Virtual Test Scenarios for ADAS: Distance to Real Scenarios Matters!

Mohamed El Mostadi, Hélène Waeselynck, Jean-Marc Gabriel

Virtual Test Scenarios for ADAS: Distance to Real Scenarios Matters!

Mohamed El Mostadi[1,2], Hélène Waeselynck[1], and Jean-Marc Gabriel[2]

Abstract— **Testing in virtual road environments is a widespread approach to validate advanced driver assistance systems (ADAS). A number of automated strategies have been proposed to explore dangerous scenarios, like search-based strategies guided by fitness functions. However, such strategies are likely to produce many uninteresting scenarios, representing so extreme driving situations that fatal accidents are unavoidable irrespective of the action of the ADAS. We propose leveraging datasets from real drives to better align the virtual scenarios to reasonable ones. The alignment is based on a simple distance metric that relates the virtual scenario parameters to the real data. We demonstrate the use of this metric for testing an autonomous emergency braking (AEB) system, taking the highD dataset as a reference for normal situations. We show how search-based testing quickly converges toward very distant scenarios that do not bring much insight into the AEB performance. We then provide an example of a distance-aware strategy that searches for less extreme scenarios that the AEB cannot overcome.**

I. Introduction

Advanced driver assistance systems (ADAS) and autonomous driving functions require a thorough validation. Their failures may cause accidents, sometimes fatal [1]. Real-world testing by driving on public roads would need a huge amount of time, effort and cost in order to demonstrate that the safety goals are met [2]. Virtual validation seems a promising complementary solution, especially with the emergence of simulation platforms dedicated to driving environments. Thus, several works in progress seek to define innovative testing methods, exploiting the simulation and virtualization facilities offered. They can be broadly classified into two main categories: methods centered on the replay of scenarios encountered during real drives, and methods based on parameterized environment models, making it possible to explore a wider space of test scenarios.

In the first category, the collected data are processed to recreate the corresponding scenarios in simulation [3]. For example, the geometry of a road section is reconstructed from the data [4], or by superimposing location data on digital road network maps [3][5]. In [6], the driving data are used to train neural networks to reproduce trajectories similar to human ones. These methods are limited to the scenarios in the database and do not allow a diversity of scenarios, especially the most critical ones. Other work has sought to obtain more diversification by creating new scenarios, corresponding to variants of the real scenarios [7]. These variants are obtained by applying transformations such as trajectory translations. However, the types of diversification remain limited.

The second category of work offers more flexibility in creating diverse scenarios. It involves defining a model of the virtual environments to be considered for testing, based on domain expertise. For example, the space of possibilities will be identified in terms of road segment parameters (number of lanes, with or without crossroads, curved segment, slope, etc.), mobile objects in the environment (type of car or pedestrian, departure coordinates, travel parameters), visibility conditions (brightness, fog or good weather), etc. The modelling can take advantage of the ontologies defined for ADAS, which provide a structured view of the concepts of situation, scene, scenario, and the elements that populate them [8][9][10]. Once the generation parameters have been identified, procedures are developed to automatically produce the content of the tests according to the values of these parameters. As in the case of data from real drives, unguided generation is likely to produce a large volume of redundant tests. Some authors have proposed either to *a priori* guide the generation by test criteria, for example, by considering the coverage of combinations of model parameter values [11][12], or to create a large base of virtual tests and then to *a posteriori* extract subsets by filtering and classification processes [13]. Another approach consists in implementing metaheuristic processes to optimize scenario hazard criteria [14][15][16][17][18][19]. The generation is then iterative, seeking to increase the dangerousness of the virtual tests created by successive trials. In the literature, this approach has been the most studied to find collision scenarios. It will also be the focus of this paper.

While the metaheuristic search can be very effective at finding collisions, we experienced that many of the found cases fail to provide any useful insight into the tested ADAS. They correspond to so extreme driving situations that an accident is simply unavoidable. Facing the same problem, some authors have recently worked on the search-based generation of avoidable collision scenarios [18]. They consider systems having a weighted decision function. A collision is then avoidable if a change in the weights is found, which makes the system able to avoid the collision. In this paper, we propose a different solution that does not rely on the internals of the system. It is based on the premise that the search produces too extreme scenarios because it lacks an alignment with reasonable driving situations. For this, we propose leveraging the datasets recorded from real drives, which provide a reference

[1]LAAS-CNRS, University of Toulouse, Toulouse, France. Contacts: mohamed.el-mostadi@laas.fr, Helene.Waeselynck@laas.fr.

[2]Renault Software Labs, Toulouse, France. Contacts: mohamed.m.el-mostadi@renault.com, jean-marc.gabriel@renault.com.

for normal situations. We then introduce a simple distance metric measuring how far the virtual tests are from those normal situations. The test alignment approach is illustrated on an Autonomous Emergency Braking (AEB) system, taking the HighD dataset [20] as the reference. We first revisit the results of a previous implementation of the search that produced numerous useless scenarios. We show how the distance metric is helpful to visualize the quick convergence of the search toward very extreme scenarios. We then revise the search to make it distance-aware and demonstrate the production of stressful scenarios that are closer to the real ones.

The structure of the paper is as follows. Section II presents the motivating example of the AEB. Section III presents the data extraction from the HighD dataset, the mapping of real data to test parameters, and the proposed distance metric. Section IV analyzes the distance of the scenarios generated without consideration for the real data. Section V presents a new generation strategy that includes the distance to real scenarios as an optimization criterion. Section VI concludes.

II. A Motivating Example

To illustrate the issue of testing against useless scenarios, we report on the example of an autonomous emergency braking (AEB) system. The AEB under test is an R&D Simulink model that we used for experimental purposes. We performed the model-in-the-loop tests on a simulator based on Unity [21]. The simulator allows us to create roads and target vehicles in the vicinity of the ego vehicle under test. The physical ego car is represented by a sensor module (a lidar) and an actuator that approximates the braking profile of a real Renault car on dry roads. The simulator API provides predefined test actions to control the behavior of the ego and target vehicles, like actions to set their position and speed, or a lane change action. They facilitate the implementation of classical AEB scenarios (car following, cut-in, cut-out). Here, we report on experiments involving cut-in scenarios.

A. Virtual Cut-in Scenario

In a cut-in scenario, a target car merges into the lane just in front of the ego. The scenario is parametrized so that numerous cut-in instances can be tested. A test case is then a vector of parameter values, yielding a cut-in instance.

We chose the following parameters:

- RelPos ∈ [10.0, 100.0]: initial longitudinal position of the target relative to the ego, in meters. The definition domain only has positive values; hence the target is always ahead of the ego when it starts the lane change.
- Ve ∈ [60.0, 160.0]: speed of the ego car in km/h. It will remain constant during the scenario unless the AEB applies the brakes.
- Vt ∈ [60.0, 160.0]: speed of the target in km/h. It will remain constant during the scenario.
- T ∈ [1.0, 7.0]: duration of the lane change in seconds. The test action to perform the change computes the lateral shift based on a cosine function, tuned by the desired duration time T. The shorter the time, the sharper the lane change.

Weather parameters are not considered because they are currently not covered by the simulator (it simulates the effect of bad weather on camera-based vision, but not on the lidar-based vision of the AEB).

The AEB under test is intended for rear-end collisions while driving in highways and trunk roads. It may apply the brakes if the driver does not react to an imminent forward collision. The aim is to prevent or at least mitigate the impact. If the impact occurs at a relative speed of less than 30 km/h, the mitigation is considered a success. In the rest of the paper, an impact at more than 30 km/h will be called a *critical* collision.

B. Search-Based Testing

We used an evolutionary test approach to search for cut-in cases where the AEB would not succeed in avoiding critical collisions. The approach is based on the Non-dominated Sorting Genetic Algorithm Version 2 (NSGA-II), a multi-objective search optimization algorithm [22]. NSGA-II follows the general steps of a genetic algorithm: it starts with an initial population of candidate test cases and evolves it toward better test cases during a number of iterations. The evolution of the population uses a selection of the fittest test cases, followed by breeding and mutation to produce new cases. Being multi-objective, NSGA-II may accommodate several possibly conflicting fitness functions at the same time. At the end, it returns a set of test cases that form a Pareto front, representing the best trade-offs found for the fitness functions.

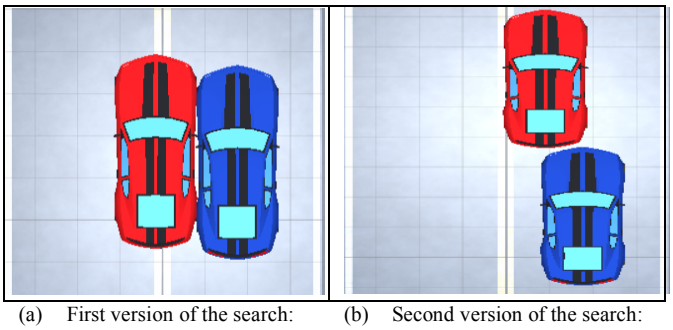
The search for critical cut-in instances was based on the NSGA-II implementation in the platypus library [23]. We defined two fitness functions: we want to minimize the Time-To-Collision (TTC) and at the same time maximize the relative speed of the vehicles. The fitness evaluation requires running each candidate test case and reporting the observed TTC at every simulation step as well as the final speed of the ego. To speed-up the search, we only run test cases that will yield a collision unless the AEB applies the brakes. That is, we run test cases with inputs Vt < Ve and assign the others predefined penalizing fitness values. A test execution is stopped whenever a collision occurs (TTC = 0), or the AEB has reduced the speed of the ego below the one of the target. The fitness functions are then the following:

- F1 (to be minimized) = if Vt < Ve return (min observed TTC) else return (100.0).
- F2 (to be maximized) = if Vt < Ve return (Ve_final – Vt) else return (-100.0).

C. Test Results

We ran the search with a population size of 100 test cases and performed 10 iterations. A complete run of the search takes approximately 8 hours with an Intel Core I7-6600U CPU performing at 2.6 GHz.

The search was successful since it returned a Pareto front of 100 test cases, all with critical collisions. However, all these cases correspond to the extreme scenario shown in Fig. 1.a: the target vehicle suddenly veers sideways into the ego, crashing into it. There would be no way for the AEB (or a human driver) to avoid the side collision. The test cases are thus irrelevant to challenge the AEB logic.



(a) First version of the search: side collisions

(b) Second version of the search: extremely aggressive lane changes

Figure 1. Critical cut-ins found by NSGA-II (blue car = the ego, red car = the target)

We then modified the fitness functions to guide the search towards rear-end collisions. F1 and F2 receive penalizing values (resp. 100.0 and -100.0) if, at the end of the test case, the target vehicle is not in front of the ego. We reran the NSGA-II search with the improved fitness. Again, the search returned a Pareto front of 100 critical test cases. All of them correspond to extremely aggressive cut-ins. The lane change is very sharp and puts the target in a position so close (a few centimeters) to the ego that the cars are almost touching each other (see Fig. 1.b). The impact is unavoidable and occurs with a high relative speed (typically, the ego is initially driving at nearly 160 km/h while the target is at 60 km/h). In several cases, the AEB even has no time to start braking. Such test cases are too extreme to be worth consideration.

The search produced numerous critical test cases – not just the ones retained in the Pareto front. The initial population of 100 random cases contains 2 critical ones (2%). At Iteration 5, the rate is already about 60%. It exceeds 80% at all subsequent iterations. All in all, 512 tests out of 1,000 (100 tests x 10 iterations) yield a critical collision. We did not manually analyze all of them but suspect that many are just variants of the ones in the Pareto front.

D. Discussion

This experience illustrates how a randomized search for dangerous scenarios may leave the tester with numerous cases to analyze, a significant proportion of which may actually prove useless. They represent so extreme driving situations that they do not bring much insight into the adequacy of the ADAS function under test. Their analysis is a waste of effort. Also, the test execution resources are poorly utilized, as the search is trapped into extreme test cases and produces variants of these cases at higher and higher rates.

It would be desirable to integrate a concept of "normality" into the test process, where the extreme nature of tests cases would be determined based on how "far" they are from normal driving situations. We propose an approach that captures the characteristics of reasonable scenarios from a dataset of real vehicle trajectories. A simple distance metric allows us to compare the virtual test cases to the real cut-ins. We demonstrate the usefulness of the metric to revisit the results of the search, visualize the convergence toward extreme cases and adapt the search to produce critical test cases better aligned to real situations.

III. Leveraging Datasets as a Reference for Normal Situations

For a proof-of-concept demonstration, we consider the HighD dataset as a reference for normal situations. HighD [20] is a German dataset gathering data from 6 German highways. It contains 110,000 vehicle trajectories which represents a recording of 16.5 hours. The recordings are made by a drone and cover a segment of 420 m.

We present below the extraction of cut-ins from the recorded data, the mapping of real data elements to virtual test parameters, and finally introduce a distance metric that characterizes the extremal degree of the test cases compared to naturalistic cut-ins.

A. Cut-in Extraction

A lane change is not necessarily a cut-in. Here, we are interested in the extraction of a subset of lane changes, those where the vehicle inserts in front of another. The cut-in detection is done according to the following criteria:

- We keep only the scenarios with a complete lane change.
- The relative position between the two vehicles must be less than 100 m.
- The vehicle in the destination lane (i.e., the blue vehicle in Fig. 2) must have a higher speed than the one changing lane.
- The vehicle in the destination lane must be the same during the entire lane change.

We applied a filter on the selected cut-in subset to detect the presence of anomalies. We verify that the evolution of the positions is consistent with the speed. We also eliminate the scenarios where the recording is partial. At the end, we have 1100 cut-in scenarios that meet our criteria. The collected data are time series describing the scenario between Positions 1 and 2 (see Fig.2). Position 1 is when the red vehicle is still in its departure lane but touches the separation line. Position 2 is when the vehicle is in the new lane and still touches the line.

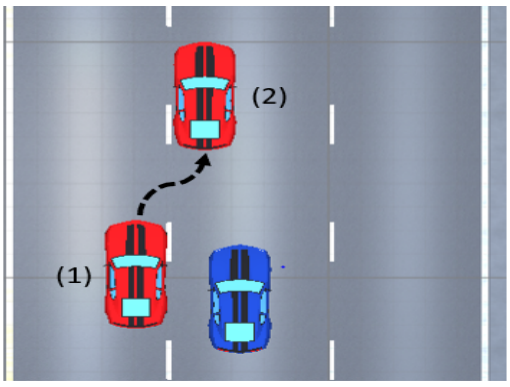


Figure 2. Positions 1 and 2 in cut-in scenarios

B. Mapping Real Data to Test Parameters

The collected data do not exactly correspond to the parameters of the virtual scenarios. It is therefore necessary to create a mapping between the two.

In the test scenarios, the speed of the vehicles is constant. While in real life, there are speed variations. We choose to take the speed captured in Position 1 (see Fig.2). For the vehicle that has the role of the target (i.e., the one making the cut-in), we could verify that the speed varies little between Positions 1 and 2. For the vehicle corresponding to the ego (i.e., the one undergoing the cut-in), there are braking actions. The constant speed in the virtual tests does not reproduce these actions but this is not an issue: we want to test the AEB in situations where the driver should brake but does not.

Another difference comes from capturing the partial duration of a lane change between Positions 1 and 2 (for real scenarios), rather than the total duration starting and ending at the lane centers (for virtual tests). In real life, vehicles never drive perfectly aligned with the center of the lane, so that it is difficult to pinpoint the precise start or end of the maneuver. We tried several detection thresholds on the lateral position and lateral speed but could not find a satisfactory setting. There were always poorly detected cases, e.g., a start detected well before the actual lane change, or an undetected end. We considered it safer to measure the time between two well-defined positions. Moreover, they correspond to a key fragment of the scenario: it is during this time interval that the ego detects the cut-in. We can then infer the total time of the virtual lane change, so that the time spent between Positions 1 and 2 exactly corresponds to what is measured in real life. Similarly, we calculate the initial relative position of the ego and the target such that, when Position 1 is reached in a virtual test case, the distance matches the one in the real situation.

B. Distance Metric

For the distance calculation, we compare the virtual test parameters to their equivalent values calculated for the real data. A test case corresponds to a vector of n test parameters (in the AEB example, n = 4). The real data, after mapping, give a set of m vectors of size n (for the AEB, we have a vector for each cut-in extracted from the database, where m = 1,100). For each parameter p_i, we define a $step_i$ which corresponds to a fraction of the interval of values of p_i from the dataset. Here we take 5%, yielding $step_{RelPos}$ = 4.89 m, $step_{Ve}$ = 4.85 km/h, $step_{Vt}$ = 3 km/h and $step_T$ = 0.15 s. The step normalizes the distance along the dimensions of the vector.

Let TC = <p_1, ..., p_n> be a test case, and let R be the set of vectors extracted from the dataset, taken as a reference for reasonable driving situations. We compute the distance from TC to R as follow:

$$\text{Distance (TC, R)} = \min_{j=1,..,m} \sum_{i=1}^{n} \frac{|p_i - R_{ij}|}{step_i}$$

The distance between a test case and a real cut-in is measured by the Manhattan distance, and we take the minimum for all cut-ins extracted from the dataset. Thus, each virtual scenario is compared to all real scenarios to identify the closest one. This approach was chosen because the total number of cut-ins is small enough to allow it. If there were a large number of cut-ins, we could for example group them into clusters, and compare TC to each cluster.

This metric will be used to revisit the results of the AEB test, by investigating how far from the actual cut-ins the test cases produced by the NSGA-II are.

IV. Revisiting the Search-Based Experiments

We study the distance of the virtual scenarios produced by the second search (see Fig. 1.b). We report the boxplot of distance values at each iteration (see Fig. 3), which allows us to confirm our manual observations. The search quickly deviates from normal situations and converges towards extreme test cases. At the 5th iteration, the median distance is nearly 20, which is already far from the reference dataset. After the 5th iteration, the median distance is always above 30, indicating that the search produces extreme cut-ins compared to the reference. This is not satisfactory because the search is not exploring enough of the scenario subspace close to the baseline. This does not allow us to know whether the system is able to handle situations close to real scenarios. The analysis of the Pareto front confirms that the retained test cases are very distant from real scenarios (see Table I, first row) yielding extreme collision cases.

Determining the relevance of critical scenarios can be complex and even require a domain expert. The interest of the distance concept lies in the possibility to identify, among all the critical cases produced, those which are the least extreme and which may deserve closer analysis. They represent a small proportion of the total number of critical cases. Each test case is then associated with the closest real scenario and we report the difference. This allows us to identify the changes made to a real cut-in and to better judge the relevance of the test case. In Table II, we illustrate this analysis for the critical test case having the smallest distance to the dataset (distance = 10.93). This test case was not retained in the Pareto front but clearly deserves analysis, being the least extreme critical case produced during the search. Table II compares it with its closest real cut-in scenario, both in terms of the difference in values and in terms of the normalized distance along each vector dimension.

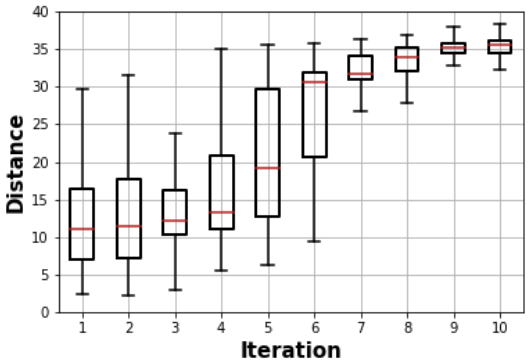


Figure 3. Boxplot of the distances at each iteration

TABLE I. PARETO FRONT DISTANCE DESCRIPTION

Configuration	Min	Median	Max
NSGA II with 2 fitness functions	13	31.08	36.57
NSGA II with 3 fitness functions	1.43	4.82	19.44

TABLE II. ANALYSIS OF THE LEAST DISTANT CRITICAL TEST CASE

ID	RelPos	Ve	Vt	T
Virtual scenario	86.68 m	157.61 km/h	75.67 km/h	4.71 s
Real scenario	108.62 m	154.04 km/h	89.06 km/h	4.89 s
Delta	-21.95 m	3.57 km/h	-13.39 km/h	-0.19 s
Distance	4.48	0.73	4.46	1.26

From the comparison in Table II, the ego speed and lane change duration values are very close. The target speed and relative position values are lower. Following this pattern of changes, we further investigate whether the virtual case can get closer to the real scenario. Starting from the virtual values, we gradually increase the target speed and relative positions until no critical collision is observed under test. We obtain a new critical case having a smaller deviation compared to a real cut-in. This variation analysis can be applied to each pair of virtual/real cut-ins of interest, selected based on the distance metric.

V. DISTANCE-AWARE TEST STRATEGY

While distance provides a valuable indicator for analyzing test results, this information can also be used in the test strategy. As an example of a strategy that takes distance into account, we can use the NSGA-II but add a third fitness function to it. We will try to minimize the distance to the real cut-in.

The search deviates much less than previously from the reference database (see Fig.4, to be compared to Fig. 3). This allows us to considerably reduce the number of extreme critical scenarios (see the comparative plots in Fig.5) and to more extensively explore the scenario subspace close to the reference dataset.

The Pareto front is this time very interesting to study because it provides us with a synthesis of AEB's performance. In the second row of Table I, we observe that the distance is much smaller than previously. Ninety percent of the scenarios are less distant than the previous case shown in Table II. We have a set of 52 scenarios that are very close to the dataset (min distance: 1.43 and median distance: 3.32) for which the AEB manages to avoid the collision. A second set of 13 scenarios with a reasonable distance (min distance: 3.87 and median distance 5.52) that cause collisions at a relative speed lower than 30 km/h. Finally, we have a set of 35 scenarios with a higher distance (min distance: 6.83 and median distance 10.16) that cause critical collisions. The test case with the min distance of 6.83 is also the least distant critical case found by the entire search.

We report this test case in Table III together with its closest real scenario. Interestingly, the real scenario is the same as in Table II. However, the virtual test case is now closer than the one found by the previous search. The variation analysis shows that the test case is actually at the boundary between critical and non-critical scenarios. Any reduction in distance beyond 0.4 makes the scenario non-critical.

Setting a distance threshold of 10, there are 3 additional critical test cases deserving analysis. These ones are not in the Pareto front. The variation analysis shows that they also are boundary cases. The four test cases, including the one in the Pareto front and the three other ones, represent distinct critical cases. There are differences on all parameters, yielding diverse boundary cases of cut-in regarding the abruptness of the lane change and the relative speed and position.

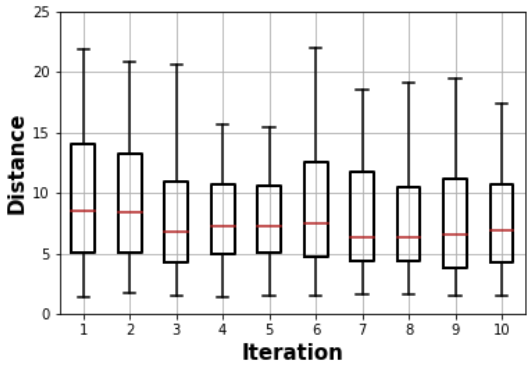


Figure 4. Boxplot of distances with the new test generation process

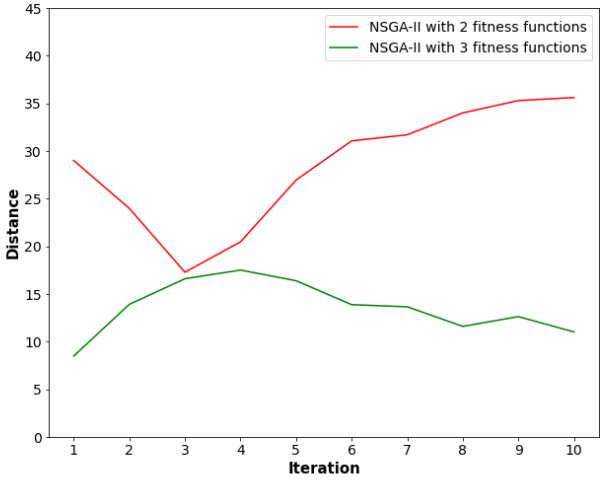


Figure 5. Evolution of the median distance of critical scenarios as a function of iterations.

TABLE III. DESCRIPTION OF THE NEW LEAST-DISTANT CASE

ID	RelPos	Ve	Vt	T
Virtual scenario	105.71 m	167.55 km/h	82.78 km/h	5.10 s
Real scenario	108.62 m	154.04 km/h	89.064 km/h	4.89 s
Delta	-2.91 m	13.51 km/h	-6.28 km/h	0.21 s
Distance	0.59	2.78	2.09	1.4

VI. Conclusion

In this paper, we have addressed the problem of the convergence of search-based testing towards extreme scenarios. If the search is not constrained, it produces a significant number of irrelevant scenarios. We have illustrated the problem through the example of virtual cut-in scenarios for an AEB. Taking a real-world dataset as a baseline allows us to considerably reduce the number of extreme scenarios. The alignment is done by means of a simple distance metric. The distance to real-world conditions proves a very useful indicator for both guiding the test generation and analyzing the results. Compared to the unconstrained search, the distance-aware search better explores the boundary between the safe and unsafe scenarios, and so allows a better characterization of the behavior of the system under test.

This work opens several perspectives. One is to study alternative distance-based test strategies. For example, we may consider strategies that move away progressively, starting with test cases similar to the dataset characteristics, and controlling the allowed distance at each iteration. Another perspective is the development of various usages of the distance concept: for debugging, for comparing two candidate versions of an ADAS function, etc. The debugging would focus in priority on the fail scenarios that are the closest to the normal operational domain. The comparison of candidate ADAS designs could include the smallest distance at which the scenarios become critical in each case. This metric would act as a robustness measurement, quantifying the ability of each ADAS version to assist the driver in situations that deviate from normal driving. Finally, we intend to work on improving the scalability of the approach. The main effort lies in the real data extraction and its mapping to virtual test parameters. The effort has to be repeated for each pair of parametrized scenario and dataset. It could be alleviated by developing a common framework to store the data, with facilities to ease the data querying and data transformation required by the scenarios.

References

[1] Uber's self-driving operator fatal crash [online] Available: https://en.wikipedia.org/wiki/Death_of_Elaine_Herzberg

[2] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?", *Transportation Research Part A :Policy and Practice*, pp. 182–193, 2016.

[3] M. Nentwig, M. Stamminger (2010), "A method for the reproduction of vehicle test drives for the simulation-based evaluation of image processing algorithms," in *Proc. 13th Int. IEEE Conf. on Intelligent Transportation Systems (ITSC)*, Funchal, 2010, pp. 1307-1312.

[4] J. Bach, J. Langner, S. Otten, E. Sax, M. Holzäpfel (2017), "Test scenario selection for system-level verification and validation of geolocation-dependent automotive control systems," in *Proc. International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, Funchal, 2017, pp. 203-210.

[5] A. Lamprecht, T. Ganslmeier (2010), "Simulation Process for Vehicle Applications Depending on Alternative Driving Routes Between Real-World Locations", In: Meyer G., Valldorf J. (eds) *Advanced Microsystems for Automotive Applications 2010*. VDI-Buch. Springer, Berlin, Heidelberg.

[6] R. Krajewski, T. Moers, D. Nerger and L. Eckstein, "Data-Driven Maneuver Modeling using Generative Adversarial Networks and Variational Autoencoders for Safety Validation of Highly Automated Vehicles," *2018 21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2383-2390, doi: 10.1109/ITSC.2018.8569971.

[7] M. R. Zofka, F. Kuhnt, R. Kohlhaas, C. Rist, T. Schamm, and J. M. Zöllner, "Data-driven simulation and parametrization of traffic scenarios for the development of advanced driver assistance systems," In *Proc. 18th International Conference on Information Fusion (Fusion)*, Washington, DC, 2015, pp. 1422-1428.

[8] S. Geyer et al. (2014), "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," *IET Intelligent Transport Systems*, vol. 8, no. 3, pp. 183-189, May 2014.

[9] A. Armand, D. Filliat, and J. Ibañez-Guzman, "Ontology-based context awareness for driving assistance systems," *2014 IEEE Intelligent Vehicles Symposium Proceedings (IV)*, Dearborn, MI, 2014, pp. 227-233.

[10] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving," *2015 IEEE 18th Int. Conf. on Intelligent Transportation Systems (ITSC)*, Las Palmas, 2015, pp. 982-988.

[11] Y. Li, J. Tao, and F. Wotawa, "Ontology-based Test Generation for Automated and Autonomous Driving Functions," *Information and Software Technology*, vol. 117, Jan. 2020.

[12] Q. Xia, J. Duan, F. Gao et al. (2017), "Automatic Generation Method of Test Scenario for ADAS Based on Complexity," SAE Technical Paper 2017-01-1992, SAE 2017 Intelligent and Connected Vehicles Symposium.

[13] C. Sippl C., F. Bock., D. Wittmann, H. Altinger, and R. German, "From Simulation Data to Test Cases for Fully Automated Driving and ADAS," in *Proc. Testing Software and Systems (ICTSS 2016)*, Lecture Notes in Computer Science, vol. 9976, Springer, Cham.

[14] Kluck, F., Zimmermann, M., Wotawa, F., & Nica, M. (2019). Genetic Algorithm-Based Test Parameter Optimization for ADAS System Testing. In *Proceedings - 19th IEEE International Conference on Software Quality, Reliability and Security, QRS 2019,* pp. 418-425

[15] R. Ben Abdessalem, S. Nejati, L. C. Briand and T. Stifter, "Testing Vision-Based Control Systems Using Learnable Evolutionary Algorithms," *2018 IEEE/ACM 40th Int. Conf. on Software Engineering (ICSE)*, 2018, pp. 1016-1026, doi: 10.1145/3180155.3180160.

[16] G. Li et al., "AV-FUZZER: Finding Safety Violations in Autonomous Driving Systems," In *Proc. 2020 IEEE 31st Int. Symp. on Software Reliability Engineering (ISSRE),* Coimbra, Portugal, 2020, pp. 25-36.

[17] R. Ben Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing advanced driver assistance systems using multi-objective search and neural networks," in *Proc. 31st IEEE/ACM Int. Conf. on Automated Software Engineering (ASE)*, Singapore, 2016, pp. 63-74.

[18] A. Calò, P. Arcaini, S. Ali, F. Hauer and F. Ishikawa, "Generating Avoidable Collision Scenarios for Testing Autonomous Driving Systems," *2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*, 2020, pp. 375-386

[19] A. Bussler, L. Hartjen, R. Philipp, and F. Schuldt, "Application of Evolutionary Algorithms and Criticality Metrics for the Verification and Validation of Automated Driving Systems at Urban Intersections," *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 128-135.

[20] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems," in *Proc. 21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2118-2125, doi: 10.1109/ITSC.2018.8569552.

[21] M. El Mostadi, H. Waeselynck, and J-M. Gabriel, "Minimalistic Simulator to Improve Virtual Testing of ADAS, " to appear in *Proc. 11th European congress Embedded Real Time Systems (ERTS)*, Toulouse, France, June 2022.

[22] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, April 2002

[23] NSGA II python library [online] Available: https://platypus.readthedocs.io/en/docs/algorithms.html#nsga-ii