A Method for the Runtime Validation of AI-based Environment Perception in Automated Driving Systems*

Iqra Aslam*, Abhishek Buragohain†, Daniel Bamal‡, Adina Aniculaesei§, Meng Zhang¶, and Andreas Rausch‖
Institute for Software and Systems Engineering, Technische Universität Clausthal
Clausthal-Zellerfeld, Germany
Email: { iqra.aslam*, abhishek.buragohain†, daniel.bamal‡, adina.aniculaesei§, meng.zhang¶, andreas.rausch‖ } @tu-clausthal.de

***Abstract*—Environment perception is a fundamental part of the dynamic driving task executed by Autonomous Driving Systems (ADS). Artificial Intelligence (AI)-based approaches have prevailed over classical techniques for realizing the environment perception. Current safety-relevant standards for automotive systems, International Organization for Standardization (ISO) 26262 and ISO 21448, assume the existence of comprehensive requirements specifications. These specifications serve as the basis on which the functionality of an automotive system can be rigorously tested and checked for compliance with safety regulations. However, AI-based perception systems do not have complete requirements specification. Instead, large datasets are used to train AI-based perception systems. This paper presents a function monitor for the functional runtime monitoring of a two-folded AI-based environment perception for ADS, based respectively on camera and LiDAR sensors. To evaluate the applicability of the function monitor, we conduct a qualitative scenario-based evaluation in a controlled laboratory environment using a model car. The evaluation results then are discussed to provide insights into the monitor's performance and its suitability for real-world applications.**

Keywords-runtime monitoring; function monitor; dependable safety-critical system; automated driving system; perception system.

I. Introduction

In principle, fully autonomous vehicles are technically feasible. However, after the initial proof-of-concept testing under ideal conditions, e.g., in lab environments [1], or on restricted test fields [2], further innovative verification and validation techniques are needed during the approval and release processes. These additional verification and validation phases are necessary to gather the required evidence for the safety and reliability of the autonomous vehicle in real-world scenarios. For the commercial approval of autonomous vehicles by certification bodies, the current state-of-the-art practices require verifying specific maneuvers using predefined test scenarios and statistically analyzing real-time data covering millions of kilometers of driving. For autonomous vehicles at Society of Automotive Engineers (SAE) Level 3/4 and above, the key challenge lies in ensuring the safe commercial release and the safe vehicle operation in all possible situations, not just only in those situations encountered during the system development, e.g., through random tests.

Today's automated driving systems are primarily designed to be fail-safe systems, capable of switching the ego-vehicle to a safe state, e.g., by activating an emergency brake. However, future ADSs must be designed as fail-operational systems, especially as there are many situations in which an immediate fail-state might not be readily accessible, e.g., when driving at high speed on the highway. Moreover, in case issues occur during the vehicle operation, the control over the dynamic driving task can no longer be simply handed back to the human driver, since human intervention is not mandated anymore at SAE L4+ [3]. Without a human fallback system, the ADS must be able to take over control and establish a safe state for the vehicle, if a problematic situation arises.

In recent years, a high-level functional architecture has been established for ADSs comprising three main subsystems: (1) environment and self-perception, (2) situation comprehension and action planning, and (3) trajectory planning and motion control [4], [5]. The environment and self-perception is particularly significant as it strongly impacts the performance of the entire ADS and the safety of the autonomous vehicle, as shown by a series of accidents involving (partially) automated driving functions, e.g., Tesla's autopilot. In the first notable accident in 2016, the Tesla's autopilot has failed to detect an articulated lorry driving in the opposite direction which was engaged in a turn maneuver, despite having been successfully tested over 200 million kilometers. In the context of a brightly lit sky, both the driver and the autopilot were unable to recognize the lorry, which had white sides [6]. In response to this accident, Tesla announced the introduction of Shadow Mode to enhance the safety of its autopilot [7]. An approach for continuous monitoring of autonomous driving functions for development, validation and series operation has been proposed in in [8] and demonstrated for the lane changing functionality of the highway pilot in [9]. This approach essentially extends the concept of shadow mode, by addressing two questions: (1) does the autonomous driving function operate correctly (qualitative oracle) and (2) is the autonomous driving function currently operating in a known environment (quantitative oracle) [9].

There is a noticeable gap in research work regarding the validation of environment and self-perception methods applied in automated driving applications. AI-based approaches have prevailed over classical approaches for realizing environment perception, as the former are used both in image processing and in signal processing of other raw sensor data, such as radar. Current safety-relevant standards for automotive systems, e.g., ISO 26262 [10] and ISO 21448 [11], assume the existence of complete requirements specifications. The system development process is usually organized using a structured process model, e.g., the V-model. However, challenges arise with AI-based perception systems since they do not have complete requirements specifications. Instead, the development usually begins with incomplete artifacts, e.g., system requirements formulated for the entire ADS structured test cases derived from other artifacts, e.g., in the format of OpenSCENARIO [12] or OpenDRIVE [13] formats. Additionally, the development of AI-based systems requires extensive training. For instance, for the development of an AI-based system for pedestrian detection an substantial training data set consisting of diverese pedestrian images is required.

The approach outlined in [9] by Mauritz and his co-authors can be understood as developing a dependability cage for monitoring the lane changing functionality of the highway pilot. While their focus is monitoring the entire autonomous driving function without particularly considering the environment perception, in this work we focus on monitoring the environment and self-perception subsystem of an ADS. We propose a dependability cage for validating an environment and self-perception system including redundant perception components that operate with multiple sensor data sources. In this dependability cage, an essential component is a function monitor which evaluates the consistency of the outputs produced by the redundant perception components during the ADS operation. This function monitor is derived from a high-level safety requirement established for the environment and self-perception subsystem. In this work, we evaluate the dependability cage approach for AI-based environment perception qualitatively in a lab environment using predefined test scenarios.

The rest of the paper is structured as follows. Section II gives an overview of related work in the area of verification and validation of AI-based system, with a specific focus on environment perception systems in both robotic and automotive applications. Section III presents the dependability cage approach for the monitoring and validating AI-based environment perception during the ADS operation. In Section IV, we conduct a a qualitative scenario-based evaluation and discuss the obtained results. Section V concludes the paper by summarinzing its contributions and outlining potential future work directions.

II. Related Work

Environment- and self-perception is an integral part of the dynamic driving task that is carried out by the ADS. Primarily based on AI models, it provides essential inputs for further safety-critical functions of the ADS, such as Situation Comprehension and Trajectory Planing. Through the interpretation of various raw sensor data into detailed semantic information about the surrounding driving environment, the AI-based environment perception subsystem enables the ADS to complete its decision making, motion planning and control command execution, by relying on its respective planning and decision components.

For this reason, monitoring and evaluating the functional behavior and the performance of AI-based environment perception systems at runtime is extremely important for the safety of the autonomous vehicle. Recently, diverse research approaches have addressed the safety issues of AI-based environment perception. Czarnecki [14] identifies a set of influencing factors for AI-based environment perception: (1) conceptual uncertainty, (2) development situation and scenario coverage, (3) situation or scenario uncertainty, (4) sensor properties, (5) labeling uncertainty, (6) model uncertainty, and (7) operational domain uncertainty. Identifying these factors is understood as the first step, which should be followed by a systematic analysis of their impact on the perceptual uncertainty. In addition, methods to eliminate or reduce their negative effects on the perceptual uncertainty. Subsequently, mitigation measures are proposed to be applied for the cases when the control of the negative effects is not possible [14]. The concept in [14] revolves around using these methods to gather the essential evidence to substantiate claims regarding the environment perception uncertainty in safety cases that contribute to demonstrating the safety of the overall autonomous vehicle [14].

Another survey presented in [15] identifies various research directions concerning the runtime performance monitoring of AI-based perception in autonomous robots. One direction encompasses approaches using past examples of failures and successes or similarity of operational context to previous experiences to predict the quality of perception output [15]. The second direction involves by methods that detect inconsistencies in perception output, using the input data from a single sensor or from multiple sensors, or by comparing outputs from different models [15]. The third direction focuses on methods for uncertainty estimation by indicating low confidence in prediction output, calculating confidence scores as a measure of the target model's output quality, and detecting anomalies [15]. Several other studies also aim to validate the accuracy or robustness of the perception system outputs concerning specific inputs of the neural network. However, the approaches surveyed in [16] and [17] are primarily limited to offline verification.

In addition to estimating uncertainty, some recent studies have shifted their primary focus to object detection as the main task of the perception system under analysis. Thus, Feng et al. [18] evaluate LiDAR-based object detectors using the Jaccard Intersection over Union (IoU) metric and KITTI and Waymo datasets, incorporating label uncertainty specifically for the bounding boxes of a particular object class, e.g., the object class *car*. In [19], the authors propose a framework to predict performance monitoring of object detection at runtime without relying on any ground-truth data. Meanwhile, in [20], they monitor the performance of the object detector deployed on mobile robots by predicting the quality of its mean average precision using a sliding window technique over the input frames. However, the approach presented in [21] for monitoring of neural networks that estimate 3D human shapes and poses from images is limited to human pose estimation.

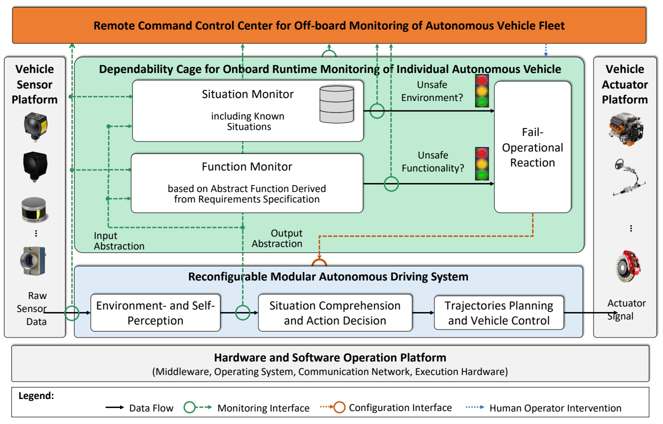
Remote Command Control Center for Off-board Monitoring of Autonomous Vehicle Fleet
Vehicle Sensor Platform
Dependability Cage for Onboard Runtime Monitoring of Individual Autonomous Vehicle
Vehicle Actuator Platform
Situation Monitor
including Known Situations
Unsafe Environment?
Function Monitor
based on Abstract Function Derived from Requirements Specification
Unsafe Functionality?
Fail-Operational Reaction
Input Abstraction
Output Abstraction
Reconfigurable Modular Autonomous Driving System
Raw Sensor Data
Environment- and Self-Perception
Situation Comprehension and Action Decision
Trajectories Planning and Vehicle Control
Actuator Signal
Hardware and Software Operation Platform
(Middleware, Operating System, Communication Network, Execution Hardware)
Legend:
Data Flow
Monitoring Interface
Configuration Interface
Human Operator Intervention

Figure 1: High-level Architecture of the Dependability Cage for Runtime Validation AI-based Environment Perception Systems.

In previous research, the majority of studies have made significant contributions towards the evaluation methods, applications and understanding of uncertainty in perception systems. Nevertheless, most of these studies focus on a single object detection network. They do not address the comprehensive validation of perception system outputs concerning reference sensors or the comparative analysis of outputs from redundant perception systems. Additionally, there is a research gap in detecting perception failures or incorrect behaviors of environment perception systems at runtime to ensure the safety of the perception-equipped system. In this work, we propose an approach for the runtime validation of environment perception in autonomous vehicles (AVs), by analysing and comparing the outputs of two redundant perception systems utilizing camera and respectively LiDAR sensors data. We evaluate this approach qualitatively using predefined scenarios and a model car in a lab environment.

III. Integrated Safety Architecture with Dependability Cage for the Runtime Validation of AI-based Environment Perception in Automated Driving Systems

This section introduces the Dependability Cage approach for the runtime monitoring and validation of AI-based environment perception in automated driving systems. Figure 1 illustrates the high-level architecture of this dependability cage, that can be understood as an instantiation of the overarching concept of the Dependability Cage approach, presented first in [22]. The subsequent sections offer a detailed introduction to the architecture of the function monitor, derived on the basis of the Dependability Cage approach.

A. *Overall Concept of the Dependability Cage Approach*

In the initial position paper, the Dependability Cage approach was introduced to address three main challenges in the development of autonomous systems: (1) guaranteeing the safe behavior of an autonomous system in an unknown and uncertain environment, (2) ensuring the safe behavior for all safety-critical system components, including machine-learning based components, even when deviations are detected in their behavior during system operation time, and (3) guaranteeing and improving the relevance and completeness of test cases for the validation of the system under test [22].

To tackle these challenges, the paper in [22] proposes an iterative development process consisting of three primary stages. The first stage is *Dependability Cages Engineering and Training in System Development*, in which the dependability cages are engineered in parallel to the autonomous system and tested using simulation tests or tests in a restricted and controlled lab environment [22]. In the second stage, *Runtime System Observation and Resilience System Stabilization*, the dependability cages are used to monitor the system behavior during its operation and record any deviation in the system behavior compared to the test results obtained during system development as well as any novel situations that occur in the environment [22]. In the third stage, *Monitored Data Analysis and Goal-Oriented System Evolution for Dependability Improvement*, the observations logged during system operation are leveraged to improve the development artifacts during the subsequent system development cycle [22]. The deployment of the dependability cages on the actual system during its operation is facilitated by a modular platform architecture used for the seamless development and operation of the system, monitor and system environment [22]. The end goal of this iterative development process is to continuously improve the system's quality in terms of its dependability requirements. For further details on the development process and its phases, the reader is referred to [22]. The safety architecture proposed for autonomous systems in [22] draws its inspiration from the second phase of the iterative development process, *Runtime System Observation and Resilience System Stabilization*. This phase is carried out through a continuous monitoring framework that observes the behavior of the overall system, as shown in [22], [23], and [2]. In this paper, we refine the focus of this monitoring framework and tailoring it to analyze the environment perception subsystem of an ADS, rather then the ADS as a whole. The monitoring framework consists of two types of monitors, a *function monitor* and a *situation monitor*. Additionally, it involves a component responsible for defining the fail-operational reaction of the ADS based on the results of these two runtime monitors, as depicted in Figure 1.

The responsibility of the *situation monitor* - denoted as the quantitative monitor in the original position paper [22] - is to evaluate the input abstract situations used in the environment and self-perception subsystem of the ADS. During system operation, the situation monitor assesses if the input situations encountered by the ADS align with those considered during the development phase of the environment perception. As the research in this paper does not focus on the situation monitor, the reader is referred to [24] and [25] for a more detailed description of its concept and functionality.

On the other hand, the *function monitor* - denoted as the qualitative monitor in the original position paper [22] - evaluates if there is a critical deviation in the behavior of the environment and self-perception subsystem during the operation of the ADS. The function monitor consists of an abstract

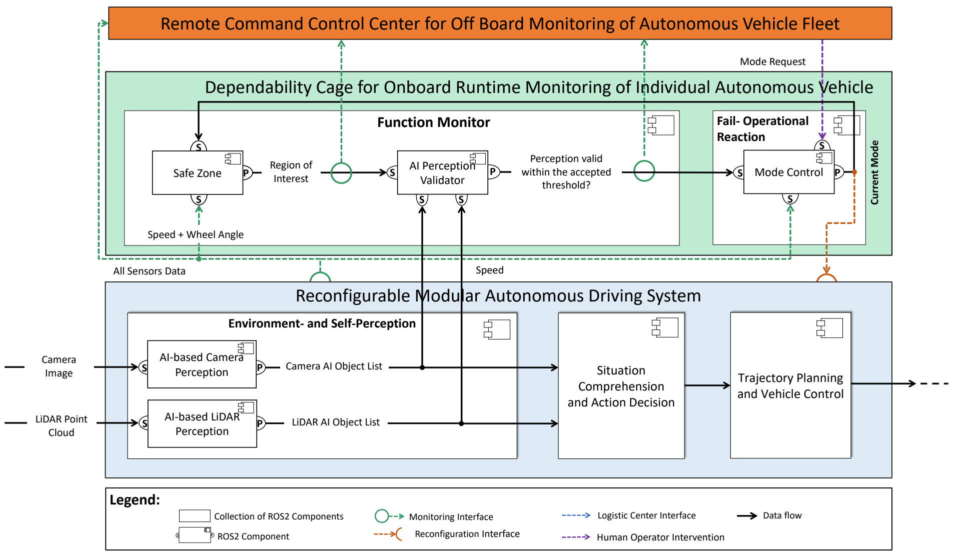
Remote Command Control Center for Off Board Monitoring of Autonomous Vehicle Fleet
Mode Request
Dependability Cage for Onboard Runtime Monitoring of Individual Autonomous Vehicle
Function Monitor
Safe Zone
Region of Interest
AI Perception Validator
Perception valid within the accepted threshold?
Fail- Operational Reaction
Mode Control
Current Mode
Speed + Wheel Angle
All Sensors Data
Speed
Reconfigurable Modular Autonomous Driving System
Environment- and Self-Perception
Camera Image
AI-based Camera Perception
Camera AI Object List
LiDAR Point Cloud
AI-based LiDAR Perception
LiDAR AI Object List
Situation Comprehension and Action Decision
Trajectory Planning and Vehicle Control
Legend:
Collection of ROS2 Components
ROS2 Component
Monitoring Interface
Reconfiguration Interface
Logistic Center Interface
Human Operator Intervention
Data flow

Figure 2: Architecture of the Function Monitor for AI-based Perception Systems.

behaviour's boundary function and a conformity oracle. The abstract behaviour's boundary function dynamically computes a set of safety boundaries, conceptualized as a Region Of Interest (ROI) for the ADS. The conformity oracle assesses the abstraction of the environment and self-perception's outputs in order to check if these outputs are consistent with each other within the safety boundary. Consistency is evaluated based on certain threshold values that are established by empirical tests during the development phase of the environment and self-perception of the ADS. The architecture of the function monitor is explained in detail in the Section III-B.

B. Dependability Cage for AI-based Environment Perception in the ADS's Integrated Safety Architecture

The dependability cage for AI-based environment perception is integrated in the three layered safety architecture developed for ADSs. Beginning from the top, the first layer is represented by the *Remote Command Control Center (CCC)*, where the sensors data stream is visualized along with the results provided by the components in the layers below [23], [1]. The remote CCC has been previously showcased in a separate work [1] and is therefore not the primary focus of the current paper.

The research focus of this paper lies in the middle layer and the bottom layer of the integrated safety architecture. The middle layer encompasses the dependability cage for the AI-based environment perception of the ADS. The dependability cage consists of two main components: (1) the function monitor, responsible for observing and analyzing the environment perception system during the ADS's operation and (2) the fail-operational reaction component, which triggers a fail-operational reaction of the ADS based on the results of the function monitor. The third layer represents the reconfigurable modular autonomous driving system [26], which draws inspiration from previous work [4], [5], and [27], and consists of three main subsystems: (1) Environment and Self-Perception, (2) Situation Comprehension and Action Decision, and (3) Trajectory Planning and Vehicle Control.

The environment and self-perception subsystem consists of two components, AI-based Camera Perception and AI-based LiDAR Perception. These components use camera and respectively LiDAR sensor data to detect objects in the AV's environment. They implement safety-critical machine-learned functions for the ADS's operation. Each component produces an object list, denoted as Camera AI-based (CAI) object list and LiDAR AI-based (LAI) object list. The object lists are used by the other components in the architectural pipeline of ADS. Several pieces of information are provided for each object in the two object lists: (1) object's class, (2) the object's dimensions, height and width, (3) object's distance from the ego-vehicle, (4) sensing timestamp, and (5) confidence level of detection. Figure 2 provides an overview of the integrated safety architecture with a focus on the function monitor and the environment and self-perception system.

The function monitor observes the behavior of the environment and self-perception subsystem against to a specified safety requirement. The safety requirement is informally formulated in a controlled natural language as follows:

Safety Requirement (Informal Specification). The environment and self-perception system must always consistently detect the objects located inside the autonomous vehicle's region of interest using at least two different sensor data sources.

This safety requirement mandates two aspects: first, that a ROI is computed around the AV, and second, that the objects detected by the perception components in the vehicle's region of interest are consistent with each other. The function monitor consists of two components, which allow it to check this safety requirement during the ADS's operation, denoted as *Safe Zone* and *AI Perception Validator*.

1) Safe Zone: This component utilizes various parameters of the AV, e.g., current speed, steering angle, physical dimensions of the AV, acceleration and deceleration, to dynamically calculate ROI. The ROI expands around the AV in the vehicle's direction of travel, and is divided in two areas: a *focus zone* marked in orange and a *clear zone* marked in green around the ego-vehicle. Figure 5 gives a visual intuition of the ROI, depicted on the Graphical User Interface (GUI) of the remote CCC. The ROI around the ego-vehicle is understood as a safety-critical area in which the outputs of the two perception components align consistently. Further details regarding the algorithm used for the ROI computation can be found in [23].

2) AI Perception Validator: This component takes as inputs the ROI computed by the Safe Zone component and the object lists produced by the LiDAR-based and camera-based perception components and computes a boolean flag *valid*, which indicates if the object lists from both perception components are consistent within certain threshold limits (as indicated in lines 1 - 2 in Figure 3). The AI Perception Validator leverages the computed ROI to prune the set of objects detected by the camera and LiDAR sensors in the respective fields of view (as indicated in lines 3 - 4 in Figure 3). It prioritizes those objects detected in close proximity to the ego-vehicle, i.e. inside the ROI of the vehicle. Such a prioritisation differentiates clearly between the objects situated inside the ROI that are safety-relevant for the AV, from the objects situated outside the ROI, which do not pose an immediate safety concern. The threshold limits for determining the consistency of the two object lists are established through a Hazard Analysis and Risk Assessment (HARA). Given that the camera and LiDAR sensors operate at different time rates due to their inherent configuration [28], the CAI object list and the LAI object list will be generated at different frequencies. This time synchronization problem is addressed by using a timeout limit in the AI Perception Validator for the timestamp of the detected objects. It means that an object with a timestamp older than the timeout limit is filtered out from the respective object list (as indicated in lines 9 - 22 in Figure 3). Subsequently, the AI Perception Validator compares the attributes of each object in the CAI object list with the corresponding attributes of each object in the LAI object list, e.g. object class, object distance from the ego vehicle, width and height of the object. If the difference between the respective attributes does not exceed the respective threshold value determined through the HARA analysis, then it is considered a match between both object lists and the AI Perception Validator returns $true$ (as indicated in lines 23 - 26 in Figure 3). It means that the outputs of the two perception components are consistent with each other and all objects have passed the validation and matching criteria. Another case for returning $true$ is that both CAI object list and LAI object list are empty. In this case, there is no need to compare the two object lists. If at least one object in any of the two lists does not meet the comparison criteria, the AI Perception Validator returns $false$, meaning that the outputs of the two perception components are inconsistent (as indicated in lines 27 - 28 in Algorithm 3).
The result of the AI Perception Validator as well as the two

Algorithm: AI Perception Validator Algorithm

```
1  Input: caiObjList - CAI object list; laiObjList - LAI
     object list, roi - vehicle's ROI
2  Output: valid - a boolean flag
3  cameraObjList ← FilterObjectList(caiObjList, roi);
4  lidarObjList ← FilterObjectList(laiObjList, roi);
5  valid ← true;
6  removed ← false;
7  if cameraObjList and lidarObjList are empty then
8      return valid
9  foreach lidarObj in lidarObjList do
10     if timestamp of lidarObj is older than the
11     timeout limit then
12         if lidarObj is not matched then
13             valid ← false;
14         remove lidarObj from lidarObjList;
15         removed ← true;
16 foreach cameraObj in cameraObjList do
17     if timestamp of cameraObj is older than the
18     timeout limit then
19         if cameraObj is not matched then
20             valid ← false;
21         remove cameraObj from cameraObjList;
22         removed ← true;
23 foreach lidarObj in lidarObjList do
24     foreach cameraObj in cameraObjList do
25         if difference between lidarObj attributes and
           cameraObj attributes are under the threshold
           values then
26             mark lidarObj and cameraObj as matched;
27     if removed is true then
28         return valid
```

Figure 3: AI Perception Validator Algorithm.

object lists are forwarded to the remote CCC for visualization (see Section IV). Additionally, the result computed by the AI Perception Validator serves as an input in the component *Mode Control*. In case of inconsistency between the two object lists, this component is responsible for triggering a fail-operational reaction of the AV, by gracefully degrading the ADS functionality. This process is similar to approach outlined in [23]. However, since the primary research focus of this paper is the function monitor, the definition, implementation and evaluation of the corresponding fail-operational reaction will be addressed in future work.

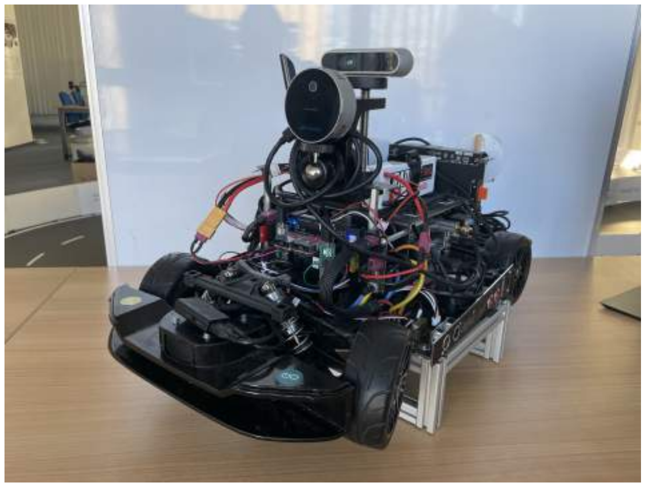
Figure 4: Model Car

IV. Evaluation and Discussion of Results

This section presents the evaluation of the function monitor for the runtime validation of AI-based environment perception in ADS. The first part of this section introduces a detailed overview of the setup, including both hardware and software details (see Section IV-A). Subsequently, in Section IV-B, various test scenarios and several working hypotheses are defined. A qualitative scenario-based evaluation is conducted to assess the defined hypotheses and the obtained results are discussed.

A. Evaluation Setup

1) Physical Hardware Platform and Test Track: For the evaluation of the function moniotr concept, a model vehicle on the scale of 1:8, developed by Digitalwerk [29], is chosen as the physical hardware platform. The model vehicle is equipped with a wide range of sensors, including a mono camera, a LiDAR sensor, wheel speed sensors, ultrasonic sensors, a Global Positioning System (GPS) sensor, and an Inertial Measurement Unit (IMU).

To enhance its environmental perception capabilities for the validation of the function monitor, further sensors have been installed on the model vehicle, e.g., an Intel RealSense LiDAR camera (L515) [30] and a stereo vision camera (D435f) [31]. The LiDAR camera provides sensor input data for the LiDAR AI-based perception component. Although a high-resolution 3D LiDAR sensor would have been ideal, the model vehicle's limited power supply led to the deployment of a LiDAR camera with lower power requirements as a good compromise solution, which still provides adequate data output. Both sensors have been calibrated based on the vehicle's rear axle, to ensure that generated object lists are in the same coordinate system, specifically in the vehicle coordinate system. This alignment is crucial for an accurate and coherent comparison of redundant perception systems. Figure 4 illustrates the model car with the LiDAR camera and stereo vision camera mounted on it.

The test track utilized for the evaluation was constructed in the lab environment using modular martial arts mats. Each black mat measures 1m $\times$ 1m and is adorned with street markings and track walls [32]. Figure 5 depicts the model vehicle placed on the test track along with other objects that emulate other traffic participants, such as a pedestrian represented by a wooden human dummy, and elements of the road infrastructure, e.g., traffic light.

2) Implementation Details: The function monitor conducts a consistency comparison between two object lists, a CAI object list and a LAI object list. These two lists are generated by respective AI-based perception components, one based on camera input and the other on LiDAR input. Both perception components apply YOLO Nano 2D object detectors [33], which yield 2D bounding boxes with object class names and and their respective confidence scores, but lack the distance information between the ego-vehicle and the corresponding objects. By leveraging the LiDAR point cloud provided by the LiDAR camera, we computed the distance between the model vehicle and the objects, referred to as depth, thereby producing 2.5D bounding boxes. The 2.5D bounding boxes differ from the 3D bounding boxes in that the latter include all three dimensions, i.e., height, width and length of the bounding box.

The outputs of the environment and self-perception subsystem along with the result of the function monitor are visualized in the GUI of the remote CCC (see Section III). Figure 7 depicts the visualization of the function monitor result in the Car Selection panel of the remote CCC, utilizing a flag called *AI perception Validator*. The flag's color indicates different results of the function monitor: (1) **green** - denotes consistency between the two object lists, (2) **red** - signifies inconsistency, and (3) **black** - denotes data not being received by the AI Perception Validator component in the function monitor.

In the center of Figure 7, two panels display the view of the LiDAR camera (upper panel) and the stereo vision camera (lower panel) with the bounding boxes corresponding to each object list highlighted in green on their respective sensor view. Additionally, the Sensor Visualization panel depicts the ROI computed around the model vehicle and comprising a focus zone, marked in orange, and a clear zone, marked in green, as introduced in Section III-B. The implementation of each component is based on the decentralized middleware ROS2, facilitating the communication between the ROS2 components through the publish-subscribe pattern. This solution provides advantageous features such as self-adaptation and component reconfiguration at runtime, aligning with the distributed nature of the AV, where components are distributed across different electronic control units (ECUs) [34]. Furthermore, the real-time capabilities of ROS2 make it appropriate for ensuring the safety and reliability of the ADS.

B. Definition of Test Scenarios and Research Hypotheses

To evaluate the function monitor concept, we conducted a qualitative scenario-based assessment. We defined several test scenarios along with working hypotheses to guide the evaluation. The test scenarios range from simple to complex, starting with a single static object and gradually increasing the

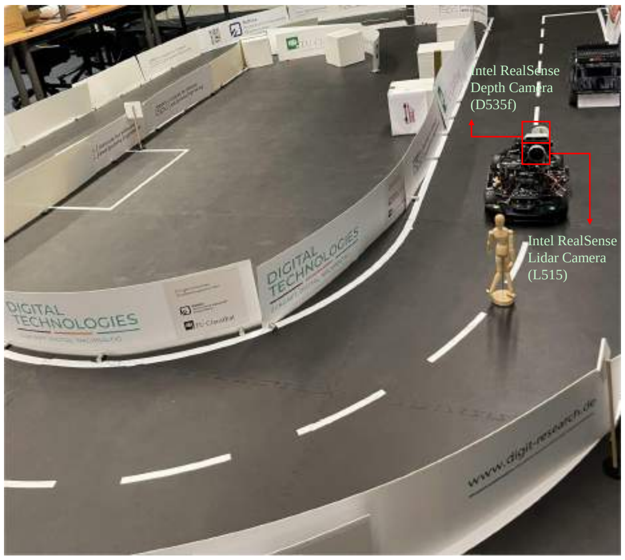

Figure 5: Model Car: bird's-eye view in a Lab Environment.

scenario complexity, by incorporating multiple static objects in a static environment. Each test scenario includes a description of the physical actions of the model vehicle and its environment. Following three test scenarios were defined evaluating the function monitor:

Test Scenario 1 (TS 1). The model vehicle is stationary on the test track and a pedestrian (represented by a wooden human dummy) is placed in front of the model vehicle, outside of its ROI. The pedestrian is placed in such a way that it is detected by the LiDAR camera, but not by the stereo vision camera.

Test Scenario 2 (TS 2). The model vehicle is stationary on the test track and a pedestrian (represented by a wooden human dummy) is placed in front of the model vehicle, inside of its ROI. The pedestrian is positioned in such a way that it is detected by the LiDAR camera, but not by the stereo vision camera.

Test Scenario 3 (TS 3). The model vehicle is stationary on the test track and a traffic light is placed in front of the model vehicle, inside of its ROI. The traffic light is positioned so that both the LiDAR camera and the stereo vision camera are able to detect it.

In addition to the test scenarion, several research hypotheses are formulated in this paper to assess the expected performance of the function monitor. The function monitor has been evaluated in the defined test scenarios with respect to the following two hypotheses:

Hypothesis 1 (H1). The function monitor accurately identifies that the CAI object list and the LAI object list are consistent with each other.

Hypothesis 2 (H2). The function monitor accurately identifies that the CAI object list and the LAI object list are not consistent with each other.

C. Discussion of Results

The evaluation results of the function monitor on hypotheses H1 and H2 in all the test scenarios defined for the evaluation

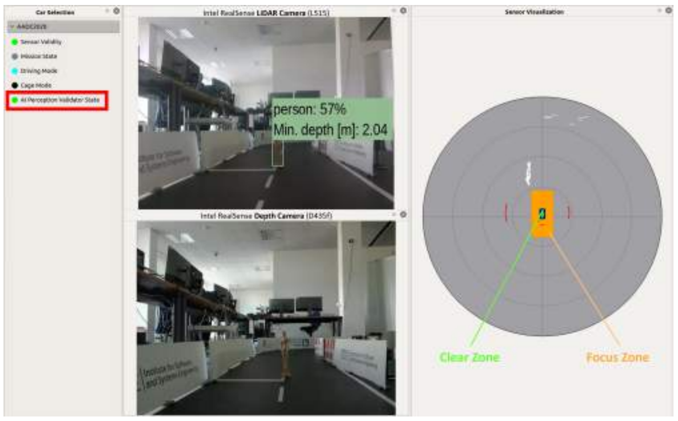

Figure 6: Wooden Human Dummy outside Model Vehicle's ROI.

are presented in Table I. In TS 1, in which the wooden dummy is placed in front of the vehicle and outside of its ROI, the LiDAR camera can detect it but the stereo vision camera cannot. In this scenario, the AI Perception Validator gives the result *consistent* since the wooden dummy is outside of the model vehicle's ROI, and thus, both processed object lists are empty. Therefore, in TS 1, hypothesis H1 is *true* and hypothesis H2 is *false*. The CAI object list and the LAI object list along with the flag of the AI Perception Validator are visually depicted in Figure 6, in which the AI Perception Validator flag shows a green status, indicating "consistent object lists".

TABLE I: EVALUATION RESULTS OF THE FUNCTION MONITOR.

Hypotheses \ Test Scenarios	TS 1	TS 2	TS3
H1	True	False	True
H2	False	True	False

In TS 2, the wooden human dummy is placed again in front of the vehicle, but this time inside its ROI. The human dummy is placed so that it is detected by the LiDAR camera but not by the stereo vision camera. In this scenario, the AI Perception Validator returns *inconsistent*, since there is at least an inconsistent object in either the CAI object list or the LAI object list, which in this case is the human dummy. Thus, in TS 2, hypothesis H1 is *false* and hypothesis H2 is *true*. The CAI object list and the LAI object list along with the flag of the AI Perception Validator are shown in Figure 7. The AI Perception Validator flag shows a red status, indicating "inconsistent object lists" since the objects are inside the vehicle's ROI but not aligned with each other. In TS 3, a traffic light is positioned in front of the model vehicle inside its ROI, so that both the LiDAR camera and the stereo vision camera can detect it. In this scenario, the AI Perception Validator returns *consistent*, since the objects are inside the ROI and were detected by both sensors. Thus, in TS 3, hypothesis H1 is *true* and hypothesis H2 is *false*. The CAI object list and the LAI object list along with the flag of the AI Perception Validator are shown in Figure 8, in which the AI Perception

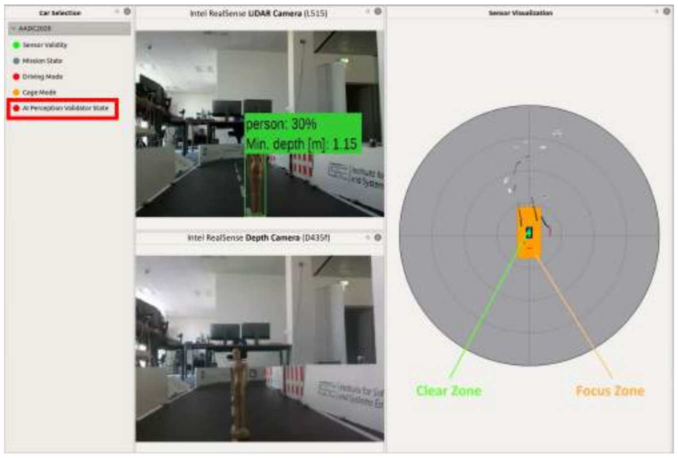

Figure 7: Wooden Human Dummy inside Model Vehicle's ROI.

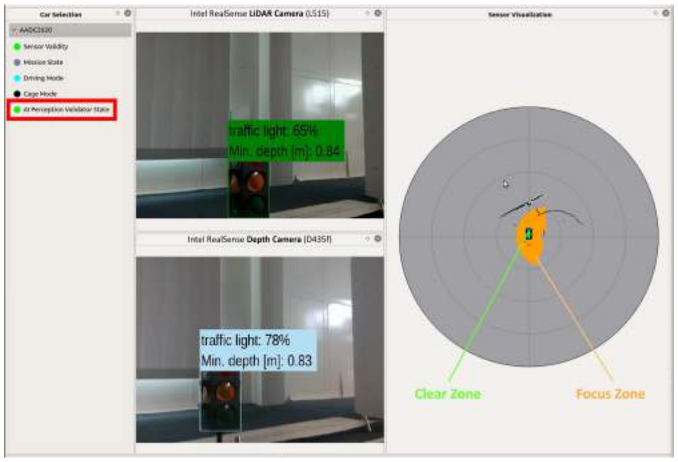

Figure 8: Traffic Light situated Inside Model Vehicle's ROI.

Validator flag indicates a green status, indicating "consistent object lists".

The results obtained in the three test scenarios have confirmed the formulated hypotheses in the defined test scenarios, showing that the function monitor is operating as intended.

V. Conclusion

This paper outlines a method for the runtime monitoring and validation of AI-based environment perception systems employed in autonomous driving contexts. It builds upon the Dependability Cage approach, initially proposed in [22], with a specific focus on the environment and self-perception subsystem in an ADS. The environment perception consists of two redundant perception components tasked with object detection in the ego-vehicle surrounding environment, which leverage multiple sensor data sources, e.g., LiDAR and camera. The dependability cage for the environment perception comprises a function monitor and a fail-operational reaction component. The function monitor checks at runtime whether the outputs of the two perception components remain consistent. Meanwhile, the fail-operational reaction component dictates the fail-safe or the fail-operational reaction of the ADS based on the feedback of the function monitor. This study was primarily focused on the function monitor, which was evaluated qualitatively using predefined test scenarios and a model car in a lab environment. The results of the evaluation demonstrated that the function monitor works as expected.

The test scenarios employed in the evaluation focused on relatively simple driving situations, with stationary objects and a stationary model car. However, in future work, we plan to enhance and extend the functionality of the function monitor so that it covers more complex scenarios, with both dynamic and static obstacles. Moreover, we intend to define a method for defining appropriate fail-operational reactions to gracefully degrade the ADS's functionality [35] in response to warning signals given out by the function monitor. Lastly, we plan to integrate the function monitor with the concept of situation monitor, similar to the one presented in [25]. Such integration enables the ADS to be aware of new object classes detected in its environment, thus enhancing its capability to handle novel environment situations. Ultimately, this integration will contribute to the safety and reliability of autonomous driving systems in diverse and challenging real-world scenarios.

Acknowledgment

The authors express their gratitude for the collaboration of all cooperative partners. This work was carried out as part of the project *safeWahr - Safe release and reliable series operation through continuous real-time monitoring of the environmental perception of autonomous vehicles* [36] under the grant number 19A21026E and was funded by the Federal Ministry for Economic Affairs and Climate Protection [37] in Germany.

References

[1] A. Aniculaesei et al., "Connected dependability cage approach for safe automated driving," in *International Stuttgart Symposium*, pp. 3–21, Springer, 2023.

[2] I. Aslam, A. Aniculaesei, A. Buragohain, D. Bamal, and P. Rausch, "Runtime safety assurance of autonomous vehicles used for last-mile delivery in urban environments," *arXiv preprint arXiv:2307.04454*, 2023.

[3] SAE, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles." SAE International, Warrendale, PA, USA, June 2018. Norm.

[4] S. Behere and M. Törngren, "A functional architecture for autonomous driving," in *Proceedings of the First International Workshop on Automotive Software Architecture* (P. Kruchten, Y. Dajsuren, H. Altinger, and M. Staron, eds.), (New York, NY, USA), pp. 3–10, ACM, 2015.

[5] M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner, *Autonomes Fahren - Technische, rechtliche und gesellschaftliche Aspekte*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.

[6] S. Zeitung, "Tödlicher unfall im selbstfahrenden tesla." online, July 2016. Newspaper.

[7] J. Golson, "Tesla's new autopilot will run in 'shadow mode' to prove that it's safer than human driving." online, Oct. 2016. The Verge.

[8] M. Mauritz, A. Rausch, and I. Schaefer, "Dependable adas by combining design time testing and runtime monitoring," in *FORMS/FORMAT 2014 - 10th Symp. Form. Methods Autom. Saf. Railw. Automot. Syst.*, pp. 28–37, 09 2014.

[9] M. Mauritz, F. Howar, and A. Rausch, "Assuring the safety of advanced driver assistance systems through a combination of simulation and runtime monitoring," in *7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA)*, vol. 9953, pp. 672–687, 10 2016.

[10] ISO, "Road vehicles – functional safety." ISO, Geneva, Switzerland, Dec. 2018. Norm.

[11] ISO, “Road vehicles — safety of the intended functionality.” ISO, Geneva, Switzerland, June 2022. Norm.
[12] “Asam openscenario.” https://www.asam.net/standards/detail/openscenario-xml/, 2022. Norm.
[13] “Asam opendrive.” ”https://www.asam.net/standards/detail/opendrive/, 2023. Norm.
[14] K. Czarnecki and R. Salay, “Towards a framework to manage perceptual uncertainty for safe automated driving,” in *Computer Safety, Reliability, and Security: SAFECOMP 2018 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, Västerås, Sweden, September 18, 2018, Proceedings 37*, pp. 439–445, Springer, 2018.
[15] Q. M. Rahman, P. Corke, and F. Dayoub, “Run-time monitoring of machine learning for robotic perception: A survey of emerging trends,” *IEEE Access*, vol. 9, pp. 20067–20075, 2021.
[16] W. Xiang et al., “Verification for machine learning, autonomy, and neural networks survey,” *arXiv preprint arXiv:1810.01989*, 2018.
[17] F. Leofante, N. Narodytska, L. Pulina, and A. Tacchella, “Automated verification of neural networks: Advances, challenges and perspectives,” *arXiv preprint arXiv:1805.09938*, 2018.
[18] D. Feng, Z. Wang, Y. Zhou, L. Rosenbaum, F. Timm, K. Dietmayer, M. Tomizuka, and W. Zhan, “Labels are not perfect: Inferring spatial uncertainty in object detection,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 9981–9994, 2021.
[19] Q. M. Rahman, N. Sunderhauf, and F. Dayoub, “Per-frame map prediction for continuous performance monitoring of object detection during deployment,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 152–160, 2021.
[20] Q. M. Rahman, N. Sünderhauf, and F. Dayoub, “Online monitoring of object detection performance during deployment,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4839–4845, IEEE, 2021.
[21] A. Gupta and L. Carlone, “Online monitoring for neural network based monocular pedestrian pose estimation,” in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–8, IEEE, 2020.
[22] A. Aniculaesei, J. Grieser, A. Rausch, K. Rehfeldt, and T. Warnecke, “Towards a holistic software systems engineering approach for dependable autonomous systems,” in *Proceedings of the 1st International Workshop on Software Engineering for AI in Autonomous Systems* (R. Stolle, S. Scholz, and M. Broy, eds.), (New York, NY, USA), pp. 23–30, ACM, 2018.
[23] F. Helsch, I. Aslam, A. Buragohain, and A. Rausch, “Qualitative monitors based on the connected dependability cage approach,” vol. 11, no. 21, p. 46 to 55, 2021.
[24] A. Rausch, A. M. Sedeh, and M. Zhang, “Autoencoder-based semantic novelty detection: Towards dependable ai-based systems,” *Applied Sciences*, vol. 11, no. 21, p. 9881, 2021.
[25] N. Habib, Y. Cho, A. Buragohain, and A. Rausch, “Towards exploring adversarial learning for anomaly detection in complex driving scenes,” in *Deep Learning Theory and Applications* (D. Conte, A. Fred, O. Gusikhin, and C. Sansone, eds.), (Cham), pp. 35–55, Springer Nature Switzerland, 2023.
[26] L. Everding et al., “Dynamically configurable autonomous vehicles for urban cargo transportation,” in *Towards the New Normal in Mobility: Technische und betriebswirtschaftliche Aspekte*, pp. 851–869, Springer, 2023.
[27] S. Behere and M. Törngren, “A functional reference architecture for autonomous driving,” *Information and Software Technology*, vol. 73, pp. 136–150, 2016.
[28] Z. Rozsa and T. Sziranyi, “Virtually increasing the measurement frequency of lidar sensor utilizing a single rgb camera,” *arXiv preprint arXiv:2302.05192*, 2023.
[29] “Welcome to digitalwerk - regensburg.” https://www.digitalwerk.net/. (Accessed on 01/31/2024).
[30] “Lidar camera l515 – intel® realsense™ depth and tracking cameras.” https://www.intelrealsense.com/lidar-camera-l515/. (Accessed on 01/31/2024).
[31] “Depth camera d435f – intel® realsense™ depth and tracking cameras.” https://www.intelrealsense.com/depth-camera-d435f/. (Accessed on 01/31/2024).
[32] T. Warnecke, J. Grieser, M. Zhang, A. Vorwald, and A. Rausch, “Teaching novices supervised learning with autonomous model vehicles,” in *2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE&T)*, pp. 1–10, IEEE, 2020.
[33] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, “Yolo nano: A highly compact you only look once convolutional neural network for object detection,” in *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pp. 22–25, IEEE, 2019.
[34] C. Raulf et al., “Dynamically configurable autonomous vehicle concepts for passenger transport,” in *Transforming Mobility–What Next? Technical and business aspects*, pp. 265–288, Springer, 2022.
[35] A. Aniculaesei, J. Grieser, A. Rausch, K. Rehfeldt, and T. Warnecke, “Graceful degradation of decision and control responsibility for autonomous systems based on dependability cages,” in *5th International Symposium on Future Active Safety Technology toward Zero, Blacksburg, Virginia, USA*, 09 2019.
[36] “Safewahr:sichere freigabe und zuverlässiger serienbetrieb durch kontinuierlisches echtzeitfähiges monitoring der umgebungswahrnehmung autonomer fahrzeuge.” https://zdin.de/digitales-niedersachsen/projektubersicht/safewahr, 2021. (Accessed on 11/03/2024).
[37] “Bmkw:bundesministerium für wirtschaft und klimaschutz.” https://www.bmwk.de/Navigation/DE/Home/home.html. (Accessed on 11/03/2024).