# Automatic Test Evaluation for Driving Scenarios Using Abstraction Level Constraints

Steffen Wittel*, Daniel Ulmer* and Oliver Bühler*
*Steinbeis Interagierende Systeme GmbH, Esslingen, Germany
Email: {steffen.wittel,daniel.ulmer,oliver.buehler}@interagierende-systeme.de

***Abstract*—Sophisticated Driver Assistance Systems (DASs) on the way to highly automated driving require extensive testing activities to verify the functionality and the safety of the developed software. With each step towards autonomous driving, the automobile manufacturers are increasingly taking on responsibility for driving maneuvers automatically performed by such systems in unknown environmental situations. Whereas recent DASs use the driver as fallback, in the future this fallback will be only available after a legally prescribed period of time since the driver might be distracted by other activities. To take account of this, robustness testing becomes more and more important to ensure a safe operation at different environments. This paper presents a constraint based approach that applies automatic testing to evaluate DASs. Thereby, the focus is set on the determination of the expected responses that are the basis for the automatic evaluation of the generated test scenarios. The introduced approach is working on different levels of abstraction in combination with an analysis of the observed behavior to classify individual situations of the scenarios after the test execution. The approach enables a full automation of the evaluation, which is the bottleneck of current state-of-the-art scenario generators.**

***Keywords–Automotive Testing; Test Generation; Test Evaluation; Test Automation.***

## I. INTRODUCTION

The decreasing development times and product cycles in combination with technical advances already require a high testing effort to ensure that the vehicle's built-in DASs are working correctly. It is expected that with each step in the direction to highly automated driving, the testing effort increases to cover the new functionality and to ensure a safe operation of the vehicle.

While the first assistance systems, like the Electronic Stability Control (ESC) [1] or the Antilock Braking System (ABS) [1], were intervening in critical situations, only the current generation of DASs supports the driver during his entire drive, but without taking on responsibility for the maneuvers performed. Even during an intervention of a DAS, the driver is still responsible for the vehicle and the possible damage. On the way to highly automated driving, this responsibility of the driver will be only given after a legally prescribed time limit, because it is assumed that the driver is distracted by other activities and can only handle the situation after a certain time.

Additionally for autonomous driving, it is not sufficient that systems are working as expected in a defined environment, but also in unknown situations. Each drive is different from the previous one, e.g., in respect to the environmental conditions like traffic or weather. An early and safe hand over to the driver would be a technical solution. But especially in the premium market, the customers do not tolerate a system, which is rarely available. The automobile manufactures have thus to find a balance between safety and availability.

The automatic test generation is an approach, which is not intended to replace the already performed testing, but rather extends them to cover a broad functional range of a system by creating a large number of different test cases. In most cases, the commercial off-the-shelf scenario generators do not determine the test result and leave it to the tester to define the expected behavior of the System Under Test (SUT), which limits the degree of automation. To get around this, an approach is presented to determine the expected response of the SUT using constraints on different levels of abstraction. An automatic analysis of the observed behavior supplements the approach to classify individual situations of the scenarios after the test execution.

The following section shows the related work. Section III and Section IV of this paper give an overview about the SUT and the automatic testing. In Section V, the approach for an automatic evaluation of the generated test scenarios is presented. Finally, Section VI shows a case study.

## II. RELATED WORK

In [2], a framework is described to construct a generic course of the road for a virtual driving scenario using a stochastic approach. It combines Markov Chain and Markov Chain Monte Carlo methods to test different input combinations. By using this automation, there would be the possibility that parameter sets, which were forgotten or erroneously ignored during the manually test creation, are tested.

A test generator is presented in [3][4], which creates, executes and evaluates test scenarios automatically. The algorithm behind the generator tries on the one hand to maximize the coverage of the reached system states by changing the input of the SUT during the test execution. On the other hand, it searches for system states that do not fulfill the given evaluation criteria. It is left to the test engineer to configure the test generator in such a way that no invalid test scenario is created and the evaluation criteria are valid for all generated test scenarios.

According to [5][6], the formal verification is currently the only known way to ensure that a system works as specified. This means that the implementation strictly follows the specification and thus it is possible to determine its behavior in every situation. To perform a formal verification, the specification must meet some requirements. Among others, the specification must be complete and correct. This is a big challenge, especially in large projects with many dependencies to external components from different suppliers.

 ISBN: 978-1-61208-500-5 

A comparative study on methods to automatically determine the expected response of the SUT is given in [7]. Such methods are necessary for the automatic testing. Otherwise, the response has to be verified manually. The presented approaches are mostly limited to their application field and cannot be generally applied.

## III. System Under Test

The SUT, which is also named as "test object", is a physical or logical unit as illustrated in Fig. 1 that is tested for correctness against the specification. It has an input interface $X$ and an output interface $Y$. A stimulus $x \in X$ at the input causes a response $y \in Y$ at the output, as can be seen in (1), where the stimulus $x$ can change over time.

$$x(t) \xrightarrow{\tau} y(t) \tag{1}$$

For the reproducibility of the test results, a defined start state of the SUT is required at the beginning of the test execution. Given that, it is possible to obtain an identical response when repeating a test case or after changing the execution order of test cases in a test run. For this purpose, however, the SUT must meet the following properties:

a) time-invariant
b) memoryless

According to [8], a system is called time-invariant, if a delay of the input causes the same delay at the output as shown in (2), and memoryless, if the response does only depend on the current input and not on an input from the past. If both properties are satisfied by the SUT, exactly one $y \in Y$ can be associated for each $x \in X$ regardless of the point in time and the sequence of the stimulation. Other systems that do not meet these properties can show different responses to the same stimulus.

$$x(t) \xrightarrow{\tau} y(t) \implies x(t+\delta) \xrightarrow{\tau} y(t+\delta) \tag{2}$$

Decisions in autonomous driving are dependent on the environment and usually on the history of events, which means that different stimulation sequences over time are needed for the testing. A static input pattern or a small number of scenarios are not sufficient to verify a DAS.

### A. Stimulation

The input interface of the SUT consists of signals providing data from other Electronical Control Units (ECUs). It represents the lowest level of stimulation and can be stimulated at a Model-in-the-Loop (MIL) or Software-in-the-Loop (SIL) test bench. At a Hardware-in-the-Loop (HIL) test bench, a bus interface and a Residual Bus Simulation (RBS), which could have effects to the time behavior of the stimulation as shown in [9], are required.

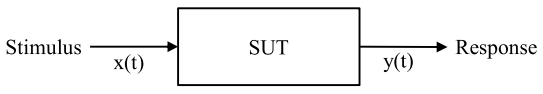


Figure 1. Schematic representation of the SUT

A direct access to the input interface on the signal-level allows a precise stimulation of the SUT. The large fan-in leads to an exponential number of test cases, which can be generated. State-of-the-art DASs have hundreds of input signals, which have to be consistently stimulated with the correct values over time. Many input signals that are not in the scope of the current test case, but are necessary for the proper operation of the SUT, must be correct and should not be manipulated by the test case generator. Deviations from the correct sequence are usually detected by the SUT and leads to a functional degradation to bring the SUT into a safe state. This outcome must be considered at the evaluation of the test.

To cope with the complexity of the input interface, it is a common practice to use models to abstract the input signals and thus to simplify the stimulation of the SUT. The models are acting as an intermediate layer between the signal-level and the used level of stimulation and ensure that the stimulation is performed in a consistent way. The advantage of the simplification is achieved by losing direct control of the input interface, which could complicate the stimulation, if specific signal characteristics are needed.

### B. Evaluation

Depending on the stimulation of the SUT a response can be observed at the output interface, which has to be checked for its correctness. The evaluation of the signal characteristics can be done at selected points in time or over a certain period of time. Within these time intervals, the observed response of the SUT is compared against the expected response to ensure that the behavior corresponds with the specification. Thereby, the specification is a single point of failure. If the specification is not reliable, the tests do not recognize an abnormal behavior of the SUT in specific situations.

## IV. Automatic Testing

As explained in the previous section, it is not sufficient to test only static input pattern or a small number of scenarios to verify the functionality and the safety of a DAS. Rather, it is required to test a broad range of different situations as they can be found in real-world environments. The variety of environmental conditions makes it difficult to verify the DAS within its operating range and to ensure that a safe state is always reached. For this reason, automatic testing in addition to the already performed testing is thought to play an important role.

### A. Setup

Fig. 2 shows the setup for the automatic testing as used by the presented approach. The *Test Generator* comprises two parts. In the first part, the *Scenario Generator* composes a *Scenario* and the corresponding *Stimulus* for the *Test Bench* that is responsible for the test execution. In the second part, the *Response Determination* determines the *Expected Response* based on the generated *Scenario*. The *Test Bench* applies the *Stimulus* created by the *Scenario Generator* to the SUT, while observing the *Response*. The *Evaluator* compares the observed *Response* of the SUT with the *Expected Response* provided by the *Response Determination*. Differences outside a specified tolerance value cause the *Result* to fail as described in [10].

 ISBN: 978-1-61208-500-5 

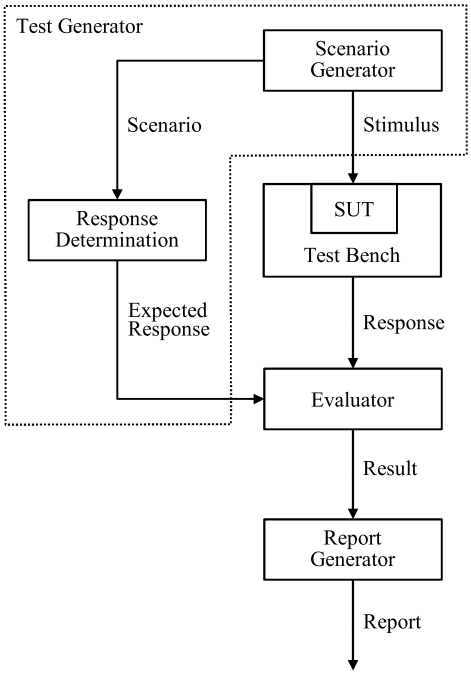


Figure 2. Setup for the automatic testing

The *Report Generator* processes the raw data provided by the *Evaluator* and creates a detailed *Report*, which allows a person with appropriate domain knowledge to analyze failed test cases and to verify successfully executed test cases. To speed-up the analysis, relevant signal characteristics are diagrammed and, where required, derived values and signals are calculated in advance.

### B. Need for an Automatic Evaluation

Tools for generating scenarios usually do not provide the expected responses of the SUT, which constitutes the basis for an automatic evaluation. They leave it to the tester to define them. Without an automatic evaluation the generated scenarios can be executed at a test bench, but the actual behavior of the SUT cannot be automatically evaluated. This means that each scenario must be analyzed manually by an expert before the first execution. In this manner the evaluation can be done for individual scenarios, but this is not feasible in practice for the expected large number of generated scenarios needed for the testing.

## V. Evaluation Based on Abstraction Level Constraints

In order to benefit from the advantages of automatic testing, a constraint based approach is presented to determine the response of the SUT working on different levels of abstraction. It uses an automatic analysis of the observed behavior to classify individual situations of the scenarios on the system-level after the test execution. Thereby the approach recognizes an unusual behavior of the SUT initially on the system-level from the viewpoint of an outside observer and is subsequently going down to lower levels.

For being able to implement the determination of the response on the signal-level, a profound expert knowledge is necessary to determine whether an output signal provides a correct value, or not. Dependencies between signals complicate the evaluation in addition. On the input interface, it is a common practice to use models to abstract the large number of input signals. The same is done by the approach on the output interface. Through the use of models the abstraction at the output is increased to a higher level. As a result of this, less knowledge about the functionality of the DAS is necessary to evaluate the response. However, missing parts of the overall system have to be simulated due to the increasing of the abstraction. The higher the level of the abstraction, the more parts are missing. For the evaluation of a DAS on the system level, at least a physical model of the system vehicle and simulations of all other involved ECUs are necessary.

The approach uses a parameterized model for the evaluation on the system-level, which spans a safety area around the road objects. As shown in Fig. 3, the safety area is thereby defined by the variables $d_1$, $d_2$, $d_3$ and $d_4$ that represents the safe distance in each direction. These variables can be changed over the time and thus dynamically adapted to the current situation. The safety area can be, e.g., increased in specific directions depending on the vehicle velocity.

### A. Classification

For the evaluation, the observed behavior of the DAS is analyzed after the test execution to classify individual situations of the performed scenario according to the following four classes.

The class “Fallback” is a specified and thus explicitly allowed state of a DAS. The state is not necessarily critical, but rather it indicates a situation that cannot be handled by the system. As a precaution, the DAS returns the vehicle control to the driver at the expense of its availability. In relation to highly automated driving, these situations still reveal functional restrictions of the system.

*Definition 1 (Fallback):* The fallback is a state of the DAS, which is entered if the system cannot handle the situation by its own and returns the vehicle control to the driver.

A “Hazardous Situation” is a critical situation without a collision that is either caused by the DAS itself or by at least one object included in the scenario. On the one hand, an object vehicle can cause such a situation, e.g., during a lane change if the scenario generator has ignored the safety distance and thus the object gets too close to the system vehicle. On the other hand, the DAS can cause the critical situation, e.g.,

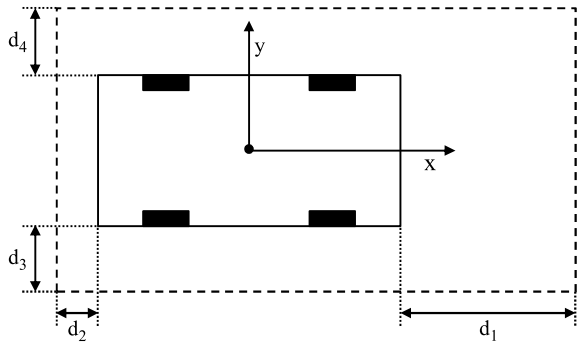


Figure 3. Safety area spanned around a vehicle

by following too closely on an object vehicle. From when a situation is considered as hazardous heavily depends on the conditions given by the scenario. The change of only one condition can lead to a new driving scenario with a different hazard potential.

*Definition 2 (Hazardous Situation):* A hazardous situation occurs, when:

a) the system vehicle leaves the lane without cause.
b) the minimum distance between the system vehicle and an object is less than a specified value.

The class "Event of Damage" means that the DAS was not able to avoid a collision. Further investigations are required to find out, whether there is any misconduct of the system, or not. The event of damage is usually preceded by a hazardous situation.

*Definition 3 (Event of Damage):* An event of damage occurs, when:

a) the system vehicle leaves the road.
b) a collision between the system vehicle and one or more objects cannot be avoided by the DAS.

All other situations are classified as "Unsuspicious", which is used as the default class.

*Definition 4 (Unsuspicious):* A situation is unsuspicious, if an event of damage or a hazardous situation does not exist, as well as, the DAS is not in the fallback state.

In general, the operating mode of the DAS during a situation is crucial for the evaluation. If the DAS is in an emergency situation, its behavior is different from the behavior in the driving mode. While a collision in the driving mode is not acceptable, an unavoidable collision that was mitigated in an emergency situation might be tolerable.

### B. Determination of the Expected Response

The determination of the expected response works on a knowledge base individually created for each DAS on the basis of the available specifications. In the knowledge base, a hierarchical structure ensures that the behavior of the SUT is stored dependent on its abstraction level. At the beginning only the behavior described on a high level is used to fill the knowledge base. Already after a short time, a state is thus reached, which allows the determination of the expected response of the DAS. This turns out to be sufficient to execute the first test cases and to get a test result, which guides the system developer to refine the behavior on lower levels and to establish relations between different abstraction levels.

In addition, the use of constraints allows a selective description of the behavior. Based on the stimulus, a distinction can be made at each abstraction level to describe deviating responses of the SUT. In this way, different situations can be handled.

## VI. Case Study

In this section, two examples, which were done as a case study, are discussed to show the idea behind the approach. All scenarios of both examples are executed at a SIL test bench. The first example demonstrates a passing maneuver that is analyzed using dynamically expandable safety areas around the objects. It describes, how individual situations of a scenario are classified corresponding to the Section V-A. Following this, a second example is presented that uses an algorithm based on [11] as a SUT to provide the functionality of a CMS. It explains the determination of the expected response on the system-level and discusses the results with respect to the performed scenarios.

### A. First Example: Classification of a Scenario

The safe distance between objects in the road traffic heavily depends on a variety of factors, such as the vehicle velocity or the weather, and cannot be specified by generally valid values. Even in the law, usually no specific values are specified, but a sufficient safe distance is stipulated, e.g., in the German Road Traffic Act [12], to avoid hazardous situations or collisions with other road users. The distances considered as safe vary with the velocity, the driving direction or environmental conditions.

As the basis for the example, the model of the safety area used for classification was parameterized according to the two-second rule [13][14] (in some states also known as three-second rule), which states that a driver of a vehicle should stay at least two seconds behind the vehicle in front. During the test execution the safety area are dynamically expanded in the driving direction based on the vehicle velocity with a lower saturation at one meter. The other parameters of the model, i.e., the safe distance to the left and to the right, as well as, the safe distance to the rear, are set to a constant value of one meter.

The scenario used for the example represents a passing maneuver at high speed, as illustrated on the left side of Fig. 4, in which an object vehicle passes the system vehicle on the left lane. The timing of the passing maneuver has been chosen such that it is hazardous by violating the safety distance but not damaging.

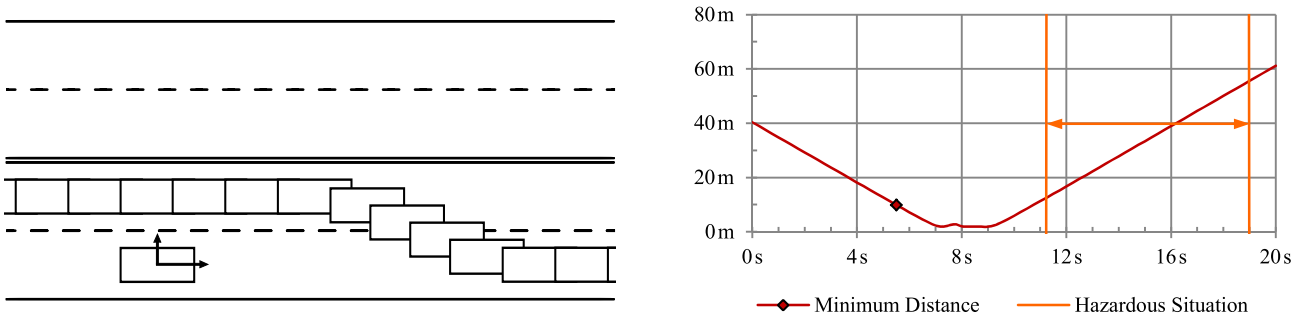


Figure 4. Passing maneuver illustrated as a difference representation (left side) and the corresponding test result (right side)

 ISBN: 978-1-61208-500-5 

The test results diagrammed on the right side of Fig. 4 show the detection of a hazardous situation. Thereby, it is striking that the situation has been classified as hazardous after the minimum distance between the system vehicle and the object vehicle had already reached its minimum value. The minimum value is achieved, when both vehicles are at the same level. At this point in time, the distance between the left side of the system vehicle and the right side of the object vehicle are considered by the model only. The first instance of violating the safe distance can be seen during the lane change. At this, the safe distance of the system vehicle is violated by the object vehicle and the hazardous situation is recognized by the model based on the two-second rule.

The example shows that the obtained sequence of classified situations describes the scenario on the system-level, which can be used in the next step to evaluate the behavior of the SUT. It is thus not necessary to directly cope with signals.

### *B. Second Example: Behavior Analysis in Different Environmental Conditions*

The Collision Mitigation System (CMS) [15], which can be found nowadays in almost all new vehicles, monitors the traffic around the vehicle and warns the driver of potential collisions in hazardous situations. If the driver does not react to the warning, an automatic braking is performed. The success of the system, whether a collision can be avoided or at least the effects of a collision can be reduced, is highly determined by the environmental conditions.

On the one hand, the automotive manufacturer has to ensure that no unnecessary triggering of an automatic braking is performed by the CMS, which can cause a threat on the road. On the other hand, the system should provide an added value to the driver in as many situations as possible. As elucidated in [16], there is a trade-off for the automobile manufacturers between safeness and availability.

In contrast to other DASs, the driver sees the CMS only in action in hazardous situations or at collisions. The same applies for testers, which have to put themselves in danger for the testing of the system. Although there is the opportunity to simulate virtual objects for the system vehicle [17] and thus to reduce the risk, the number of tests that can be performed is limited and in no relation to the possible scenarios. Through automatic testing, a variety of different scenarios can be executed on test benches. Thereby, the presented approach provides the expected responses of the DAS for the generated scenarios on different abstraction levels, which can be compared with the observed response obtained from the test bench.

The determination of the expected response is based on the specification of the CMS. Only the behavior described on a high level is used in the following to get a test result within a short time. In further work, the determination can be extended to support additional abstraction levels down to the signal-level.

Two characteristic scenarios for the CMS are presented in the following:

*1) Reaching the Tail End of a Traffic Jam at Low Speed without the Driver Applying the Brake:* Based on the scenario a hazardous situation is determined, where it comes to a brake intervention by the CMS. Through the intervention, the system vehicle should be slowed down to standstill, before there can be a front-end collision with the object vehicle ahead. As shown by the test results on the left side of Fig. 5, the minimal distance between the system vehicle and the object vehicle decreases over the time. A hazardous situation is recognized, but there is, as expected, no collision. Shortly before standstill, the situation is no longer evaluated to be hazardous due to the automatic braking.

*2) Reaching the Tail End of a Traffic Jam at High Speed without the Driver Applying the Brake:* In contrast to the previous scenario, the system vehicle has a much higher velocity in this situation than before. Due to the increased velocity, a brake intervention with subsequent collision is determined. The test results, as diagrammed on the right side of Fig. 5, shows that the minimal distance between both vehicles rapidly decreases. Also a hazardous situation is recognized, but this time there is an event of damage caused by the collision of the system vehicle and the object vehicle. After the collision, no further information was provided by the test bench.

The example shows that the determination of the expected response on the system-level can be used for an automatically evaluation of driving scenarios within unknown environmental situations. The abstraction leads to a simplification of the evaluation.

## VII. Conclusion and Future Work

Automatic testing, which can be used within traffic simulations, would have the potential for analyzing the response of a DAS based on a large number of different scenarios with reasonable effort. However, an appropriated determination of the expected response and a convincing approach for an evaluation are mostly missing nowadays. Due to the expected large number of generated test cases for the automatic testing and the time-consuming definition of the expected responses, the

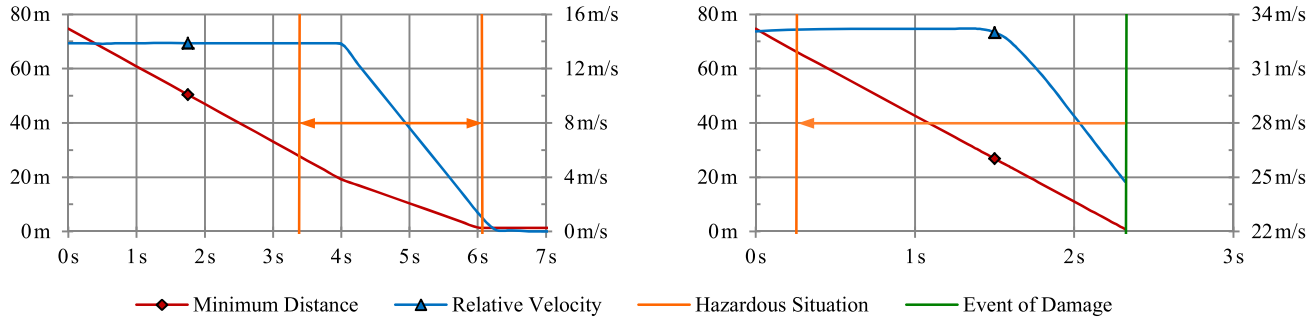


Figure 5. Test result of the slow maneuver (left side) and the test result of the fast maneuver (right side)

determination and the evaluation should be done automatically by the test generator and not manually performed by experts with appropriate domain knowledge about the DAS and its included functionality.

It has been shown that the complexity of the determination and the evaluation, which arises due to the number of signals in the output interface of the SUT, is manageable through the use of models. The higher the level of abstraction at the output interface is chosen, the less domain knowledge is necessary for the evaluation. The abstraction and the resulting simplification cause that not all information from the signal-level are available at each level of abstraction. By increasing the level of abstraction, parts of the overall system must be simulated. The closer the simulation to reality, the more reliable the results obtained. However, for software-driven testing issues a sufficient imitation of the real-world system is supposed to be precise enough. A simulation that considers all eventualities is usually not necessary.

Furthermore, it has been shown that the behavior of a SUT can be determined after the test execution by classifying the response observed at the test bench. Thereby, the obtained sequence of classified situations describes the driving scenario on the system-level. This sequence can be automatically evaluated based on the determined response und used to find errors or missing parts in the specification.

It is left for future work to apply the current approach to a DAS with several assistance functions. One aspect might be to analyze, how many abstraction levels are required to model the behavior of the SUT and to examine at which abstraction level constraints are necessary to correctly determine the expected response based on the stimulus. Another aspect might be to optimize the scenario generator to increasing the search space coverage with a minimum number of additional test cases.

## REFERENCES


[1] A. Zanten and F. Kost, Handbook of Driver Assistance Systems: Basic Information, Components and Systems for Active Safety and Comfort. Cham: Springer International Publishing, 2016, ch. Brake-Based Assistance Functions, pp. 919–967. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-12352-3_40

[2] S. Prialé Olivares, N. Rebernik, A. Eichberger, and E. Stadlober, Advanced Microsystems for Automotive Applications 2015: Smart Systems for Green and Automated Driving. Cham: Springer International Publishing, 2016, ch. Virtual Stochastic Testing of Advanced Driver Assistance Systems, pp. 25–35. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-20855-8_3

[3] A. Rink, E. Chrisofakis, and M. Tatar, "Automating test of control software," ATZelektronik worldwide, vol. 4, no. 6, pp. 24–27, 2009. [Online]. Available: http://dx.doi.org/10.1007/BF03242245

[4] M. Tatar, "Test and validation of advanced driver assistance systems automated search for critical scenarios," ATZelektronik worldwide, vol. 11, no. 1, pp. 54–57, 2016. [Online]. Available: http://dx.doi.org/10.1007/s38314-015-0574-1

[5] G. Klein, J. Andronick, K. Elphinstone, T. Murray, T. Sewell, R. Kolanski, and G. Heiser, "Comprehensive formal verification of an os microkernel," ACM Trans. Comput. Syst., vol. 32, no. 1, pp. 2:1–2:70, Feb. 2014. [Online]. Available: http://doi.acm.org/10.1145/2560537

[6] G. Klein, K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood, "sel4: Formal verification of an os kernel," in Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles, ser. SOSP '09. New York, NY, USA: ACM, 2009, pp. 207–220. [Online]. Available: http://doi.acm.org/10.1145/1629575.1629596

[7] S. R. Shahamiri, W. M. N. W. Kadir, and S. Z. Mohd-Hashim, "A comparative study on automated software test oracle methods," in Software Engineering Advances, 2009. ICSEA '09. Fourth International Conference on, Sept 2009, pp. 140–145.

[8] M. D. Adams, Continuous-Time Signals and Systems. Victoria: University of Victoria, 2013, ch. Continuous-Time Signals and Systems, pp. 7–44, ISBN: 978-1-55058-495-0.

[9] D. Ulmer, S. Wittel, K. Hünlich, and W. Rosenstiel, "Testing Platform for Hardware-in-the-Loop and In-Vehicle Testing Based on a Common Off-The-Shelf Non-Real-Time PC," International Journal on Advances in Systems and Measurements, vol. 4, no. 3 & 4, pp. 182–191, 2011, ISSN: 1942-261x. [Online]. Available: http://www.iariajournals.org/systems_and_measurements/

[10] K. Hünlich, D. Ulmer, S. Wittel, and U. Bröckl, "Optimized Testing Process in Vehicles Using an Augmented Data Logger," International Journal on Advances in Systems and Measurements, vol. 6, no. 1 & 2, pp. 72–81, 2013, ISSN: 1942-261x. [Online]. Available: http://www.iariajournals.org/systems_and_measurements/

[11] H. Winner, Fundamentals of Collision Protection Systems. Cham: Springer International Publishing, 2016, pp. 1149–1176. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-12352-3_47

[12] H. Janker, Road Traffic Law - Text Edition ("Strassenverkehrsrecht - Textausgabe"), 53rd ed. Stuttgart: Dt. Taschenbuch-Verlag, 2015.

[13] Road Safety Authority Ireland, "Driving safely in traffic - the two second rule," 2016, URL: http://www.rotr.ie/rules-for-driving/speed-limits/speed-limits_2-second-rule.html [retrieved: July, 2016].

[14] New York State Department of Motor Vehicles, "Driver's Manual & Practice Tests - Chapter 8: Defensive Driving," 2016, URL: https://dmv.ny.gov/about-dmv/chapter-8-defensive-driving/ [retrieved: July, 2016].

[15] L. Walchshäusl, R. Lindl, K. Vogel, and T. Tatschke, Advanced Microsystems for Automotive Applications 2006. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, ch. Detection of Road Users in Fused Sensor Data Streams for Collision Mitigation, pp. 53–65. [Online]. Available: http://dx.doi.org/10.1007/3-540-33410-6_6

[16] A. Pruckner, R. Stroph, and P. Pfeffer, Handbook of Intelligent Vehicles. London: Springer London, 2012, ch. Drive-By-Wire, pp. 235–282. [Online]. Available: http://dx.doi.org/10.1007/978-0-85729-085-4_11

[17] Steinbeis-Stiftung für Wirtschaftsförderung (StW), "Virtual Testing of Reality," 2015, URL: http://www.steinbeis.de/en/publications/transfer-magazine/edition-042015/virtual-testing-of-reality.html [retrieved: July, 2016].


 ISBN: 978-1-61208-500-5 