Search-based Testing of Vision Language Models for In-Car Scene Understanding

Lev Sorokin
BMW Group, Technical University of Munich
Munich, Germany
lev.sorokin@tum.de

Chen Yang
Technical University of Munich
Munich, Germany
c.yang@gmx.net

Ken E. Friedl
BMW Group
Munich, Germany
ken.friedl@bmw.de

Andrea Stocco
Technical University of Munich, fortiss GmbH
Munich, Germany
andrea.stocco@tum.de

Abstract

In the automotive domain, in-car scene understanding (ISU) enables the detection of safety-critical events, such as driver distraction, and supports drivers or passengers by analyzing the in-car scene and adapting the environment (e.g., ambient lighting). The industry is increasingly exploring vision-language models (VLMs) to interpret camera-recorded in-car scenes and extract information for downstream reasoning tasks. However, VLMs may generate incomplete, erroneous, or misleading scene descriptions, highlighting the need for systematic testing. Collecting real in-vehicle data is costly, difficult to scale, and often infeasible, particularly in early design stages. In this paper, we present ISU-Test, an automated testing approach that combines rendering-based scene generation with search-based testing to evaluate ISU systems. By framing testing as an optimization problem and systematically modifying scene parameters, our method generates diverse in-car scenarios and explores a wide range of configurations. We evaluate ISU-Test on both an industrial prototype and open-source VLMs across two case studies: question answering and captioning, comparing against randomized scenario generation. Results show that ISU-Test significantly outperforms the baseline, achieving up to 10× higher failure rates and up to 3.6× higher coverage.

1 Introduction

In the automotive domain, in-cabin monitoring systems play a crucial role in the detection of safety-relevant events, such as driver distraction. Furthermore, it can be used to set up entertainment or comfort features, providing a personalized experience to control, e.g., music and light functions, or helping passengers to locate objects in the car [1–3]. In addition, regulatory frameworks such as the European Union's General Safety Regulation (GSR) [4], and consumer safety assessment programs like Euro NCAP [5] increasingly mandate the deployment of driver monitoring functions, including distraction and drowsiness detection. These systems must operate reliably under a wide range of real-world conditions, making systematic testing and validation a critical requirement for certification and deployment.

A viable solution to implement in-cabin monitoring systems is to employ vision language models [1, 3] to interpret 2D camera images from the vehicle's cabin and provide structured scene information for subsequent reasoning. Despite their capabilities, such models remain susceptible to providing incorrect or incomplete scene descriptions, highlighting the need for rigorous testing [6–8]. Obtaining real-world in-cabin data is costly, difficult to scale, and often infeasible, especially during early design stages when vehicle interiors are not yet available for comprehensive data collection. While manually recorded datasets exist [9], these are limited to specific in-cabin scenarios and have no controllability or diversity. In addition, static datasets are likely part of the training data of VLMs, limiting their usage for testing and validation practices.

Figure 1: Rendered scene in a vehicle where a loose suitcase is placed in the back, and the driver's seat belt is not fastened.

In this paper, we present an automated testing approach that combines controlled and rendering-based image generation with search-based testing to identify scenes where the monitoring system provides incorrect detection. This work addresses the lack of controllable and systematic validation techniques for VLM-based in-car scene understanding systems. In particular, our work focuses on the processing of single 2D images. Unlike prior work on testing DNN-based perception systems, our approach targets VLM-based scene understanding, where outputs are semantic and open-ended, requiring novel fitness and oracle definitions.

Our approach is based on the following steps. First, we define the features on which the system needs to be tested. In our example, this includes driver features such as emotion, pose, gender, interaction with objects, static objects, and the car model. For passengers in particular, we employ SMPL-X [10], which allows us to model humans parametrically according to height, weight, and pose. For environmental parameters, such as light, we model a light source of different intensity levels and use a captured panoramic photo

to define the external scene. In the second step, we parametrize the scene by (a) altering the positions of objects, e.g., placing and positioning luggage on the back or front seats, (b) altering the existence of objects, e.g., having the seatbelt attached or detached, and (c) adjusting the driver's pose. The parametrization allows us to control the generation of diverse scenarios and the controlled evaluation of scenario descriptions provided by the system under test. Finally, we use search-based optimization to find failures. The optimization algorithm iteratively generates new scenes by modifying features of previous scenes guided by the evaluation results of previous executions. We evaluate our approach by testing a public and industrial prototype on the two use cases, visual question answering and captioning with multiple VLMs. The results show that ISU-Test can successfully and efficiently identify failures in ISU systems, outperforming random test generation approaches.

The contributions of this paper are as follows:

Framework. A modular testing framework (ISU-Test) that combines controllable scene generation with search-based optimization to systematically expose failures in VLM-based ISU systems.

Techniques. Novel fitness and oracle definitions for evaluating both structured (VQA) and open-ended (captioning) outputs of VLMs.

Evaluation. An extensive empirical study on public and industrial systems demonstrating up to 10× higher failure rates and improved failure diversity compared to random testing.

2 Background

In the following, we provide the definitions and illustrative examples required to understand our work. We begin with the definition of the system under evaluation and the testing problem.

Definition 2.1. An in-car scene understanding system (ISU) is a vision-language model (VLM)-based system that takes as input an image S, referred to as the *scene*, and produces a textual representation describing relevant aspects of that scene.

Depending on the interaction mode, the output can take different forms. In *visual question answering* (VQA), the system generates a structured output in form of a concise answer to a given query about the scene. In *visual captioning* (VC), the system produces an unstructured natural language description summarizing the scene content without a specific query.

Example 2.1. Consider the scene depicted in Figure 1, where a smiling female driver is sitting next to a baby in a car seat, and a yellow luggage is visible in the backseat. An ISU can extract information such as the driver's emotions, gender, or clothing color by prompting the system in a structured manner.

For example, a structured output for a given prompt, such as Figure 2, could be: R = ("gender" : "female", "seat-belt" : "off", "baby_on_board" : "true"). In contrast, if the mode is visual captioning, the system is asked to provide an unstructured output to describe the scene. For instance, given the prompt *"Describe what is visible in the scene, paying attention especially to humans and loose objects"*, a possible response could be: *"The car shows a smiling female driver sitting next to a baby, holding a phone in her hand."* Unstructured outputs are valuable because they allow the model to provide richer and more flexible information, including details not anticipated in a predefined schema, which can be leveraged in downstream natural language applications.

```
You are an image analysis assistant to evaluate in-cabin scenes.
Your task is to answer the following
questions about the visual scene and return the result in strict
JSON format.


# Rules
- Return only the JSON.
- Do not provide explanations.
- For each question, choose exactly one answer from the given
ànswer_options›
- Do not create new answers, only choose from the provided
options.
- If a question references something not present, select the
provided “None”.


# Questions
question: "Is the driver male or female?"
answer_options: ["MALE", "FEMALE"]


question: "What emotion is the driver expressing?"
answer_options: ["HAPPY", "SERIOUS"]


...
```

Figure 2: (VQA) Prompt used for a VLM in an in-car scene understanding system to describe the cabin scene.

VLM-based systems can make inaccurate evaluations or fail in different ways to provide appropriate scene descriptions: they can wrongly classify the driver's emotions or not detect a baby in the car. Also, they could hallucinate and provide the incorrect value in the structured description or unstructured summary of the scene. Failures can happen because of low-light conditions (dark in- and outside), partially occluded objects such as holding phones or luggage in the back, or the complexity or number of objects in the car to detect.

Definition 2.2. A search-based testing problem for an ISU-System ISU is defined as a tuple $P = (ISU, D, F, O)$, where

- ISU is the in-car scene understanding system; the system under test.
- $D \subseteq \mathbb{R}^n$ is the search domain, where n is the *dimension* of the search space. The vector $\mathbf{s} = (x_1, \ldots, x_n) \in D$ represents the discretized representation of the scene passed to ISU.
- F is the vector-valued fitness function defined as $F : D \mapsto \mathbb{R}^m$, $F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$, where each f_i is a scalar fitness function that assigns a quantitative score to an input based on its ability to expose faulty VLM behavior. The objective space $\mathbb{R}^m$ corresponds to the number of evaluation criteria.
- O is the oracle function, $O : \mathbb{R}^m \mapsto \{0, 1\}$, which decides whether a generated scene reveals a failure. An input for which $O(F(\mathbf{x})) = 1$ is considered *failure-inducing*.

In the following, we present our testing approach based on search-based testing to evaluate the capabilities of the ISU in providing correct scene descriptions.

3 Approach

Our testing approach is illustrated in Figure 3 and receives the following inputs: a set of features $\mathcal{F}$ describing scenes with feature domains $D_{\mathcal{F}}$, a fitness function F, an Oracle O, a simulator S, a population size n, and the ISU as system under test. In addition, hyperparameters, such as the testing budget, are defined.

ISU-Test is characterized by the following stages: Test Space Definition, Static Scene Preparation & Generation, Execution and Evaluation, and Optimization. Search-based testing is particularly suited for this setting, as the input space is high-dimensional, structured, and constrained, making exhaustive or purely random exploration inefficient. In the following, we explain each step in detail.

3.1 Scene Specification and Representation

Initially a set of *scene* features $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n\}$ with domains $\mathcal{D}_{\mathcal{F}}$ is defined to parametrize the possible space of scenes. Further $\hat{\mathcal{F}}$ is defined, while $\hat{\mathcal{F}} \subseteq \mathcal{F}$ is called the set of *predicted* features. Predicted features are used in the evaluation as later explained in Section 4 to compare the system's output with the scene input. A feature $\mathcal{F}_i$ can be numerical, categorical, or ordinal. For instance, the features for Figure 1 can be defined by $\mathcal{F}_{seat}$, $\mathcal{F}_{gender}$, $\mathcal{F}_{baby}$. The domain $\mathcal{F}_{seat}$ is categorical and for instance $D_i = \{yes, no\}$. A continuous feature would, for instance, model the driver's weight, while an ordinal would model the illumination ranging the values, low, medium, and high. A scene vector is defined as a set of concrete feature values $x_i \in F_i$, $F_i \in \mathcal{F}$. For instance, the scene vector for the scene in Figure 1 would be $x = \{belt : false, baby : yes, luggage : yes, \ldots\}$.

3.2 Scene Sampling

The first step of ISU-Test consists of random sampling of feature vectors. Hereby, we map all feature values based on the type, whether they are ordinal, numerical or continuous to a numerical space as proposed by Sorokin et al. [11]. I.e., all values are mapped to the range [0,1] based on min-max normalization, Further, constraints have to be constrained to generate valid scene vectors. A possible constraint restricts, for instance, that a *luggage_color* is defined while no luggage exists in the scene.

3.3 Scene Generation

The scene generation consists of the rendering engine/simulation-based generation of the scene and the static scene setup. The static scene setup is to be performed only once, where scene elements are modeled and parametrized to be used in connection with the features defined. Hereby, we distinguish between *indoor* assets such as vehicles cabin, luggage, bottle, baby seat, and *driver-oriented assets* like the drivers pose, emotion, appearance, the *adaptive assets* such as the seat-belt which should adopt to the human body shape for realistic modeling, and *environmental*-related such as the lighting in the car, external lighting or the external view around the cabin/car.

Objects. Objects are provided through CAD models, and parametrized with respect to their position or orientation in the cabin. The granularity of the parameterization is use case specific.

Driver-oriented assets. For human modeling, we leverage SMPL-X [10], which is based on a data-driven parameterization of realistic human body shapes. The underlying model of SMPL-X is trained with a dataset of thousands of 3D scans of humans of different genders, poses, ages, and emotions in different setups (e.g., seating, standing). We retrieve two models for the different genders and manually adapt the models to fit in the cabin. The SMPL-X technique allows us to automatically generate different body shapes of the driver. To model the emotional state, we tune the coefficient of several shapekeys on the SMPL-X human's face. To model the clothing of the driver, we manually provide different overlays based on the parametrization of different overlays in the desired color and type of clothing.

Adaptive assets. For adaptive modeling, such as seat belt modeling, we employ surface modeling and apply a shrink-wrap modifier [12] to position the seat belt over the passenger's body while avoiding intersections with the body mesh.

Environmental assets. Environmental lights are modeled by providing functions that set the internal state of the light source in the simulator scene, where the lighting intensities depend on the simulator's rendering engine. For the external view, we record first an image of the exterior of the car in the selected environment in the HDRI format.

Novel Scene Synthesis. For a given scene vector, all feature values are forwarded based on the corresponding feature type to the respective function to set up the scene component based on the linking from the previous step. When all features have been applied, the rendering of the image is triggered, producing an image of the in-cabin scene.

Fitness Function Definition. For the fitness function, we distinguish between *response-related* and diversity-optimization related fitness functions.

The **response-oriented** fitness function assesses the quality of the response and receives as input the set of prediction features $\hat{\mathcal{F}}$ and the system response R and outputs a score between 0 and 1, where 0 is the worst score, and 1 is the best score. The implementation of the fitness function depends on the execution mode of ISU.

1) Question Answering. For question answering, the fitness function compares the values of predicted features in the input vs. the values of those in the output for exact matches, i.e., it is defined as follows:

$$F_{\text{qa}} = \frac{1}{|C|} \sum_{F_i \in C} w_{F_i} \cdot \mathbf{1}\{\hat{f}_i = f_i\}.$$

where C is the set of features, and w_{F_i} is a so-called feature weight defining the importance of the detection of the corresponding feature in the scene. Weighting allows us to penalize incorrect outputs for features that are more important than others, e.g., those related to the driver or a child in the car.

2) Visual Captioning. For captioning evaluation, we first generate a reference caption by instantiating a parametrized text template with the scene vector X. An example template is *The scene shows a **gender** driver with **t-shirt color** t-shirt having the emotion **emotion**.* The evaluation of the output scene description is performed by evaluating the similarity between the reference caption and the generated caption, i.e., we apply the following four metrics to capture different evaluation dimensions and assess the generated scene description:

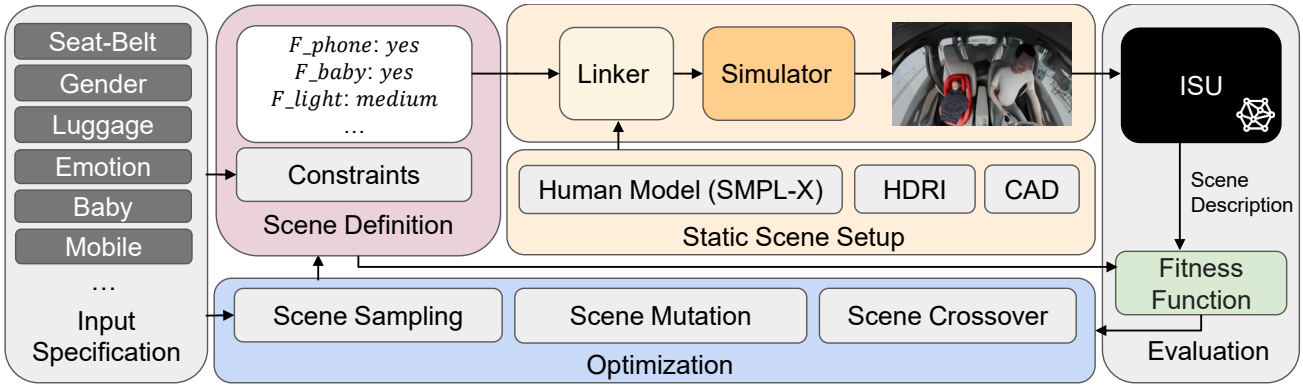

Figure 3: Overview of ISU-Test.

Embedding Similarity (F_{emb}): The embedding similarity captures the description semantics on the sentence level. We embed the reference and predicted captions using a text encoder, i.e., `all-MiniLM-L6-v2`, and compute cosine similarity between the encodings.

BLEU Score (F_{bleu}): BLEU measures the n-gram overlap between generated and reference text and allows us to assess the lexical alignment and local phrase correctness [13].

METEOR score (F_{meteor}): METEOR focuses on the precision, recall, and word order with stemming and synonym, balancing strict overlap metrics such as BLEU [14].

BERTscore (F_{bert}): BERTScore provides contextual semantic alignment at the token level. Unlike sentence-level embeddings, BERTScore computes semantic similarity via contextual token embeddings and cosine similarity [15].

The **diversity-oriented** fitness function F_{div} evaluates the diversity of a scene compared to previously generated scenes by measuring for non-continuous features, its Hamming Distance to other scenes [16], and for continuous features, the actual normalized Euclidean distance. In particular, for two scenes s and s' the function is defined as

$$F_{\text{diversity}}(s, s') = \sum_{i \in \mathcal{D}} \mathbf{1}\left[f_i \neq f_i'\right] + \sum_{i \in C} \frac{|f_i - f_i'|}{\Delta_i}, \tag{1}$$

where $\mathcal{D}$ are discrete and C continuous features.

Oracle Function. The oracle function defines when a test is failing and is defined in a use case specific. For the question answering-based mode, it is defined as:

$$O(test) : F_{qa} \leq th$$

while for the captioning-based mode it is defined as:

$$O : (F_{\text{emb}} \leq th_{emb}) \wedge (F_{\text{bleu}} \leq th_{bleu}) \wedge (F_{\text{meteor}} \leq th_{meteor}) \wedge (F_{\text{bert}} \leq th_{bert})$$

where th, th_{emb}, th_{bleu}, th_{meteor} and th_{bert} are thresholds.

Optimization Algorithm. All defined fitness functions are used by the optimization algorithm to rank executed tests based on a score to guide the generation of new scenes. While the response-oriented fitness function/s is/are to be minimized to prioritize tests that fail for guiding the generation of new scenes, the diversity fitness function is to be maximized to foster detection of diverse scenes.

In our study, we employ a genetic population-based algorithm to guide the generation of failure-revealing scenes. In particular, the algorithm applies genetic operators such as mutation or crossover to generate promising scenes.

Scene Mutation and Crossover. The mutation and crossover operators allow the generation of new scenes from existing scenes. While the mutation operator modifies single feature values, e.g., changing the seat belt from on to off, or lighting medium to low, the crossover operator recombines features of two scenes to generate two new scenes. The recombination of two scenes can lead to scenes where features become incompatible. We impose, therefore, scene constraints after generation, such as those applied after scene sampling and selecting predefined feature values without conflicts. An example restriction would be placing both luggage and a child in the same place. If a scene vector is generated, which has already been witnessed before, a new scene is randomly sampled (duplicate elimination).

Survival. In this step, a fixed number of scene vectors/candidates is selected based on the fitness values to be taken over to the next iteration of the algorithm. This step is necessary as otherwise the set of test cases to be executed would always grow, and candidates would remain in the population with poor fitness scores.

When the testing budget is exhausted, ISU-Test invokes the oracle to classify the generated conversations based on their fitness values and outputs the set of failing test cases during the search.

4 Evaluation

Our evaluation considers the following research questions:

4.1 Research Questions

RQ$_1$ (effectiveness). *How effective is ISU-Test in identifying failures?*

RQ$_2$ (diversity). *How diverse are failures identified by ISU-Test?*

RQ$_3$ (validation). *What is the validity rate of identified failures when comparing rendered scenes of found failures to real-world scenes?*

Table 1: List of test subjects used in our study.

Category	Model	Version	Case Study
OpenAI	GPT-5-Chat	2025-04-01	VQA, VC
Google	GEMINI-2.5-Flash	via VertexAI	VQA, VC
OpenSource	MoonDream2	2025-06-21	VQA, VC
Industrial	ISU	2025	VQA
Industrial	ISU-Flash	2025	VQA

In the first place, we want to evaluate with RQ_1 how effective our approach is in finding failures, which is the primary goal when performing automated testing.

However, the failing scenes identified may be highly similar, for instance, when scenes differ only by a minimal change in the orientation of a piece of luggage. Prior research has emphasized [17, 18] the importance of identifying diverse failures to support debugging and fault localization. Therefore, in RQ_2, we evaluate the diversity of failures detected by ISU-Test.

In RQ_3, we aim to evaluate the extent to which the system under test produces consistent description outcomes when provided with a reconstructed real-world scene in place of a simulated scene.

4.2 Study Subjects

We evaluate our approach on the case studies of visual question answering (VQA) and visual captioning (VC). In VQA, a scene is provided to the system with the goal of retrieving a structured description of the scene with a predefined format (features predefined). In visual captioning, the system under test has to generate a caption summarizing the scene.

For VQA, we evaluate three standalone general-purpose VLM models from three different providers, namely GPT-5-Chat, Gemini-2.5-Flash, and MoonDream2, a small-sized VLM of size 2B, and two industrial prototypes from our partner company BMW. The industrial systems are available in two variants, *ISU − Flash* and *ISU*, where in the first version a more cost-efficient and lightweight node is used than in the second variant. Both variants natively output a structured JSON dictionary in the required format.

For general-purpose VLM models, we instruct the model with the prompt as shown in Figure 5.1. The complete prompt can be found in our supplementary material [19]. The selection of the models is guided by the different capabilities, pricing, size, and inference latency of the models. For the industrial system, we adopt the prompts to capture all features modeled in our scenes.

Table 1 provides an overview of all evaluated configurations.

4.3 Metrics

RQ_1. We evaluate the effectiveness of ISU-Test by measuring the number of failing scenarios identified and the ratio between failures and the number of scenes produced.

RQ_2. To evaluate the diversity of found failing scenes, we first map scenes applying CLIP embeddings into a numerical space from all runs and cluster the mapped scenes. Then we apply k-means-based clustering to retrieve a finite set of failure clusters. Then, for each approach, we evaluate coverage of the approximated clusters as well as the failure distribution among the clusters as proposed by Biagiola et al. [17].

RQ_3. To evaluate how well the generated failing test inputs correspond to actual failing tests, we first sample for each testing approach over all runs and recreate a portion of the scenarios in reality. In the first step, both the rendered images and the recreated real-world scenes are presented to human evaluators, who are asked to assess whether they depict the same scenario. In the second step, the performance of the system under test is evaluated using both the simulated images and the recreated scenes.

Table 2: Overview of features used in our study.

Cat.	Feature	Values
Driver	Gender	Female, Male
	Emotion	Happy, Serious
	Phone Pos.	None, P1–P10
	Weight	W1–W4
	Height	H1–H4
	Seatbelt	Y/N
	T-shirt	Black, White
Child	Seat Exists	Y/N
	Child in Seat	Y/N
	Orientation	Front, Back
Env.	Car Lights	On/Off
	External Light	High, Medium, Low
Objects	Luggage Exists	Y/N
	Orientation	20–140
	Position	Passenger, Back
	Colour	Yellow, Anthracite
	Phone Exists	Y/N
	Coke Exists	Y/N
	Can Exists	Y/N

Table 3: Feature weights used in the VQA case study.

Features	Weight
Gender, Emotion, Cola Bottle, Cola Can	0.05
Phone, Seat Belt, Luggage Existence, Luggage Location, Codriver Phone, Child Seat, Child Seat Orientation, Child Existence	0.10

4.4 Experimental Configuration VQA

For VQA, we select the features as defined in Table 2. The features and their parametrization have been first derived based on studies from relevant use cases available in the literature [1, 20] and then confirmed and refined with experts from the industrial partner.

4.4.1 Fitness Function. The fitness evaluation for VQA consists of two fitness functions: F_{cat}, and $F_{\text{diversity}}$. While the first is minimized to find failing tests, the second is to be maximized to find more diverse scenes. The weighting of features is illustrated in Table 3 and has been defined based on discussions with BMW experts. The weighting can be adopted based on the use cases and is chosen

Table 4: Failure rate (%, mean ± standard deviation) for VQA and VC across SUTs under ISU-Test and RS.

Thr.	GPT-5-Chat		Gemini-2.5-Flash		MoonDream2		ISU		ISU-Flash	
	ISU-Test	RS	ISU-Test	RS	ISU-Test	RS	ISU-Test	RS	ISU-Test	RS
VQA										
Tests	606.7±10.3	560.3±7.9	563.3±15.1	538.2±10.2	286.7±10.3	200.0±16.0	490.0±11.0	477.7±6.1	606.7±30.1	570.3±29.1
1.0	**88.2**±7.5	71.7±0.8	**60.0**±8.4	38.2±2.5	100.0±0.0	100.0±0.0	**66.5**±8.5	37.2±2.4	**61.7**±7.6	30.0±1.1
0.9	**46.5**±7.2	18.7±1.2	**29.7**±2.6	4.2±0.8	100.0±0.0	100.0±0.0	**32.7**±6.1	12.2±1.7	**23.2**±3.4	6.3±1.2
0.8	**10.5**±3.6	1.8±0.4	**15.2**±2.9	1.5±0.5	100.0±0.0	100.0±0.0	**20.8**±5.2	5.7±0.5	**14.2**±2.5	2.3±0.5
0.7	**3.8**±2.1	0.0±0.0	**1.7**±2.0	0.0±0.0	**98.0**±1.5	95.5±1.9	**5.0**±2.8	0.2±0.4	**2.5**±1.5	0.0±0.0
0.6	0.0±0.0	0.0±0.0	**0.5**±0.5	0.0±0.0	**90.2**±5.4	80.5±8.3	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
0.5	0.0±0.0	0.0±0.0	**0.2**±0.4	0.0±0.0	**64.0**±7.2	52.5±20.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
VC										
Tests	**180.0**±6.3	124.2±12.2	**166.7**±10.3	113.3±17.1	**156.7**±5.2	88.8±3.1	–	–	–	–
0.8	**99.7**±0.8	99.2±0.8	**96.0**±2.0	87.5±4.2	100.0±0.0	100.0±0.0	–	–	–	–
0.7	**88.0**±6.2	72.3±4.6	**58.3**±6.2	27.5±4.5	**96.2**±4.2	92.3±8.7	–	–	–	–
0.6	**36.3**±12.1	13.8±3.5	**19.3**±4.5	4.0±2.1	53.7±25.2	**54.5**±39.4	–	–	–	–
0.5	**3.5**±2.9	0.5±0.5	**2.2**±2.6	0.0±0.0	12.8±14.4	**13.3**±14.8	–	–	–	–

in a way to penalize safety-related features more than non-safety-related (like finding small objects in a car).

4.4.2 Oracle Function. For VQA, a test is classified as failing if at least one feature classification is incorrect, i.e., $O : F_{\text{qa}} < 1.0$. We additionally report results under relaxed thresholds to study the failure severity when considering the custom weighting of features.

4.5 Experimental Configuration VC

VC is a canonical open-ended VLM task [21, 22] and complements the VQA case study [23]. Unlike VQA, VC does not admit a single correct answer [24], which makes oracle construction inherently difficult. Correctness is therefore typically assessed using similarity-based metrics rather than exact matching [25, 26]. We use this case study primarily to study the robustness of our approach when using different processing techniques in the SUT. We use the same search budget, the same hyperparameter configuration, and the same features as defined for VQA.

4.6 Fitness

To evaluate the fitness, we use four functions as defined in Section 3.3, including the diversity-oriented function. We provide a predefined template which is filled as explained in Section 3.3 with values of the corresponding feature vector to generate a reference caption. An example reference prompt for the scene vector is provided in Figure 4.

4.7 Oracle

For the oracle, we only apply a threshold for the first fitness function F_{llm} and use a threshold of 0.65. We select the threshold value empirically based on preliminary experiments.

```
The image is taken from inside a car parked in a garage. A happy
male driver wearing a black T-shirt is seated in the car. The
driver is not wearing a safety belt. A anthracite suitcase is
visible on rear seat. A rear-facing baby seat with a baby is
positioned on the front passenger seat.
```

Figure 4: Caption derived from ground-truth parameters. Blue text segments are injected from the scene vector.

4.8 Results Effectiveness (RQ_1)

The effectiveness results comparing ISU-Test with random search (RS) are shown in Table 4. We can see that, for both case studies, ISU-Test achieves for the majority of compared systems and threshold higher scores than the randomized baseline approach.

In particular, for the VQA cases study ISU-Test achieves a threshold of 1 to 0.7 for all comparisons, with higher failure rates for the open-source as well as the industrial case study. A less strict oracle with a lower threshold below 0.7 allows only for the more efficient but smaller models Gemini-2.5-Flash and MoonDream2 to detect failures. Also, for this configuration, setups ISU-Test still outperform the random baseline in the failure discovery. However, the highest failure rates are found in GPT-5-Chat and MoonDream2 for the most strict threshold of 1.0.

For the VC study, we can observe that for thresholds between 0.9 and 1, there is no difference between the approaches, exhibiting 100% failure rates. This strict comparison rather considers expression variations and likely does not take semantic variations into account, making it not suitable for comparison of the systems' scene evaluation behaviour. For the thresholds 0.8 to 0.5, we can see that ISU-Test outputperforms the randomized baseline for the majority of comparisons (11 out of 12).

Table 5: Diversity analysis under CLIP encoding for VQA and VC. Each cell reports Coverage (%) and Entropy.

Alg.	GPT-5-Chat		GEMINI-2.5-Flash		MoonDream2		ISU		ISU-Flash	
	Coverage	Entropy	Coverage	Entropy	Coverage	Entropy	Coverage	Entropy	Coverage	Entropy
VQA										
ISU-Test	100.00	**97.48**	100.00	**92.84**	100.00	89.18	**96.25**	**94.80**	**97.52**	94.49
RS	100.00	91.52	100.00	91.37	100.00	**91.74**	96.18	94.44	95.37	94.46
VC										
ISU-Test	**82.61**	**92.52**	**86.35**	**91.61**	**79.31**	**91.67**	–	–	–	–
RS	37.83	80.42	23.74	68.52	73.64	92.05	–	–	–	–

In addition, we have performed the statistical Wilcoxon test [27] and evaluated the Vargha-Delaney effect size [28] to assess whether the differences are statistically significant. The results show that all differences are statistically significant with large effect sizes. Further results showing the number of failures over time can be found in the replication package [19].

RQ_1 (Effectiveness). ISU-Test consistently identifies a substantially larger number of failures and achieves higher failure rates than random testing across most configurations in both case studies. In visual question answering, the greatest improvement is observed with a Gemini-based VLM, showing up to a 10× higher failure rate. For captioning, the same system exhibits the largest gain, with failure rates up to 5× higher than those obtained through random testing.

4.9 Results Diversity (RQ_2)

The diversity results are presented in Table 5, which shows both the coverage of failure clusters and the entropy. We can see that for VQA, the coverage for ISU-Test and RS are maximum and equally high for the open source systems, while for the industrial system, the values are slightly higher for ISU-Test. Regarding entropy, we see that for four out of five comparison values for ISU-Test are higher. For visual captioning, we can see that both coverage and entropy are always higher for ISU-Test than for RS, reaching up to 3.6× higher coverage and 34% higher entropy than RS.

The statistical test results show that the difference in entropy is for both case studies, and that for VC also the coverage difference is statistically higher with large effect sizes.

RQ_2 (Diversity). ISU-Test consistently achieves higher entropy with a maximal increase of 34%, indicating a more balanced distribution of failures across failure clusters in both case studies. At the same time, coverage values for VQA are maintained at levels comparable to randomized testing. For VC, optimization-based testing further improves coverage up to 3.6× compared to randomized testing.

4.10 Results Validity (RQ_3)

To study the agreement between test executions with rendering-based generated scenes from ISU-Test and scenes in real life, we recreate a portion of rendered scenes in a real vehicle and curate a dataset of 53 image pairs. The dataset is sampled randomly from both failing and passing tests generated during runs of ISU-Flash. Data collection involves three participants with different but feature-consistent body characteristics modeled in ISU-Test regarding weight and height, including two males and one female. Static assets such as the mobile phone, cola can, door keys, and baby seat are provided as real objects. For the baby, a baby doll is used. The total recreation process requires up to 6 hours because of the number of features and the type of physical assets varied.

An example recreation is shown in Figure 6. In the first step, we assess whether the reconstructed/real scenes correspond *semantically* to the simulated scenes. For this, we employ three human annotators who were not involved in the development of ISU-Test and decide on the alignment for each scene based on majority voting. The evaluation shows a reconstruction validity rate of 100%.

In the second step, to evaluate the system's behaviour when using real scenes, we pass both the real and reconstructed scenes to ISU-Flash. We select here ISU-Flash as it provides lower failure rate scores in the previous analysis compared to ISU. To take into account the stochastic nature of ISU-Flash, we pass each scene three times and select the majority of the predicted labels per feature. Outcomes are categorized into four cases: (i) both predictions correct, (ii) both incorrect, (iii) correct only for the synthetic image, and (iv) correct only for the real image (see Figure 5). The agreement rate is then defined as:

$$\text{Agreement Rate} = \frac{\text{Sim/Real Correct + Sim/Real Wrong}}{\text{All Features Compared}}.$$

The evaluations show an agreement rate of 89%. We observe that especially driver-related features such as *emotion detection*, *phone existence*, and *gender* are robust across both domains (disagreement 2%), while the scenes including luggage in the back or when the driver is belted wearing a black shirt are often misclassified. This could be attributed to the fact that, on the one hand, luggage is partially obscured by the front seats, making the correct recognition difficult. On the other hand, a black belt in simulation can become overlooked because of a similar color appearance.

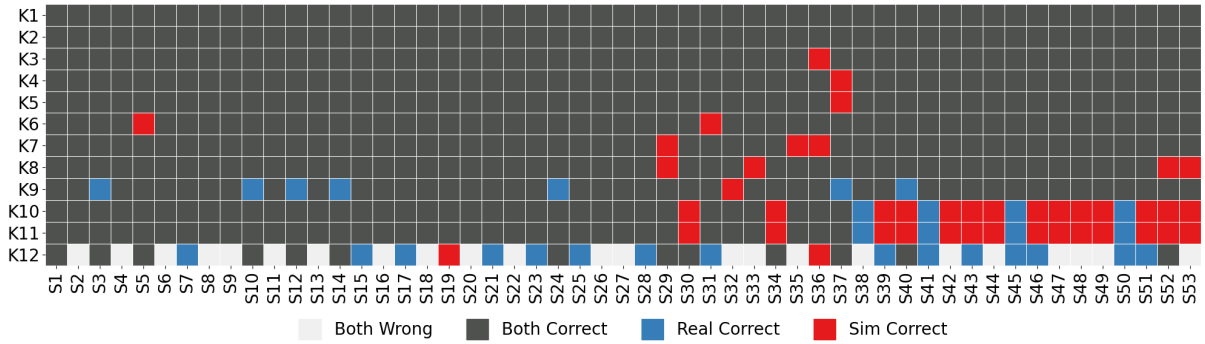

Figure 5: Agreement between real and simulated evaluations. Features: K1 phone_codriver_seat, K2 gender, K3 baby_seat, K4 colacan_codriver_seat, K5 colabottle_codriver_seat, K6 emotion, K7 baby, K8 phone_driver, K9 safety_belt, K10 suitcase_location, K11 suitcase, K12 baby_seat_orientation. Samples are indexed by S1–S53.

RQ_3 (Validity). ISU-Test generated images show a validity rate of 89% when comparing the system performance on simulated vs. recreated scenes.

Key Insights. Our results reveal three main insights. First, optimization-based testing is substantially more effective than random exploration in identifying failure-inducing scenarios, particularly under strict correctness thresholds. Second, while failure coverage for VQA saturates quickly, entropy analysis shows that ISU-Test explores a more balanced and diverse set of failure modes. Third, the high agreement between simulated and real-world evaluations suggests that simulation-based testing can serve as a reliable proxy for early-stage validation of VLM-based systems.

4.11 Threats to Validity

Internal Validity. To assess the validity of oracle results, we evaluated the non-determinism of the system under test by repeating identical inputs and measuring output consistency. The system achieved over 90% consistency across runs, indicating limited variability. Therefore, no additional mechanisms to control non-determinism were applied during the search process.

Construct Validity. Human–object interactions (e.g., a driver holding a mobile phone) are modeled using a simplified animation-based approach instead of a physics-based model, prioritizing controllability over realism. This may limit interaction fidelity and pose a threat to construct validity, particularly for tasks requiring fine-grained physical cues.

External Validity. The approach is evaluated on in-car scene understanding for a single car model. It can generalize to other domains and models whose scenes are composed of assets that can be represented in 3D and rendered using engines such as Maya or Blender—for example, activity monitoring in enclosed environments or the aircraft domain. Fitness and oracle functions need to be defined for each specific use case; however, the overall framework (e.g., weighted averages over prediction matches or threshold-based oracles) can remain unchanged.

The light exposure alignment in our study is tailored to a specific in-car camera and replicates its adaptive exposure behavior in the rendering pipeline. As exposure and calibration are camera-dependent, transferring the approach to other sensor setups may require additional adaptation. Regarding the scene parametrization, feature values have been selected in a way to be able to explore diverse scenes, balancing the possibility for recreation and failure validation. Feature values can span different and bigger search intervals for other applications. A further limitation concerns the abstraction of the feature space, which may not fully capture complex dependencies between features (e.g., correlated driver behaviors and environmental conditions), potentially limiting realism in certain edge cases. This suggests that the approach is applicable to other domains involving structured visual environments, such as robotics, surveillance, or human–machine interaction systems.

5 Qualitative Analysis

5.1 Improving Transferability to Real World

To study how data transformation techniques could further improve the quality of the generated scenes, we applied data transformation as a postprocessing step, specifically employing Neural Style Transfer (NST) on the simulated images [29]. In particular, we used 300 simulated images from prior executions, together with a single reconstructed scene, to fine-tune an existing VGG model [29]. We manually evaluated different model configurations trained with different hyperparameters responsible for the style and content transfer, and used default values for early stopping and max epoch size. We manually selected the hyperparameter setup, yielding the best result in terms of stylization of the image. Example results are provided in Figure 6.

Our results show that augmentation increases the agreement rate by 2.6% over all feature classifications. We can observe that in general, the coloring and the blurriness appear for augmented images closer to the recreated images. The reason for the improved rate could be that objects or features that are not detectable because of their appearance in reality also become undetectable after transformation. To assess whether NST processed images are closer to real-world images, we evaluate the distributional similarity using the Fréchet Inception Distance (FID) and Kernel Inception Distance (KID) [30]. The results show that NST reduces the FID distance by

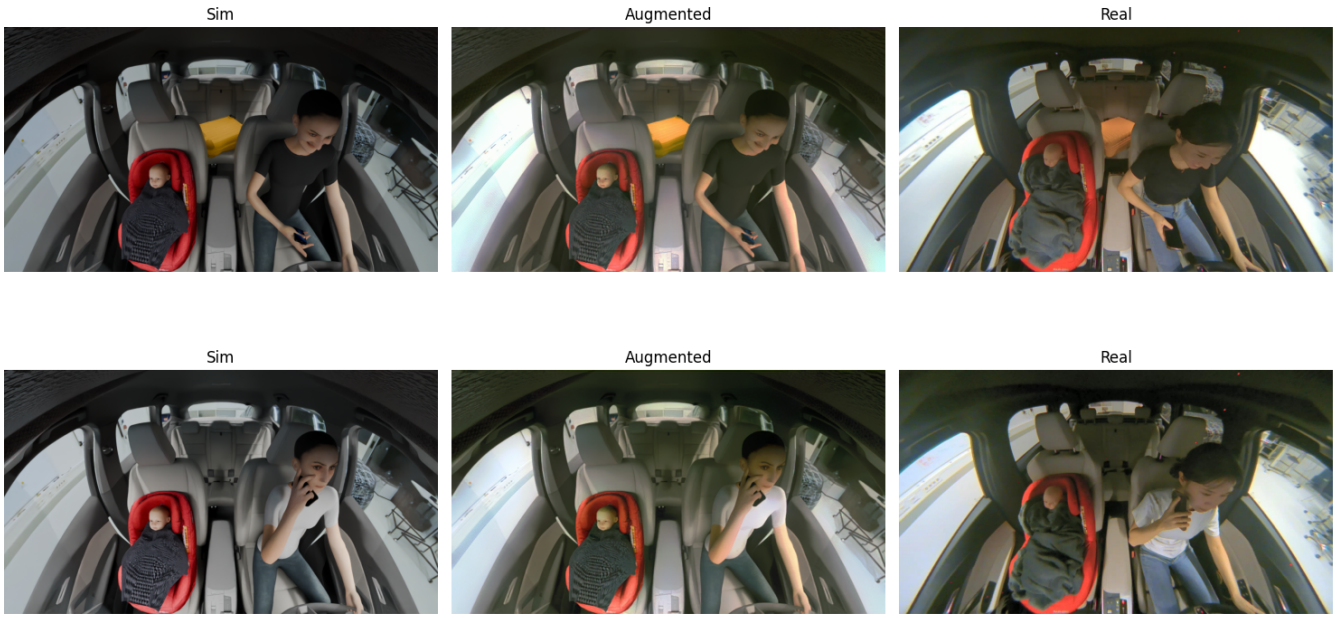

Figure 6: Overview of two different scenes (top/bottom): rendered scene (left), augmented scene using neural style transfer (middle), and reconstructed scene (right).

7.6 scores compared to only simulated images (FID with NST: 151.9 vs. FID for simulated 159.5), while for KID the improvement is 0.025 (KID with NST: 0.15 vs. 0.175) with variance below 0.007. While the results indicate that NST improves the realism of simulated scenes, absolute distance values remain high.

Other Scene Transformation/Augmentation Experiments. In principle, beyond NST, vision–language model–based approaches such as NanoBanana [31] and `GPT-Image-1`, as well as ControlNet-based methods (e.g., NVIDIA Cosmos) [32], could be integrated into ISU-Test to generate more photorealistic scenes. However, preliminary experiments indicate that these approaches do not preserve semantic consistency. For example, the position of the driver's arms may change, or the camera viewpoint may shift (e.g., from a frontal to a lateral view). Such inconsistencies adversely affect failure assessment and limit their suitability for integration into ISU-Test. Similarly, image-to-image-based techniques can be applied to modify clothing colours or allow a visually more realistic seat-belt modeling. Applicable methods include in-painting [33], diffusion-based augmentation [33], and ControlNet-based approaches [34]. However, as before, the scene geometry and semantics were not preserved, which is why further research is required to understand how such techniques can be integrated in a robust manner.

5.2 Failure analysis

We analyzed the failures for both case studies for VQA and VC to understand the different potentials of the underlying VLMs/SUTs. For VQA we used feature-wise comparisons, while VC requires manual inspection as the output is unstructured. We excluded the model MoonDream2 from the analysis because of its high failure rates.

For visual question answering, across both test approaches RS and ISU-Test, for the systems Gemini-2.5-Flash,ISU, and ISU-Flash the most frequent misclassifications happen for *baby seat orientation* with up to 45%, followed by suitcase misdetection with 23.5% for ISU-Test, while for RS the second highest failure category is still baby seat orientation but in the inverse direction.

GPT-5-Chat fails most on emotion detection, detecting a happy person as serious (43%, vs. 48%), followed similarly by a baby seat direction misclassification with 27% vs 21% failure rates. The complete failure distribution can be found in our replication package. Reviewing samples of executed tests in the captioning case study shows that generated tests often contain rich and vivid descriptions. Failures happen in general due to factual hallucinations or missed details. Figure 7 shows, for instance, a failing test with a score of 0.56.

```
The image captures the clean interior of a modern car,
viewed from the rear. A driver, wearing a white shirt
and jeans, is buckled into the driver's seat, holding a
phone to their ear with their right hand while their left
hand rests on the steering wheel. The light gray front
passenger and rear seats are unoccupied.
```

Figure 7: Example of a produced scene description with a low similarity (0.56) to the reference caption. The driver's gender and the seat belt status are missing in the description.

5.3 Expert Feedback

We discussed the evaluation results of ISU-Test with a company expert who has more than 20 years of experience in developing vision-based systems such as ISU and was not involved in this work. The discussion was structured around the following questions:

Based on the results, is ISU-Test helpful for benchmarking ISU systems developed within the company? ISU-Test shows potential as a tool for evaluating prototypical ISU implementations, particularly for identifying failure cases and performance limitations. It is especially suitable for exploring complex and diverse scene configurations, including variations in in-vehicle objects, driver poses, and vehicle models.

How realistic are the scenes generated by ISU-Test? The generated scenes enable the representation of a wide range of conditions. Compared to existing synthetic interior scene datasets and generation approaches, ISU-Test provides more realistic and configurable scene appearances.

Beyond benchmarking, do you see additional application domains for ISU-Test? Simulator-based scene generation facilitates also the creation of large static datasets for comparing different ISU implementations. This capability is not limited to benchmarking but can also support training and fine-tuning of ISU systems. In practice, ISU-Test can be integrated into existing validation workflows as a complementary component to dataset-based evaluation. For instance, it can be used during development to identify failure-prone regions of the input space, and in later stages to perform regression testing across model updates. Compared to manual data collection, the ability to systematically generate targeted scenarios enables faster iteration and more comprehensive coverage.

6 Related Work

6.1 Datasets

SVIRO [9] is a Synthetic dataset for Vehicle Interior Rear seat Occupancy detection as well for classification tasks compromising different data modalities such RGB image, depth image, IR images covering different vehicle types and human appearances. TICAM [35] is an in cabin monitoring dataset compromising synthetic as well as driving simulator recorded and human labeled images focusing on the front interior (more than 100,000). Drive&Act [20] is a dataset which focuses on static simulator in cabin recordings of fine grained driver activities such as the usage of laptops or in-car entertainment functions.

However, all these datasets have in common that they can become part of the training data of an VLM-based in cabin monitoring system limiting their usability for failure/inaccuracy detection. At the same, there is no guidance to potential corner cases what is in contrast enabled in our optimization-based approach.

6.2 Test Generation

Attaoui and Pastore [36] developed DESIGNATOR, a toolset for generating datasets for testing and retraining deep neural networks (DNNs) performing computer vision tasks in Martian-like environments. While their approach can also generate synthetic images through simulation and relies on parameter optimization, it is limited to the testing of DNNs focusing solely on the mars environment simulation. Haq et al. [37] used as ISU-Test an optimization based approach and a rendering engine to find facial expression/images making a key-point detection systems misclassify key points. Also this approach is limited to DNN testing and uses a specific simulation setup. Baresi et al. [33] apply diffusion-based transformation techniques to augment driving scenarios when testing a vision-based lane-keeping assistant.

To the best of our knowledge, ISU-Test is the first approach to implement automated and guided test generation for benchmarking VLM-based in-car scene understanding systems.

7 Conclusions and Future Work

We presented ISU-Test, an automated testing framework for evaluating vision-language model (VLM)-based in-car scene understanding systems. Our approach combines controllable, rendering-based scene generation with search-based testing to systematically explore the space of in-cabin scenarios and identify failure-inducing inputs. Our evaluation on both open-source and industrial systems demonstrates that ISU-Test consistently outperforms randomized testing, achieving substantially higher failure rates and improved failure coverage while maintaining comparable or higher diversity. Importantly, the agreement between simulated and real-world evaluations indicates that failures discovered in simulation largely transfer to physical settings, supporting the use of simulation-based testing as a practical proxy for early-stage validation. These findings highlight a key limitation of current validation practices for VLM-based systems: static datasets are insufficient to expose rare, safety-critical behaviors. Instead, systematic and controllable test generation is required to ensure robustness under diverse and previously unseen conditions.

Future work will extend ISU-Test along several dimensions. First, we plan to integrate behaviorally consistent generative models (e.g., diffusion- or control-based approaches) to improve photorealism while preserving semantic correctness. Second, we aim to support temporal and multi-view reasoning to evaluate dynamic scenarios involving driver actions and object interactions over time to detect safety-critical situations.

8 Data Availability Statement

We provide a replication package with our framework, evaluation scripts, and datasets [19] for the studies with standalone VLMs for in-car scene understanding. The SUT and data related to the industrial case study cannot be shared due to confidentiality constraints; for these, we report only aggregate results.

References

[1] F. Diederichs and F. IOSB, "Pioneering in-cabin monitoring: Unmasking the power of 2d and 3d cameras through sensor fusion," Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB), Tech. Rep., — 2025, white Paper. [Online]. Available: https://www.iosb.fraunhofer.de/content/dam/iosb/iosbtest/documents/kompetenzen/bildauswertung/hai/projekte-und-produkte/White%20Paper_In-Cabin%20Monitoring_2025_red..pdf

[2] A. Mishra, S. Lee, D. Kim, and S. Kim, "In-cabin monitoring system for autonomous vehicles," *Sensors*, vol. 22, no. 12, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/12/4360

[3] E. N. C. A. P. E. NCAP), "Assessment protocol – assisted driving: Highways & interurban assist systems. technical bulletin sd 202 – driver monitoring test procedure," Euro NCAP, Tech. Rep., Mar. 2025, implementation January 2026. [Online]. Available: https://www.euroncap.com/media/85831/euro-ncap-protocol-assisted-driving-v10.pdf

[4] European Parliament and Council, "Regulation (eu) 2019/2144 on type-approval requirements for motor vehicles," 2019, official Journal of the European Union.

[5] Euro NCAP, "The european new car assessment programme," https://www.euroncap.com/en, 2025, accessed: 2025-10-26.

[6] N. Humbatova, G. Jahangirova, G. Bavota, V. Riccio, A. Stocco, and P. Tonella, "Taxonomy of real faults in deep learning systems," in *Proceedings of the ACM/IEEE 42nd international conference on software engineering*, 2020.

[7] V. Riccio, G. Jahangirova, A. Stocco, N. Humbatova, M. Weiss, and P. Tonella, "Testing machine learning based systems: a systematic mapping," *Empirical Software Engineering*, vol. 25, 2020.

[8] Y. Gao, M. Piccinini, Y. Zhang, D. Wang, K. Moller, R. Brusnicki, B. Zarrouki, A. Gambi, J. F. Totz, K. Storms, S. Peters, A. Stocco, B. Alrifaee, M. Pavone, and J. Betz, "Foundation models in autonomous driving: A survey on scenario generation and scenario analysis," *IEEE Open Journal of Intelligent Transportation Systems*, 2026. [Online]. Available: https://arxiv.org/abs/2506.11526

[9] S. D. D. Cruz, O. Wasenm¨uller, H.-P. Beise, T. Stifter, and D. Stricker, "Sviro: Synthetic vehicle interior rear seat occupancy." TIB, dec 2024. [Online]. Available: https://service.tib.eu/ldmservice/dataset/sviro--synthetic-vehicle-interior-rear-seat-occupancy

[10] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. A. Osman, D. Tzionas, and M. J. Black, "Expressive body capture: 3D hands, face, and body from a single image," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10 975–10 985.

[11] L. Sorokin, I. Vasilev, K. E. Friedl, and A. Stocco, "STELLAR: A search-based testing framework for large language model applications," in *Proceedings of the 33rd IEEE International Conference on Software Analysis, Evolution and Reengineering*. IEEE, 2026.

[12] Blender Foundation, *Shrinkwrap Modifier*, Blender Foundation, 2023, blender 3.6 Manual, accessed 2026-04-23. [Online]. Available: https://docs.blender.org/manual/id/3.6/modeling/modifiers/deform/shrinkwrap.html

[13] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02. USA: Association for Computational Linguistics, 2002, p. 311–318.

[14] A. Lavie and A. Agarwal, "Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments," in *Proceedings of the Second Workshop on Statistical Machine Translation*, ser. StatMT '07. USA: Association for Computational Linguistics, 2007, p. 228–231.

[15] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," *arXiv preprint arXiv:1904.09675*, 2019.

[16] R. W. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[17] M. Biagiola and P. Tonella, "Testing of deep reinforcement learning agents with surrogate models," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 3, 2024.

[18] R. Feldt, S. Poulding, D. Clark, and S. Yoo, "Test set diameter: Quantifying the diversity of sets of test cases," in *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, 2016.

[19] Anonymous, "Replication package," https://figshare.com/s/cb5b0eae0411e54b1bbd.

[20] M. Martin, A. Roitberg, M. Haurilet, M. Horne, S. Reiß, M. Voit, and R. Stiefelhagen, "Drive&act: A multi-modal dataset for fine-grained driver behavior recognition in autonomous vehicles," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 2801–2810.

[21] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," 2015. [Online]. Available: https://arxiv.org/abs/1412.2306

[22] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," 2015. [Online]. Available: https://arxiv.org/abs/1411.4555

[23] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Batra, and D. Parikh, "Vqa: Visual question answering," 2016. [Online]. Available: https://arxiv.org/abs/1505.00468

[24] R. Bernardi, R. Cakici, D. Elliott, A. Erdem, E. Erdem, N. Ikizler-Cinbis, F. Keller, A. Muscat, and B. Plank, "Automatic description generation from images: A survey of models, datasets, and evaluation measures," 2017. [Online]. Available: https://arxiv.org/abs/1601.03896

[25] R. Vedantam, C. L. Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," 2015. [Online]. Available: https://arxiv.org/abs/1411.5726

[26] M. Kilickaya, A. Erdem, N. Ikizler-Cinbis, and E. Erdem, "Re-evaluating automatic metrics for image captioning," 2016. [Online]. Available: https://arxiv.org/abs/1612.07600

[27] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, 1945.

[28] A. Vargha and H. D. Delaney, "A critique and improvement of the "cl" common language effect size statistics of mcgraw and wong," *Journal of Educational and Behavioral Statistics*, vol. 25, no. 2, pp. 101–132, 2000. [Online]. Available: http://www.jstor.org/stable/1165329

[29] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," *Journal of Vision*, vol. 16, no. 12, p. 326, 2016.

[30] S. C. Lambertenghi and A. Stocco, "Assessing quality metrics for neural reality gap input mitigation in autonomous driving testing," in *Proceedings of 17th IEEE International Conference on Software Testing, Verification and Validation*, ser. ICST '24, 2024.

[31] Nanobana, "Nano banana ai image generator," https://www.nanobana.net/, 2026, aI-based image generation and editing platform.

[32] NVIDIA Corporation, "Nvidia cosmos: World foundation models for physical ai," https://www.nvidia.com/en-us/ai/cosmos/, 2026, open platform with world foundation models and data processing for robotics, autonomous systems, and physical-AI research.

[33] L. Baresi, D. Y. Xian Hu, A. Stocco, and P. Tonella, "Efficient domain augmentation for autonomous driving testing using diffusion models," in *2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)*, 2025, pp. 398–410.

[34] P. Alimisis, I. Mademlis, P. Radoglou-Grammatikis, P. Sarigiannidis, and G. T. Papadopoulos, "Advances in diffusion models for image data augmentation: A review of methods, models, evaluation metrics and future research directions," *Artificial Intelligence Review*, vol. 58, p. 112, 2025. [Online]. Available: https://doi.org/10.1007/s10462-025-11116-x

[35] J. S. Katrolia, A. El-Sherif, H. Feld, B. Mirbach, J. R. Rambach, and D. Stricker, "Ticam: A time-of-flight in-car cabin monitoring dataset," in *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*. BMVA Press, 2021, p. 277. [Online]. Available: https://www.bmvc2021-virtualconference.com/assets/papers/0701.pdf

[36] M. O. Attaoui, F. Pastore, and L. C. Briand, "Search-based dnn testing and retraining with gan-enhanced simulations," *IEEE Trans. Softw. Eng.*, vol. 51, no. 4, p. 1086–1103, Apr. 2025. [Online]. Available: https://doi.org/10.1109/TSE.2025.3540549

[37] F. U. Haq, D. Shin, L. C. Briand, T. Stifter, and J. Wang, "Automatic test suite generation for key-points detection dnns using many-objective search (experience paper)," in *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ser. ISSTA '21. ACM, Jul. 2021, p. 91–102. [Online]. Available: http://dx.doi.org/10.1145/3460319.3464802