medium, high
  - Construction presence: none, some, many
  - Accident presence: none, minor, major
- *Initial Population*: An initial population of scenarios is generated with random values for each parameter. For example, we tested with an initial population of 20 randomly choosing between the categories mentioned above (e.g., road type, traffic density, etc).
- *Fitness Function*: The fitness function evaluates each scenario based on its ability to cover diverse and challenging conditions. More challenging scenarios (e.g., foggy weather, nighttime, high traffic) receive higher scores.
- *Selection*: It identifies the top-performing scenarios to serve as parents for the next generation. This ensures that high-quality scenarios are more likely to contribute to the subsequent population. We tested with a number of ten parents.
- *Crossover and Mutation*: New offspring are generated by combining genes from two parent scenarios, facilitating the exploration of new scenario combinations. Hence, mutation introduces random changes to some scenarios, maintaining genetic diversity and preventing premature convergence. For example, a random choice between ['sunny', 'rainy', 'foggy']. The tested mutation rate for our experiments was set to 0.1.

Finally, the new population is formed by combining the selected parents and the newly generated offspring, ready for the next generation. The process is iterated over 50 generations to evolve the population of scenarios to maximize coverage and diversity.

Scenario coverage evaluation: Evaluating the coverage strategy presented above can be achieved using various algorithms that quantify how well the generated scenarios cover the desired space of driving conditions and edge cases. The result is a score between 0 and 1, where 0 indicates no coverage and 1 indicates full coverage.

We apply a weighted sum of multiple metrics. This method combines multiple metrics to provide a comprehensive coverage score by combining diversity, parameter space coverage, and edge case inclusion into a single score. Below, we briefly explain each one of these metrics:

- *Coverage Metric Based on Scenario Diversity*: This method evaluates the diversity of the generated scenarios and assigns a score based on the variety and uniqueness of the scenarios. It uses the Shannon diversity index (Konopiński, 2020) which can be used to measure the diversity of scenario parameters. It calculates the proportion of each unique scenario in the population and assigns a higher score to more diverse populations.
- *Coverage Metric Based on Parameter Space Exploration*: This method evaluates how well the parameter space is covered by the generated scenarios. It uses a Parameter Space Coverage (Burrage, et al., 2014). This metric assesses the extent to which each parameter's possible values are represented in the scenario population. The score is normalized to the range 0-1.
- *Coverage Metric Based on Edge Case Inclusion*: This method evaluates how well the generated scenarios include critical edge cases. It uses an approach called Edge Case Inclusion (Bjorklund and Husfeldt, 2006). This metric assesses the presence of predefined critical edge cases in the population and assigns a higher score to populations that include more of these cases.

Finally, the weighted sum of the metrics is calculated based on the score of aforementioned metrics multiplied by importance weights. We calculate the final weighted sum score ws as follows:

$$ws = (\alpha * ds) + (\phi * ps) + (\rho * es)$$

Where ds means diversity score, ps means parameter coverage score, and es means edge case score. If we want to give more importance to diversity and the coverage of parameters than to the search of edge cases, the weights for each score could be respectively $\alpha = 0.4, \phi = 0.4, \rho = 0.2$.

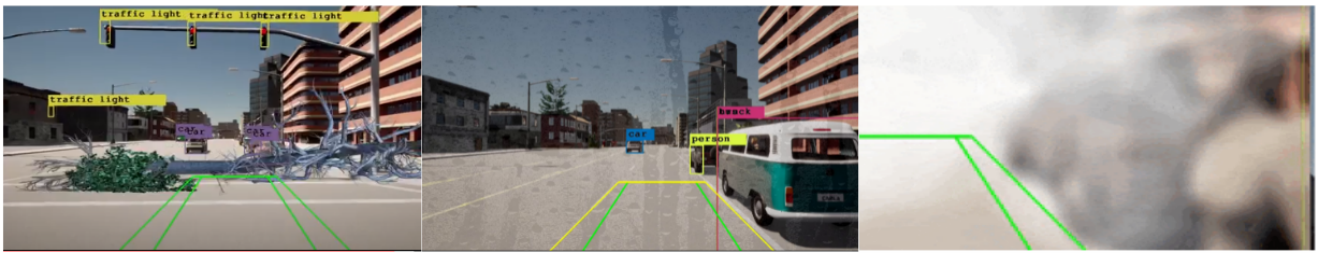
Next, we present results for the scenario generation.

Results

Below, we grouped all results for *scenario generation* on one page for easy readability.

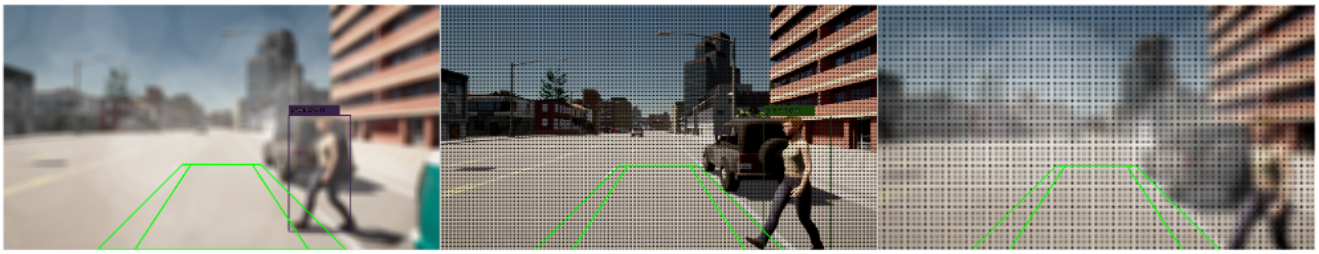
Conclusion and future work

In this paper, we presented an ongoing framework for scenario generation and test case validation for perception-based ADAS. It combines the expertise provided by the SUNRISE project and integration with recent works libraries proposed in the literature for testing perception functions in AVs. Moreover, we customized and combined traditional approaches that increase the testing coverage over a huge search space, covering both the space of pixels and the parametric space of traffic participants. The framework is already integrated with CARLA, an



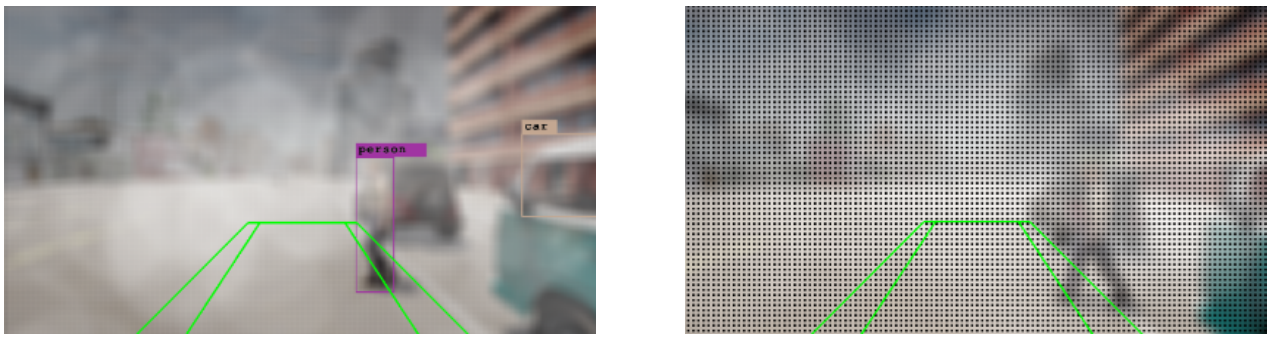
a) Novelty class b) Condensed water c) Heavy smoke

Figure 3: Hazards uncovered by applying single image perturbations.
a) An accident due to a *unknown* object (tree) not detected by the ML model; **b)** A dangerous stop due to a false detection (ghost pedestrian) provoked by condensed water on the camera lens; **c)** An accident due to a *known* object (pedestrian) not detected by the ML model when exposed to heavy smoke in the environment.



a) Smoke (0.25) b) Grid dropout (2) c) Smoke(0.18) + Grid dropout(2)

Figure 4: Hazards uncovered by combining two types of image perturbations.
a) ADAS detects a pedestrian in an environment with light smoke (intensity 0.25); **b)** ADAS detects a pedestrian despite a grid dropout failure on the camera sensor (intensity 2); **c)** A collision due to a failure of ADAS on detecting a pedestrian when both conditions happen even with smoke transformation having a lower intensity than before.



(a) Grid dropout (1) + smoke (0.3). (b) Smoke (0.3) + grid dropout (1).

Figure 5: Same image perturbations in different order produce different outcomes, uncovering new hazards.
a) ADAS detects a pedestrian in the scenario with the image combination *grid dropout + smoke*. **b)** ADAS does not detect a pedestrian in the same scenario but with image perturbations in inverted order.

open-source simulator for AV, and generates scenarios that lead to hazards during simulation.

Worth mentioning, that due to space restrictions and confidentiality, we could not add more details about our solution and provide more results and analysis from it. However, we provide details on the main parts of this framework which we believe can provide valuable insights to the community.

As a next step, we intend to add an ODD parser that translates from requirements to logical scenarios to be simulated inside the framework using the help of large language models (LLM) (Yang, et al., 2024). For example, applying finetuned LLMs such as the open-source code llama (Roziere, et al., 2023) to generate from natural language ODD descriptions to openscenario (Chen, et al., 2022) and opendrive (Dupuis, Strobl, and Grezlikowski, 2010) files, ready to be interpreted by the CARLA simulator. New safety-oriented metrics (Guerin, et al., 2022) can also be added to enrich the analysis.

Acknowledgements

The SUNRISE project is funded by the European Union's Horizon Europe Research & Innovation Actions under grant agreement No. 101069573.

We also thank Sergey Abrashov for initiating the development of the presented framework, and all team members from Continental Engineering Services (CES) who worked on it.

References

Arcaini, P., Calò, A., Ishikawa, F., Laurent, T., Zhang, X.-Y., Ali, S., Hauer, F., and Ventresque, A., 2021. Parameter-Based Testing and Debugging of Autonomous Driving Systems. In: *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*. IEEE, pp. 197–202.

Bjorklund, A. and Husfeldt, T., 2006. Inclusion–exclusion algorithms for counting set partitions. In: *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. IEEE, pp. 575–582.

Bogdoll, D., Nitsche, M., and Zöllner, J. M., 2022. Anomaly Detection in Autonomous Driving: A Survey. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4488–4499.

Burrage, K., Burrage, P. M., Donovan, D., McCourt, T. A., and Thompson, H. B., 2014. Estimates on the coverage of parameter space using populations of models. *Environ. Water Resour. Manag*, 10, pp. 2014–813.

Chen, H., Ren, H., Li, R., Yang, G., and Ma, S., 2022. Generating autonomous driving test scenarios based on openscenario. In: *2022 9th International Conference on Dependable Systems and Their Applications (DSA)*. IEEE, pp. 650–658.

Dal Pont, M. P., Ferreira, R. S., Teixeira, W. W., Lima, D. M., and Normey-Rico, J. E., 2019. MPC with machine learning applied to resource allocation problem using lambda architecture. *IFAC-PapersOnLine*, 52(1), pp. 550–555.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V., 2017. CARLA: An Open Urban Driving Simulator. In: *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16.

Dupuis, M., Strobl, M., and Grezlikowski, H., 2010. Opendrive 2010 and beyond–status and future of the de facto standard for the description of road networks. In: *Proc. of the Driving Simulation Conference Europe*, pp. 231–242.

Ferreira, R., Praia, V., Bonecini, F., Vieira, A., Lopez, F., et al., 2017. Platform of the brazilian csos: open government data and crowdsourcing for the promotion of citizenship. In: *Anais do XIII Simpósio Brasileiro de Sistemas de Informação*. SBC, pp. 17–24.

Ferreira, R. S., Arlat, J., Guiochet, J., and Waeselynck, H., 2021. Benchmarking safety monitors for image classifiers with machine learning. In: *2021 IEEE 26th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, pp. 7–16.

Ferreira, R. S., Guérin, J., Guiochet, J., and Waeselynck, H., 2022. Simood: Evolutionary testing simulation with out-of-distribution images. In: *2022 IEEE 27th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, pp. 68–77.

Ferreira, R. S., Guerin, J., Guiochet, J., and Waeselynck, H., 2023. Sena: Similarity-based error-checking of neural activations. In: *ECAI 2023*. IOS Press, pp. 724–731.

Ganesan, M., Kandhasamy, S., Chokkalingam, B., and Mihet-Popa, L., 2024. A Comprehensive Review on Deep Learning-Based Motion Planning and End-To-End Learning for Self-Driving Vehicle. *IEEE Access*.

Guerin, J., Ferreira, R. S., Delmas, K., and Guiochet, J., 2022. Unifying Evaluation of Machine Learning Safety Monitors. *arXiv preprint arXiv:2208.14660*.

Hendrycks, D. and Dietterich, T., 2019. Benchmarking neural network robustness to common corruptions and perturbations. In: *International conference on learning representations (ICLR), New Orleans, United States*.

Hoss, M., Scholtes, M., and Eckstein, L., 2022. A review of testing object-based environment perception for safe automated driving. *Automotive Innovation*, 5(3), pp. 223–250.

Hu, Z., Guo, S., Zhong, Z., and Li, K., 2021. Coverage-based Scene Fuzzing for Virtual Autonomous Driving Testing. *arXiv preprint arXiv:2106.00873*.

Kirovskii, O. and Gorelov, V., 2019. Driver assistance systems: analysis, tests and the safety case. ISO 26262 and ISO PAS 21448. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 534. 1. IOP Publishing, p. 012019.

Konopiński, M. K., 2020. Shannon diversity index: a call to replace the original Shannonâs formula with unbiased estimator in the population genetics studies. *PeerJ*, 8, e9391.

Li, G., Li, Y., Jha, S., Tsai, T., Sullivan, M., Hari, S. K. S., Kalbarczyk, Z., and Iyer, R., 2020. Av-fuzzer: Finding safety violations in autonomous driving systems. In: *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, pp. 25–36.

Marcus, H. J., Ramirez, P. T., Khan, D. Z., Layard Horsfall, H., Hanrahan, J. G., Williams, S. C., Beard, D. J., Bhat, R., Catchpole, K., Cook, A., et al., 2024. The IDEAL framework for surgical robotics: development, comparative evaluation and long-term monitoring. *Nature medicine*, 30(1), pp. 61–75.

Mirjalili, S., 2019. Genetic algorithm. In: *Evolutionary algorithms and neural networks*. Springer, pp. 43–55.

Putter, R., Neubohn, A., Leschke, A., and Lachmayer, R., 2023. Predictive Vehicle SafetyâValidation Strategy of a Perception-Based Crash Severity Prediction Function. *Applied Sciences*, 13(11), p. 6750.

Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Remez, T., Rapin, J., et al., 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Sabokrou, M., Khalooei, M., Fathy, M., and Adeli, E., 2018. Adversarially learned one-class classifier for novelty detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3379–3388.

Singh, V., Hari, S. K. S., Tsai, T., and Pitale, M., 2021. Simulation Driven Design and Test for Safety of AI Based Autonomous Vehicles. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 122–128.

Yang, X.-S., 2014. Chapter 5 - Genetic Algorithms. In: X.-S. Yang ed. *Nature-Inspired Optimization Algorithms*. Oxford: Elsevier, pp. 77–87. ISBN: 978-0-12-416743-8.

Yang, Z., Liu, F., Yu, Z., Keung, J. W., Li, J., Liu, S., Hong, Y., Ma, X., Jin, Z., and Li, G., 2024. Exploring and unleashing the power of large language models in automated code translation. *Proceedings of the ACM on Software Engineering*, 1(FSE), pp. 1585–1608.

Zapridou, E., Bartocci, E., and Katsaros, P., 2020. Runtime verification of autonomous driving systems in carla. In: *International Conference on Runtime Verification*. Springer, pp. 172–183.