Scenario Reduction of ALKS Development by Using Searching Methods

Namık Zengin[1], **Oğuzhan Derebaşı**[1], **Serdar Kınay**[1], **Bekir Öztürk**[1], **and Harun Kutucu**[1]

[1] AVL Research and Engineering, Turkey

Abstract: This study presents an approach in Automated Lane Keeping Systems (ALKS) within Automated Driving Systems (ADS), integrating scenario parameterization with Particle Swarm Optimization (PSO) and contrasting it with combinatorial testing (CT). Focusing on critical scenarios vital for ALKS safety, the approach uses UN Regulation 157 to establish a parameter space mirroring real-world driving conditions, ensuring practicality. The integrated parameterization-optimization technique efficiently reduces test scenarios without compromising critical performance aspects and deepens the understanding of system behavior under various conditions. Exploring diverse searching algorithms, particularly CT, enriches ADS development processes. The effective use of PSO in identifying critical scenarios and k-means clustering for directing search efforts highlights the potential of combining multiple methods. This research marks a pivotal step in ADS development, especially in scenario-based testing for ALKS, offering insights for more efficient ADS development and laying the groundwork for future refinements aligned with evolving ADS.

Keywords: automated driving systems, scenario parametrization, automated lane keeping systems, critical scenario generation, scenario-based testing

Arama Yöntemleri Kullanılarak ALKS Geliştirme Süreçlerinde Senaryo Sayısının Azaltılması

Özet: Bu çalışma, Otomatik Sürüş Sistemleri (ADS) içindeki Otomatik Şerit Takip Sistemlerinde (ALKS) senaryo parametrelendirmesini Parçacık Sürü Optimizasyonu (PSO) ile entegre eden ve bunu kombinatoryal testle (CT) karşılaştıran bir yaklaşım sunmayı hedeflemektedir. ALKS güvenliği için hayati olan kritik senaryolara odaklanmasının yanında, gerçek dünya sürüş koşullarını yansıtan bir parametre uzayını kurmak için ALKS gereksinimlerini içeren UN Regulation 157'den yararlanılmıştır. Entegre parametrizasyon-optimizasyon tekniği, test senaryolarını verimli bir şekilde azaltmanın yanı sıra çeşitli koşullar altında sistem davranışının anlaşılmasını geliştirmeyi hedeflemektedir. Özellikle CT gibi farklı arama algoritmalarının uygulanması, bu algoritmaların ADS geliştirme süreçlerindeki olası katkılarını analiz etmekle beraber, kritik senaryoların belirlenmesinde PSO'nun ve arama yönelimi için k-means kümelemesinin etkili kullanımı, birden fazla yöntemi birleştirme potansiyelini ortaya çıkarmaktadır. Bu çalışma, ADS geliştirme süreçlerindeki önemli bir adıma çözüm sunmasının yanı sıra, daha verimli ADS geliştirme süreçleri için de öngörüler sağlamakta ve bununla beraber gelişen teknolojiyle birlikte hedeflenen sonraki aşamalara da zemin oluşturmayı amaçlamaktadır.

Anahtar Kelimeler: otomatik sürüş sistemleri, senaryo parametrizasyonu, otomatik şerit takip sistemleri, kritik senaryo üretimi, senaryo tabanlı test

RESEARCH PAPER

Corresponding Author: Namık Zengin, namik.zengin@avl.com

Reference: Namık Zengin *et al.*, (2024), *ITU Computer Science, AI and Robotics*, 1, (1) 59–69.

Submission Date: January, 22, 2024
Acceptance Date: June, 8, 2024
Online Publishing: July, 20, 2024

1 Introduction

ADS refer to the systems with both hardware and software capabilities, enabling sustainable execution of all dynamic driving tasks. Within the Society of Automotive Engineers (SAE) taxonomy for driving automation systems, these systems represent levels 3, 4, and 5 ([1]). Although ADS encompasses levels 3 and beyond, these systems might operate within limited Operational Design Domains (ODD). For example, a level 4 driving system may only function within its designated ODD area. The rapid evolution of ADS technologies not only aims to ensure safer and more comfortable driving experiences for vehicle users but also escalates the effort required in the design and validation processes ([2]). Consequently, there is an increasing demand for more sophisticated simulation tools and virtual validation platforms to meet these needs ([3]).

In the domain of Advanced Driver Assistance Systems (ADAS), validation and testing phases have historically relied on distance-centric approaches. However, for level 3 and higher systems, which are expected to function across all designated ODD conditions, this approach demands a significant amount of time and effort. For instance, a highway-exclusive function may require approximately 6.2 billion kilometers of testing ([4]). In response to the encountered challenges, the PEGASUS working group has undertaken efforts to confront and address these issues. They have conducted studies focusing on a scenario-driven testing approach, believing that will reduce the workload associated with testing, verification, and validation processes for ADS systems ([5]).

As known, the V-model approach is commonly used in system design processes, wherein use cases are created based on stakeholder expectations and user stories ([6]). The creation of use cases generally encompasses the systems' Operational Design Domain (ODD), the scenarios' purpose, actors, preconditions, triggering events, and scenario steps, albeit it may vary based on manufacturers. Each use case is completed by defining main and extension scenario steps, which consequently contribute to the formulation of system requirements and subsequently technical specifications. Following that, the system is developed based on these technical specifications, and the validation and verification process commences ([7]).

Although V-cycle processes tend to function as intended in the development of ADAS, the increasing system complexity in ADS presents challenges in the estimation of effort within the development and verification/validation processes ([8]). Therefore, some studies in the literature propose an iterative progression of steps and the verification of requirements in this manner ([9]). Moreover, considering that conventional testing approaches within the ADS domain demand significant effort, there is a compulsion to implement scenario-based engineering practices. Despite the widespread acceptance in numerous studies regarding the crucial need for scenario engineering approaches in ADS applications, these techniques are gradually finding their role in current practices ([10],[11], [12]). The utilization of scenario engineering approaches in ADS applications is recognized as a significant necessity. However, its integration is currently emerging slowly, primarily due to the recognition of the challenges posed by the iterative nature of steps and the verification of requirements in such complex systems.

To employ scenario-based testing approaches, a common scenario definition, and structure are necessary ([13]). Safety of Intended Functionality (SOTIF) ([14]) standardizes the definitions of scenarios and related terms. Additionally, the term "scenario" should comprise higher-level, abstract representations of the system in the conceptual stage and detailed, concrete descriptions during the developmental phase ([10]). Accordingly, three primary abstraction levels, namely functional, logical, and concrete, are used in scenario representation ([5]). While functional scenarios are delineated as natural language expressions of general outlines, logical scenarios are used for the representation of scenarios with parameter spaces. Conversely, concrete scenarios present specific instances of logical scenarios in tests and evaluations. As the transition progresses from functional to concrete scenarios, the abstraction level decreases, resulting in an increased number of scenarios ([15]).

With the onset of scenario engineering in validation processes, a need for a common scenario standard has emerged. The ASAM OpenSCENARIO ([16]) format (OSC) is utilized for defining logical and concrete scenarios. OSC allows standardization in writing these scenarios, easing scenario sharing across diverse platforms. Consequently, it establishes a common language for scenarios used in driving simulations and autonomous vehicle technologies.

Despite aiming for more efficient test processes through scenario engineering, an increase in the number of scenarios still necessitates a significant workforce for testing. Thus, prioritizing critical areas of system operation, assigning priority to these areas in test processes, and automating test scenario generation based on this prioritization offer a more manageable process ([17]). Scenario parameterization involves defining scenarios by identifying various parameters covering real-world driving situations for ADAS and ADS. These parameters can include factors found in ODD layers such as road geometry, traffic elements, weather conditions, vehicle speeds, and behaviors of other road users. Hence, through the identification of potential maneuvers of the vehicle and surrounding environmental factors in an appropriate scenario, parameters are established, consequently leading to the formulation of a test scenario framework. Through scenario parameterization, it becomes feasible to systematically explore how ADS systems behave under different conditions, reducing the number of test scenarios required. This approach aids in evaluating the performance and reliability of ADS functions, identifying potential corner cases, and ultimately enhancing the safety and reliability of systems. There are various methods and techniques in the field of scenario parameterization for ADS, focusing on diverse sources such as regulations published by government or independent bodies, pilot studies, data obtained from real-road tests, scenario catalogs, or expert knowledge ([18]–[22]).

Furthermore, prevalent techniques employed in the generation of test scenarios embrace probabilistic sampling, machine learning-driven parameterization, combinatorial testing, and genetic algorithms. ([23]). The validation phase of the scenario-based ADS design process starts with a pool of concrete scenarios obtained from scenario parameterization methods. Reasonable interpretation and as-objective-as-possible evaluation of simulation outputs related to scenarios will serve as indicators of the effectiveness and reliability of vital ADAS/ADS functions. In this context, criticality metrics (CM) and key performance indicators (KPI) hold crucial importance in ensuring the safe operation of a function within its defined ODD.

Criticality refers to the total risk involving actors within a traffic situation ([24]). CMs delve into specific aspects of criticality quantitatively and can be associated with influential factors such as spatial, temporal, dynamic, and environmental considerations ([25]). CMs include temporal metrics such as, time-to-collision (TTC), time-to-brake (TTB), time-to-reaction (TTR), spatial metrics such as, minimum longitudinal distance, and minimum lateral distance, and dynamic metrics such as, maximum acceleration demand.

These metrics are instrumental in evaluating the performance and reliability of ADAS functions in real traffic scenarios within their ODD. For instance, KPIs like TTC or minimum longitudinal distance (MLD) can be crucial in the assessment of the safety performance of several functions. Similarly, information on whether an ADAS vehicle collided with another object during a scenario can provide significant insights into the scenario's criticality. These metrics, when combined with different weighting coefficients during the function's validation, can be tailored for different traffic scenarios and ADAS functions, rendering them more practical ([26]).

The criticality information derived from these concrete scenarios can contribute feedback to the parameterization process, thus aiding in its improvement. For example, in one study, TTC was used as a CM, and based on scenario outcomes, a probability density function for scenario parameter interval was established using an importance sampling method. This function indicated that selected parameter combinations would lead to more critical scenarios, thereby reducing the number of scenarios that needed testing. In another study, criticality was determined based on whether a collision occurred in a scenario, and a machine learning model was trained using initial test results. This model was used as a performance boundary in the parameter space to identify critical parameter intervals ([27]). In both cases, these metrics and methods help in assessing the performance and reliability of ADAS/AD functions, identifying potential corner cases, and ultimately enhancing the safety and reliability of systems. Furthermore, ongoing research focuses on different scenario parameterization techniques and these metrics.

In light of all this information, within this study, use cases and scenarios have been generated specifically for the ALKS function. Subsequently, the parameter ranges of the components constituting the scenario are determined, and a closed-loop scenario parameterization approach is applied by integrating it with the PSO algorithm. The main objective of the study is to define the scenario space primarily by employing the proposed integrated parameterization-optimization approach, aiming for a more robust methodology.

The content of the study is outlined as follows: In the second section, the problem definition is presented, and the workflow utilized in the project process is demonstrated. Subsequently, Section 3 discusses why the ALKS function is chosen, followed by an explanation of the stages of scenario generation and scenario parameter selection. In Section 4, the search algorithms employed in the study are elucidated, detailing how they were implemented. Section 5 involves the analysis and discussion of the obtained results and lastly, the study concludes with Section 6, which presents the final remarks.

2 Problem Definition

In the context of ADAS/AD development based on scenarios, one of the objectives is to minimize the necessary testing effort. Besides that, as it is not possible to test such a large scenario space on a real vehicle, the number of scenarios should be tested in a simulation environment. As highlighted in the introduction, the selection of parameters for parameterization varies across scenarios, and given the substantial number of static and dynamic parameters within the parameter space associated with each scenario, simulating all possible combinations leads to an exceptionally expansive concrete scenario space.

Given the time and cost constraints of vehicle development, it is not feasible to test all possible scenarios. Therefore, it is beneficial to reduce the number of concrete scenarios in the scenario space. For instance, if there are five thousand functional scenarios, simulating all possible combinations may result in more than five billion scenarios. Assuming all scenarios are simulated in the Software-in-the-Loop (SiL) environment, each scenario will require a minimum of 20-30 seconds, resulting in thousands of years of simulation duration.

As seen in Fig. 1, firstly the function to be used for parameterization is selected. Various usage scenarios are then created for the function, from which functional scenarios are obtained. Static and dynamic parameters used in these scenarios are selected to create logical scenarios. After this step, two different approaches are used: closed-loop and open-loop. Following these approaches, concrete scenarios are created and the advantages and disadvantages of open-loop and closed-loop approaches are evaluated.

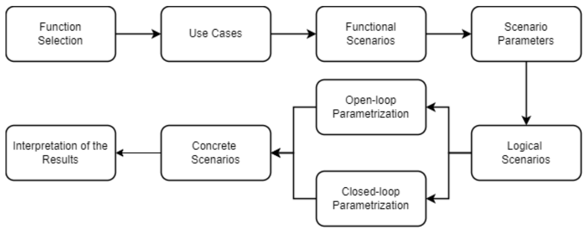


Fig. 1 Workflow of the project.

Therefore, it will be necessary to reduce the number of scenarios to be performed. To achieve this, various open-loop methods are available in the literature, such as combinatorial testing ([28]), various sampling and searching methods ([17]). However, although these methods narrow down the scenario space, they also have some disadvantages. When these methods are applied, it is observed that the most significant disadvantage is the reduction of scenario space, which results in the loss of some critical scenarios during testing.

As seen in Fig. 2 different searching methods used in the open-loop system generate scenarios by pairing and grouping parameter values. Test scenarios are created based on the maximum coverage rate of combinations. This method aims to create the minimum number of test scenarios that cover all parameter values in the entire scenario space. However, it does not include any comments on the criticality of the scenario. As shown in the graph below, test cases generated across the entire scenario space follow a pattern in some areas while being scattered in others.

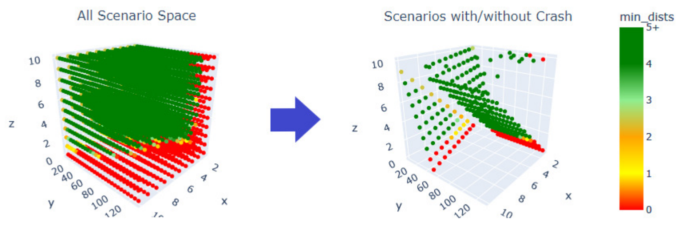


Fig. 2 Characteristic of open-loop parameterization.

In the approach used, the number of concrete scenarios should be reduced, while the sampling outputs should be located in critical areas so that the scenario space is narrowed without losing critical scenarios. If the behavior in critical regions is observed successfully, it can be concluded that, this area is deemed safe. As a result, by testing only the critical scenarios, all other scenarios are also covered. To illustrate with an example, the TTC parameter as the CM is selected. If the scenarios that cover values where the TTC between two vehicles is less than 1 are tested, scenarios where the TTC value is greater than 1 from a criticality perspective will already be covered. To achieve this, a closed-loop toolchain is required.

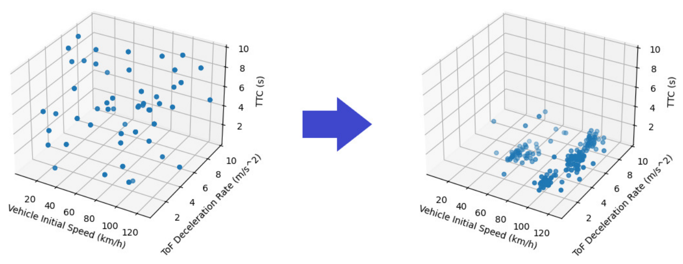


Fig. 3 Characteristic of closed-loop parametrization.

When developing a closed-loop system, the aim is to create a function using critical metrics and identify critical test scenarios region by region. As seen in Fig. 3, in the first step, particles spread throughout the entire scenario space are collected in areas containing criticality based on the function result at the end of the loop. This way, the algorithm will focus on detecting scenarios in critical areas rather than focusing on all scenarios or a single region.

3 Feature Selection and Scenario Generation

As already mentioned in the previous sections, the feature that is selected as the scope of this work is ALKS. Previous works ([29]) were concerned with the Automated Emergency Braking (AEB) feature which had limited potential scenarios to consider since the autonomy provided by the feature is only braking. As detailed in the following parts, ALKS on the other hand provides higher autonomy which means that a vehicle equipped with the feature can find itself in a variety of challenging traffic situations, necessitating a through research of validation scenarios.

ALKS controls the longitudinal and lateral movement of the vehicle without requiring any input from the driver for extended periods. This technology is an intermediate step towards full automation, situated between current driver assistance systems and the advanced, fully autonomous vehicles expected to reach Level 4 or Level 5 automation. ALKS operates under specific conditions to improve driver convenience and efficiency. When ALKS is active, it should take over the driver's responsibilities and manage critical situations, however the driver must be able to regain control when requested. ALKS must also comply with existing traffic laws. From a safety perspective, ALKS aims to reduce potential risks for drivers, passengers, and other road users. Therefore, its reliable operation must meet strict safety and performance criteria set out in the UN regulation 157 ([30]). Various test conditions specified in the regulation assess performance and safety in real-world driving scenarios. Some of these include:

- System Activation Test: Verifies that ALKS activates only in safe conditions, like clear road markings.
- Lane Keeping Test: Assesses the system's ability to maintain the vehicle within its lane in different driving scenarios.
- Transition Demand Test: Tests ALKS's capability to prompt the driver to resume control when necessary.
- Minimum Risk Maneuver Test: Evaluates the system's ability to perform safe maneuvers if the driver doesn't respond to a transition demand.
- Emergency Intervention Test: Ensures the system can handle sudden emergencies requiring immediate driver intervention.
- Sensor Performance Test: Checks the accuracy and reliability of the system's sensors under various environmental conditions.
- Communication with Other Road Users Test: Assesses how the system communicates its actions to nearby traffic.
- Failure Mode and Effect Analysis (FMEA): Simulates system failures to ensure ALKS responds safely and appropriately.
- Operational Domain Test: Confirms that ALKS operates within its designated speed and environmental parameters.

To assess whether these conditions are met and to prove the vehicles' reliability in critical moments, specific use cases have been established, including follow deceleration, target cut-out, target cut-in, and ego lane change. As shown in Fig. 4, target cut-out refers, a lead vehicle in front of the ego vehicle performs a cut-out maneuver to another lane after which a second target vehicle appears on the front. Target cut-in involves the reverse, where the ego is following another vehicle while an adjacent one performs cut in. Ego lane change involves the ego vehicle performing a lane change towards a lane where a target vehicle is closing in from behind. In follow deceleration, a lead vehicle in ego vehicles' lane of travel inside the feature detection range decelerates, as a result ego vehicle decelerates or performs an emergency maneuver which is sketched in Fig. 4. These use cases encapsulate a variety of actions that the ego vehicle can take including longitudinal and lateral movement as well as several target vehicle behaviors. In this work, example results will be demonstrated for the follow deceleration use case.

Since they are both formalized descriptions on actors' behaviors as well as the environmental information, the use cases can be considered as the baseline of the functional scenarios. In order to make the transition into logical scenarios from the functional ones, the necessary parameters to simulate the scenarios as well as their parameter ranges must be determined. These ranges are based on the traffic safety related scenarios as defined in UN regulation 157 and can be seen in Table 1. The initial TTC value essentially allows the simulator to calculate the initial distance between ego and target vehicles based on their speeds. In addition to the parameters in the table, the lateral offset range is determined as a part of designing the system ODD since it is directly based on the lane width. Finally, the initial target speed is selected to be the same as the initial ego speed.

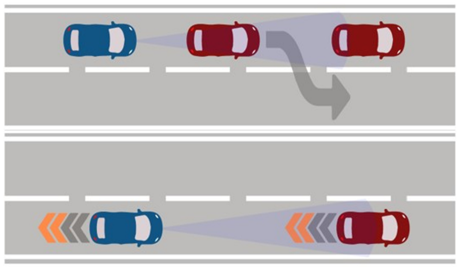


Fig. 4 Demonstration of target cut-out and follow deceleration scenarios.

4 Integration of Searching Algorithm and PSO

4.1 Searching Algorithms and Parametrization Methods

According to ADS regulation ([31]), the functional scenarios are assessed from a qualitative perspective, categorized into three classes: nominal, critical, and failure. Nominal scenarios represent the predictable behaviors of the ADS within the defined ODD. Critical scenarios, on the other hand, are more related to corner cases of the system, encompassing insufficiencies dependent on environmental, human, and operational factors. Failure scenarios, meanwhile, depict the normal or emergency operational status of

Table 1 Lower and upper limits of the parameter ranges for the follow deceleration logical scenario

Ego Vehicle Initial Speed (km/h)	ToF Deceleration Rate (m/s^2)	Initial TTC (s)
0	0	0
130	10	10

the system in cases where components of the ADS experience failure.

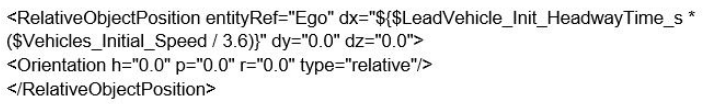
Furthermore, in ISO 3450x ([32], [33]), particularly in ISO 34505 ([33]), the subsequent section after determining the test cases to be executed is expressed as "reduce/combine the test cases." As mentioned in previous sections, while scenario engineering fundamentally aims to reduce the impractical effort of traditional test processes in ADS development, the resulting numbers of functional, logical, and concrete scenarios can still be quite high. At this point, combinatorial testing and design of experiment (DoE) searching methods are recommended in ISO 34505 to address this challenge.

While searching approaches serve to obtain specific regions in the defined scenario space, different areas of interest can be achieved using diverse searching methodologies. These methods generally utilize logical scenarios as input, facilitating the identification of regions in the scenario space that can be deemed critical at the point of concrete scenario transformation ([15]). Moreover, critical scenario regions can be optimized by expanding the boundaries of selected parameters. In addition to the naive search methods created by sampling and combinatorial testing during the exploration of the scenario space, guided search approaches such as optimization and learning-based testing are also available. In this study, a CT approach was employed for scenario reduction, and PSO was utilized to identify the desired areas of interest.

CT which is used as parametrization method in this study is utilized to generate concrete scenarios by defining the range of values for parameters in logical scenarios. The selection of an appropriate parametrization method is contingent upon the parameters within the logical scenario and the desired output. In this work, a case study scenario was formulated based on a following deceleration maneuver, and the input parameters, as well as the potential value range, were established during the earlier phases of the study. The chosen parameters in the section 3 Table 1 encompass the ego vehicle's initial speed (VIT), the deceleration rate of the target object front (ToF), and the time to collision trigger. The combinations of these three parameters constitute the complete factorial concrete scenario space. Depending on the values assigned to parameters, the scenario space may encompass thousands or millions of concrete scenarios. Due to the impracticality of selecting and testing all these concrete scenarios as test cases, parametrization methods are employed, enabling scenario reduction.

4.2 Combinatorial Testing

The Combinatorial Testing approach primarily utilizes the principles of covering array logic. When implementing combinatorial testing on a given dataset, an integer strength coefficient is chosen, which does not surpass the total parameter count. Based on this strength coefficient, an covering array is generated, minimizing the number of test scenarios required to cover the combinations among the values within the dataset. For instance, if the strength coefficient is set to 3 for a dataset with a total of 4 parameters, test cases are formulated to encompass three combinations of the parameters. Various algorithms, such as IPOG (In-Parameter-Order-General) and PICT (Pairwise Independent Combinatorial Testing), are employed to execute the combinatorial testing approach. These methods can generate cases with varying numbers and coverage rates by systematically examining the parameter values in the dataset using different methods based on the strength coefficient. In this study, the FIPOG algorithm, designed for datasets with a substantial amount of data, was utilized to generate an optimal number of test cases. Test cases were created with a chosen strength coefficient of 2. Before executing them in the simulation environment, these test cases were transformed into scenarios in the OSC format. At this stage, the .xosc file was employed to define the initial parameters and performance constraints of the vehicles. To dynamically adjust the relative distance between two vehicles based on their speed, the initial headway time of the leading vehicle was multiplied by the speed of the Ego vehicle. An illustration of this definition in the .xosc format is depicted in Fig. 5.



```
<RelativeObjectPosition entityRef="Ego" dx="${$LeadVehicle_Init_HeadwayTime_s *
($Vehicles_Initial_Speed / 3.6)}" dy="0.0" dz="0.0">
<Orientation h="0.0" p="0.0" r="0.0" type="relative"/>
</RelativeObjectPosition>
```

Fig. 5 Structure of .xosc definition.

After the test scenarios were created in OSC format, these scenarios were executed in the ESMINI simulation environment. ([34]). The assessment of outcomes was predicated on the utilization of the minimum longitudinal distance (MLD) as a metric. From an expert-driven standpoint, instances in which the MLD in the simulation falls within the range of $0-2\ m$ signify critical scenarios indicative of an impending or occurring accident. In cases where the MLD is between $2-3\ m$, it is considered a non-accident yet critical situation. Conversely, distances exceeding $3\ m$ are regarded as indicative of a safe scenario.

4.3 Particle Swarm Optimization (PSO) and Closed Loop Parametrization Approach

Due to the emergence of missing data points during the application of the combinatorial testing method, efforts were initiated to design a closed-loop system. The intent behind implementing the closed-loop system was to concentrate on critical areas in each iteration and eliminate scenarios that deviated from the targeted objectives. Subsequently, a decision was made to construct a closed-loop system employing the PSO method, chosen for its adaptability among search methodologies and its capacity to yield targeted outputs.
PSO is characterized as a heuristic search algorithm utilized for approximating solutions to optimization and search problems. PSO initializes a predetermined particle population in the value space of the parameters in a random and homogeneous manner as in Fig. 6. Each particle in this population initially has its own position and velocity and represents a potential solution to the optimization problem. An initial fitness value is assigned to each particle based on the objective function of the optimization problem. Subsequently, particles adjust their positions and speeds in accordance with their individual costs and the costs of neighboring particles. The movement of a particle is influenced by two primary components: The first is the previous speed of the particle and the second is the difference between its current position and the best positions found by itself and neighboring particles. As a result of mathematical calculations based on these two components, the position and speed of the particle are updated after each iteration. At this stage, the suitability of the particles position is determined using the objective function of the optimization problem. Throughout iterations, particles converge towards the globally best position identified by the entire swarm. The PSO algorithm continues to run until it reaches a satisfactory result or reaches a predetermined number of iterations. In the last iteration, the particles converge to a certain region and the algorithm outputs the particle that offers the best solution.

One of the main reasons for choosing the PSO algorithm for the closed-loop system is to ensure that the particles are directed to the critical region by defining an objective function. Another reason is that the algorithm provides the flexibility to be modified. In this way, it is possible to manipulate the algorithm and direct it to more than one critical region using different methods instead of searching for a single optimum solution. As the output of the system, particles will be spread over a wide area in different critical regions, representing critical scenarios.

Before proceeding with the design of the closed-loop system, essential metrics influencing the chosen scenario were identified. These metrics are Ego maximum deceleration rate ($EgoDec_{max}$), time to brake (TTB), deceleration rate to avoid collision ($DRAC$) ([35]) and minimum longitudinal distance ($MinDist$). Due to the disparate value ranges of these critical metrics, a normalization process was deemed necessary to enable their comprehensive evaluation. Furthermore, certain metrics exhibit a direct proportional impact on criticality, while others demonstrate an inverse effect. For example, an increase in the value of the ego maximum deceleration rate metric corresponds to an increase in the criticality of the scenario, whereas a decrease in the braking time metric renders the scenario more critical. Additionally, the value ranges of these metrics play a role in determining scenario criticality and require variable weighting according to their respective value ranges. These considerations were systematically assessed and processed within the Python script. Eq. 1 shows the mathematical expression of the normalized critical metric as an objective function. The variable C represents the cost, and the terms with hats denote the normalized parameters. Subsequently, the critical metrics were aggregated, and the scenario criticality rate was normalized within the $0-1$ range as shown in Eq. 2. On the resulting $0-1$ scale, 0 represents the most critical scenario, while 1 represents the safest scenario, which is achieved by performing the operation $1-C'$. C' denotes the normalized result (NR) of the cost function. The reason for this is that the PSO algorithm is based on minimizing cost. The resulting cost is used by the objective function in PSO to find target scenarios.

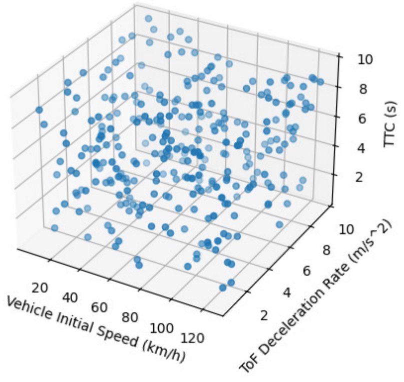


Fig. 6 Distribution of particles in the parameter space when initializing PSO.

$$\begin{aligned} C =& [(\mathrm{Ego}\hat{\mathrm{D}}\mathrm{ec}_{max} \times \mathrm{EgoDec}_w) \\ &+ ((1-Min\hat{D}ist) \times \mathrm{MinDist}_w) \\ &+ ((1-T\hat{T}B) \times \mathrm{TTB}_w) \\ &+ (D\hat{R}AC \times \mathrm{DRAC}_w)] \end{aligned} \tag{1}$$

where:

$$
\begin{aligned}
\hat{\text{EgoDec}}_{max} &= \text{normalized } EgoDec_{max}\\
EgoDec_w &= \text{the weight of } EgoDec_{max}\\
\hat{MinDist} &= \text{normalized } MinDist\\
MinDist_w &= \text{the weight of } MinDist\\
\hat{TTB} &= \text{normalized } TTB\\
TTB_w &= \text{the weight of } TTB\\
\hat{DRAC} &= \text{normalized } DRAC\\
DRAC_w &= \text{the weight of } DRAC
\end{aligned}
$$

$$
C' = \left[\frac{C}{\text{EgoDec}_w + \text{MinDist}_w + \text{TTB}_w + \text{DRAC}_w}\right] \quad (2)
$$

Another issue addressed in closed loop system design is that the PSO algorithm converges to a single global best result. To mitigate this, the k-means method was incorporated to guide particles toward distinct areas. The number of clusters was set to 5 empirically and without searching for an optimum solution. After the first iteration of ESMINI and calculation of metrics are completed, k-means method selects the 5 most critical regions with the help of the particles with minimum cost. During the second iteration of the PSO algorithm, particles are partitioned into regions using the k-means method, which then identifies their central points. Particles belonging to relevant regions are marked and all particles are directed to critical scenarios in those regions, especially the low-cost particles. This implementation results in a multi-directional PSO algorithm.

The operational framework of the closed-loop system primarily comprises small Python modules responsible for executing PSO, ESMINI, clustering method (k-means), and critical metric calculations. Blocks with discrete functions are executed sequentially through a single Python script, initiating the loop. How many iterations the loop will continue is determined by the $iteration_{no}$ parameter. When the cycle starts, the PSO algorithm first runs and produces random particles as many as the specified number of particles. The scenarios corresponding to the produced particles in .osc format are executed in ESMINI, and the resulting data is consolidated in a single output file. The parameters compiled in this phase are normalized using the normalization block, and the costs of the scenarios are ascertained. Before starting the second iteration, the system utilizes the clustering method to segregate and label particles into regions. The particles and their designated regions are continuously updated within the loop until the planned iterations are completed. The complete closed-loop system is illustrated in Fig. 7.

While this study primarily focuses on the longitudinal behavior of the vehicle, future research will expand the scope to include metrics that encompass lateral behavior as well. This expansion will allow for consideration of critical points that may arise based on the AD system's lateral behavior.

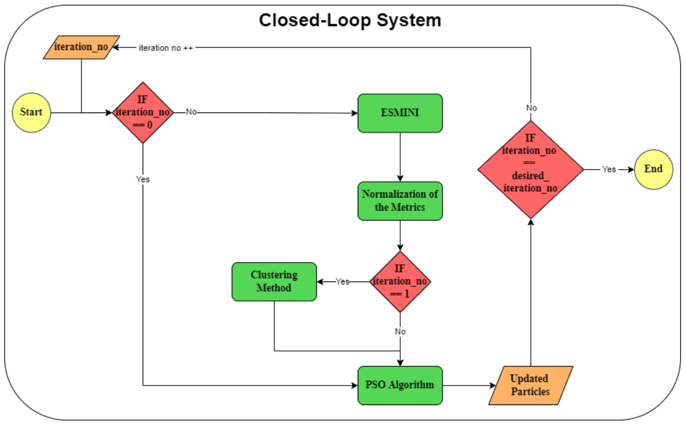


Fig. 7 Complete closed-loop framework.

5 Results and Discussions

In this section, the scenario outputs generated in open-loop and closed-loop systems are analyzed, respectively. As mentioned in Section 4.2, concrete scenarios were initially generated using the FIPOG algorithm for the CT method based on the strength coefficient of 2. The generated scenarios are shown in the Fig. 8. On the left of the Fig. 8, the scenarios produced by the FIPOG algorithm within the entire value space of the relevant scenario are expressed with green dots. The graph in the middle shows which class the produced scenarios belong to on the basis of criticality according to the MLD. In this graph, the MLD represents red between 0 and 1 m, orange represents between 1 and 2 m, and in these scenarios the probability of an accident frequency is high. Situations where the MLD is $2-3$ m are the yellow part that is critical and represents the transition between the accident and the safe zone. Those that are $3-5$ m and above are the points with safe scenarios are expressed in green dots.

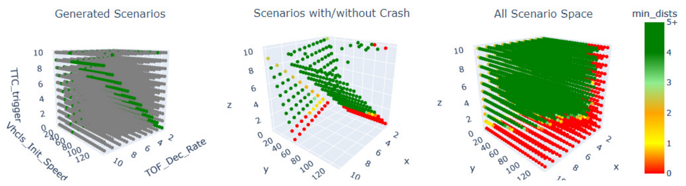


Fig. 8 Classification of the scenarios by using MLD.

A total of 67 scenarios were produced where the minimum longitudinal distance ranges between $0-3$ m, constituting 23.43% of all generated test cases. However, this open-loop system, operating solely with consideration to the coverage of parameter values and lacking an objective function, did not exhibit a conscious orientation towards regions where critical scenarios are likely to be found. Consequently, it is observed that the weight of generating test cases within the $0-3$ m range, which can be considered critical, remained low. Additionally, only a minimal number of scenarios were generated for the relatively rarer scenarios in the $1-4$ m range within the scenario space. Furthermore, employing different metrics beyond the minimum longitudinal distance metric, which was used to assess the generated test scenarios, may yield more precise results regarding criticality. Distribution of the scenarios with respect to MLD is shown in Table 2.

Table 2 Scenario distribution of open-loop system

MLD (m)	Generated scenario number	Rate
$0 \leq MLD < 1$	52	18.18%
$1 \leq MLD < 2$	6	2.10%
$2 \leq MLD < 3$	9	3.15%
$3 \leq MLD < 4$	4	1.40%
$4 \leq MLD < 5$	15	5.24%
$5 \leq MLD$	200	69.93%

In the closed-loop system segment of this study, a loop with 300 particles and 5 clusters running for 20 iterations was established. The obtained results were analyzed based on normalized results, which represent the costs of the particles. Similar to the analysis in the open-loop system, the normalized result output has been divided into five regions concerning criticality. The most critical region is identified within the range of $0-0.2$. The intervals $0.2-0.4$ and $0.4-0.6$ are considered critical and near-critical, respectively, as they are evaluated as situations close to accidents. Particles within the intervals $0.6-0.8$ and $0.8-1.0$ represent safe scenarios. By directing particles to different regions, the particles have successfully identified critical scenarios with costs approaching 0 in these regions. As seen in the Table 3, the most critical region $0-0.2$ has generated a total of 228 scenarios, reaching a proportion of 76%. In the critical ranges of $0.2-0.4$ and $0.4-0.6$, 53 and 19 particles were detected, respectively. These particles may be located in different regions in terms of their positions.

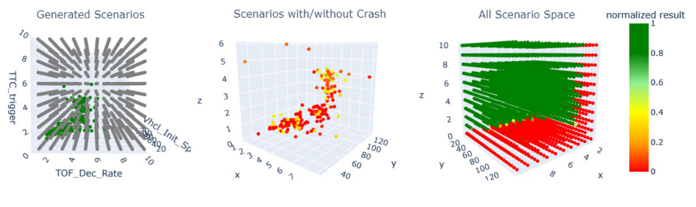


Fig. 9 Classification of the scenarios by using normalized result.

In contrast to the outcomes generated by the open-loop system, the closed-loop system exhibits a higher degree of success in identifying critical regions.

6 Conclusion

As a result, the primary objective of this research was the reduction of testing efforts through the adoption of a scenario-based approach in the design of ADS systems at level 3 and beyond. The study comprehensively examined the ODD of scenarios, along with static and dynamic parameters, and critical metrics that influence scenarios, particularly concerning the ALKS function.

Table 3 Scenario distribution of closed-loop system

Normalized result	Generated scenario number	Rate
$0.0 \leq NR < 0.2$	228	76.00%
$0.2 \leq NR < 0.4$	53	17.67%
$0.4 \leq NR < 0.6$	19	6.33%
$0.6 \leq NR < 0.8$	0	0.00%
$0.8 \leq NR < 1.0$	0	0.00%

One of the key findings of this research is the effectiveness of the integrated parameterization-optimization approach in reducing the number of scenarios without compromising the critical aspects of the systems performance. This approach not only enhances the efficiency of the testing process but also contributes to a more thorough understanding of the systems behavior under various conditions. The methodology adopted in this study offers a practical solution to one of the significant challenges in ADS development: the need to balance the comprehensiveness of testing with time and resource constraints.

Moreover, the studies exploration of various searching algorithms and their application in scenario generation provides valuable insights into the potential of these techniques in refining the development process of ADAS/ADS. Both open-loop and closed-loop methodologies were employed to systematically minimize concrete scenarios, progressing from foundational functional scenarios to identifying critical concrete scenarios. Upon analysis, it was observed that the closed-loop system outputs exhibited a greater efficacy in identifying critical regions. The successful application of the PSO algorithm in identifying critical scenarios and the innovative use of k-means clustering to direct the search towards different critical regions demonstrate the potential of combining multiple methods to achieve optimal results.

Nevertheless, the next plan involves an optimized combination of both methods to ensure comprehensive coverage of scenarios within critical areas, aiming for the minimal number of required test cases. Additionally, an area for potential future exploration lies in determining the optimal number of clusters when applying the k-means method, presenting an additional avenue for research. Lastly, in the upcoming phases, the criticality metrics employed will be extended to incorporate the lateral behavior of the vehicle, facilitating the analysis of critical situations arising from both longitudinal and lateral motion dependencies and a validation strategy will be developed to assess the effectiveness of the employed strategies. This will be followed by an analysis of the performance of the searching methods, particularly within broader ODD and parameter spaces.

References

[1] O.-R. A. D. ORAD Committee, *Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles*. SAE International, 2021.

[2] S. Wagner, A. Knoll, K. Groh, T. Kühbeck, D. Watzenig, and L. Eckstein, "Virtual assessment of automated driving: Methodology, challenges, and lessons learned," *SAE International Journal of Connected and Automated Vehicles*, vol. 2, no. 12-02-04-0020, pp. 263–277, 2019.

[3] D. Kibalama, P. Tulpule, and B.-S. Chen, "Av/adas safety-critical testing scenario generation from vehicle crash data," SAE Technical Paper, Tech. Rep., 2022.

[4] W. Wachenfeld and H. Winner, "Die freigabe des autonomen fahrens," *Autonomes Fahren: technische, rechtliche und gesellschaftliche Aspekte*, pp. 439–464, 2015.

[5] A. Audi and A. Volkswagen, "The PEGASUS method,"

[6] I. R. W. Group, "Guide for writing requirements," *INCOSE: San Diego, CA, USA*, 2019.

[7] ISO/IEC 15288, *Iso/iec 15288, systems and software engineering-system life cycle processes*, 2008.

[8] C. Sippl, F. Bock, C. Lauer, A. Heinz, T. Neumayer, and R. German, "Scenario-based systems engineering: An approach towards automated driving function development," in *2019 IEEE International Systems Conference (SysCon)*, IEEE, 2019, pp. 1–8.

[9] C. Bergenhem, R. Johansson, A. Söderberg, *et al.*, "How to reach complete safety requirement refinement for autonomous vehicles," in *CARS 2015-Critical Automotive applications: Robustness & Safety*, 2015.

[10] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1821–1827.

[11] M. Zipfl, N. Koch, and J. M. Zöllner, "A comprehensive review on ontologies for scenario-based testing in the context of autonomous driving," *arXiv preprint arXiv:2304.10837*, 2023.

[12] B. Schütt, S. Otten, and E. Sax, "Exploring the range of possible outcomes by means of logical scenario analysis and reduction for testing automated driving systems," *arXiv preprint arXiv:2306.12738*, 2023.

[13] M. Steimle, T. Menzel, and M. Maurer, "Toward a consistent taxonomy for scenario-based development and test approaches for automated vehicles: A proposal for a structuring framework, a basic vocabulary, and its application," *IEEE Access*, vol. 9, pp. 147 828–147 854, 2021.

[14] ISO 21448, "Road vehicles-safety of the intended functionality," *Standard, International Organization for Standardization, Geneva, CH*, 2019.

[15] X. Zhang, J. Tao, K. Tan, *et al.*, "Finding critical scenarios for automated driving systems: A systematic literature review," *arXiv preprint arXiv:2110.08664*, 2021.

[16] ASAM. "ASAM OpenSCENARIO." Online; Accessed: 01.10.2022. ().

[17] H. Felbinger, F. Klück, Y. Li, *et al.*, "Comparing two systematic approaches for testing automated driving functions," in *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, IEEE, 2019, pp. 1–6.

[18] T. A. Dingus, S. G. Klauer, V. L. Neale, *et al.*, "The 100-car naturalistic driving study, phase ii-results of the 100-car field experiment," United States. Department of Transportation. National Highway Traffic Safety, Tech. Rep., 2006.

[19] A. Zlocki, A. König, J. Bock, *et al.*, "Logical scenarios parameterization for automated vehicle safety assessment: Comparison of deceleration and cut-in scenarios from japanese and german highways," *IEEE Access*, vol. 10, pp. 26 817–26 829, 2022.

[20] J. Hiller, S. Koskinen, R. Berta, *et al.*, "The l3pilot data management toolchain for a level 3 vehicle automation pilot," *Electronics*, vol. 9, no. 5, p. 809, 2020.

[21] Y. Barnard, S. Innamaa, S. Koskinen, H. Gellerman, E. Svanberg, and H. Chen, "Methodology for field operational tests of automated vehicles," *Transportation research procedia*, vol. 14, pp. 2188–2196, 2016.

[22] B. Schütt, J. Ransiek, T. Braun, and E. Sax, "1001 ways of scenario generation for testing of self-driving cars: A survey," *arXiv preprint arXiv:2304.10850*, 2023.

[23] J. Tao, Y. Li, F. Wotawa, H. Felbinger, and M. Nica, "On the industrial application of combinatorial testing for autonomous driving functions," in *2019 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, IEEE, 2019, pp. 234–240.

[24] L. Westhofen, C. Neurohr, T. Koopmann, *et al.*, "Criticality metrics for automated driving: A review and suitability analysis of the state of the art," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, pp. 1–35, 2023.

[25] L. González, E. Martí, I. Calvo, A. Ruiz, and J. Pérez, "Towards risk estimation in automated vehicles using fuzzy logic," in *Computer Safety, Reliability, and Security: SAFECOMP 2018 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, Västerås, Sweden, September 18, 2018, Proceedings 37*, Springer, 2018, pp. 278–289.

[26] B. Huber, S. Herzog, C. Sippl, R. German, and A. Djanatliev, "Evaluation of virtual traffic situations for testing automated driving functions based on multidimensional criticality analysis," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–7. DOI: 10.1109/ITSC45102.2020.9294169.

[27] F. Batsch, A. Daneshkhah, M. Cheah, S. Kanarachos, and A. Baxendale, "Performance boundary identification for the evaluation of automated vehicles using gaussian process classification," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 419–424. DOI: 10.1109/ITSC.2019.8917119.

[28] X. Hu, B. Zhu, D. Tan, N. Zhang, and Z. Wang, "Test scenario generation method for autonomous vehicles based on combinatorial testing and bayesian network," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, p. 09 544 070 221 125 523, 2022.

[29] N. Zengin, O. Derebası, B. Öztürk, H. Kutucu, and S. Kınay, "Arama metotları kullanılarak aeb senaryolarına yönelik parametrizasyon tabanlı senaryo üretimi ve analizi parameterization based scenario generation and analysis for aeb scenarios by using search methods," 2023.

[30] UN Regulation 157, *157–uniform provisions concerning the approval of vehicles with regards to automated lane keeping systems [2021/389], 1 2021*, 2022.

[31] European Commission and Council, "Regulation (eu) 2012/1426 of the european parliament and of the council of 5 august 2022 on the the automated driving system (ads) of fully automated vehicles," 2022.

[32] ISO 34502, "Road vehicles — Scenario-based safety evaluation framework for automated driving systems," *International Organization for Standardization, Geneva, Switzerland*, 2022.

[33] ISO 34505, "Road vehicles — scenario evaluation and test case generation," *International Organization for Standardization, Geneva, Switzerland*, 2022.

[34] E. Knabe *et al.*, "Environment simulator minimalistic (esmini)," *Accessed on*, vol. 20, 2021.

[35] B. Huber, S. Herzog, C. Sippl, R. German, and A. Djanatliev, "Evaluation of virtual traffic situations for testing automated driving functions based on multidimensional criticality analysis," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2020, pp. 1–7.