Efficient Domain Augmentation for Autonomous Driving Testing Using Diffusion Models

Luciano Baresi[1], Davide Yi Xian Hu[1], Andrea Stocco[2,3], Paolo Tonella[4]

[1]*Politecnico di Milano*, Milano, Italy, luciano.baresi, davideyi.hu@polimi.it

[2]*Technical University of Munich*, Munich, Germany, andrea.stocco@tum.de

[3]*fortiss GmbH*, Munich, Germany, stocco@fortiss.org

[4]*Software Institute - USI*, Lugano, Switzerland, paolo.tonella@usi.ch

Abstract—**Simulation-based testing is widely used to assess the reliability of Autonomous Driving Systems (ADS), but its effectiveness is limited by the operational design domain (ODD) conditions available in such simulators. To address this limitation, in this work, we explore the integration of generative artificial intelligence techniques with physics-based simulators to enhance ADS system-level testing. Our study evaluates the effectiveness and computational overhead of three generative strategies based on diffusion models, namely instruction-editing, inpainting, and inpainting with refinement. Specifically, we assess these techniques' capabilities to produce augmented simulator-generated images of driving scenarios representing new ODDs. We employ a novel automated detector for invalid inputs based on semantic segmentation to ensure semantic preservation and realism of the neural generated images. We then performed system-level testing to evaluate the ability of the ADS to generalize to newly synthesized ODDs. Our findings show that diffusion models help to increase the coverage of ODD for system-level ADS testing. Our automated semantic validator achieved a percentage of false positives as low as 3%, retaining the correctness and quality of the images generated for testing. Our approach successfully identified new ADS system failures before real-world testing.**

Index Terms—**autonomous driving systems, deep learning testing, diffusion models, generative AI**

I. INTRODUCTION

Before deploying Autonomous Driving Systems (ADS) on public roads, extensive simulation-based testing [1] is performed to ensure that these systems can effectively handle the scenarios of the Operational Design Domain (ODD), which defines the specific conditions under which an ADS is designed to operate [2], [3]. Current driving simulators are based on game engines (e.g., Unity [4] or Unreal [5]) and enable testing using a predefined set of ODD conditions. Simulators are often limited in the range of ODD they represent, as they primarily focus on photorealistic rendering and accurate physics representation. Thus, they fail to cover many ODD scenarios that are instead critical for testing ADS. This limitation hinders the effectiveness of ADS system-level testing [6], [7], particularly for edge cases *beyond* the predefined ODD conditions available in these simulators.

Enhancing ODD coverage in a simulator typically requires developing new conditions within the simulator engine, a task that demands significant domain knowledge and development effort. Moreover, even if improving the simulation platform were feasible, recent research highlighted the problem of fidelity gap between the virtual environments represented in the simulators and the real world [8].

Generative Artificial Intelligence (GenAI) solutions have been employed to improve and extend the range of ODD conditions for ADS testing [8]–[13]. However, approaches such as DeepRoad [12] and TACTICS [11] use GenAI techniques that require mining a training corpus of ODD data and focus on offline model-level testing of individual Deep Neural Networks (DNNs) against different neural-generated images. Furthermore, these approaches have not been evaluated for system-level testing, which is crucial for assessing the safety requirements of autonomous driving [14]. On the other hand, closed-loop data-driven approaches, such as DriveGAN [15], use GenAI to produce a continuous stream of driving images. However, their primary limitations lie in their dependence on learned physics models, which may be inaccurate or unrealistic. Consequently, these approaches are mainly useful for training data augmentation, rather than for testing. The lack of robust physics engines leads to inconsistent physical behaviors and interactions, making it impossible to simulate system-level failures such as collisions or vehicles driving off-road, as such scenarios are not available in the training data.

In this paper, we evaluate three different augmentation strategies based on state-of-the-art pre-trained diffusion models [16]–[18]: Instruction-editing, Inpainting, and Inpainting with Refinement, used to expand the set of ODD conditions in a driving simulator via real-time image-to-image translation. Unlike existing GenAI techniques that require explicit training, diffusion models only require an input image and conditioning inputs that represent the transformation to be applied (e.g., a textual description).

As the diffusion models' output can be affected by artifacts, distortions, or inconsistencies that undermine the effectiveness of ADS testing [19], our methodology includes an automated validation technique based on image semantics and segmentation maps. This check is performed to assess the correctness and reliability of the generated images (e.g., ensuring that the original road shape is preserved after the augmentation).

Although diffusion models can generate diverse images for testing, their direct use in a simulator faces two main challenges. First, diffusion models exhibit high inference times, in the order of seconds per image, which makes them impractical for real-time applications. Second, simulation platforms typically generate multiple image frames per second, but diffusion models lack the rendering consistency required for a coherent simulation, resulting in consecutive frames that may be drastically different. To address these limitations, our approach leverages knowledge distillation [20] by integrating diffusion models with a cycle-consistent network [21] to ensure domain generation consistency and high throughput.

In our experiments, our approach generated images that exhibited a validity rate between 52% and 99%, the best approach being Inpainting. When used as a rendering engine within the Udacity simulator to produce 108 simulations using various ODD conditions, our approach was able to reveal a total of 600 failures across four lane-keeping ADS, 20 times more than using the ODD conditions available in the simulator, at the cost of an increase in simulation time of 2%. Furthermore, we evaluated the generalizability of our approach to complex urban scenarios using the CARLA simulator and InterFuser [22], successfully exposing 25 red-light infractions and 3 vehicle collisions previously undetected.

Our paper makes the following contributions.

Approach. A system-level testing technique for ADS that combines different GenAI (e.g., diffusion models and cycle-consistent generative networks) as a rendering engine and physics-based simulators for effective failure detection. This novel combination of techniques achieves high ODD diversity, realism, semantic preservation, temporal consistency, and high throughput, while the simulator's underlying physics ensures accurate representation. Our approach is integrated in the Udacity simulator for self-driving cars and is publicly available [23]. To the best of our knowledge, this is the first solution that uses GenAI techniques within a driving simulator to improve the ODD coverage of DNN-based ADS.

Evaluation. An empirical study on the validity and realism of neural-generated driving images and their use for system-level ADS testing.

Dataset. A dataset of more than 1 million pairs of images and 52 OOD conditions based on the Udacity simulator and a dataset of 1 million images and 9 OOD conditions based on the CARLA simulator. These datasets can be used to evaluate the generalizability of ADS to novel environmental conditions, as well as the performance of failure prediction systems.

II. BACKGROUND

A. System Level Testing of ADS

ADS must adhere to specific regulations that establish essential safety requirements for public acceptance and large-scale deployment. In particular, standards such as the ISO/PAS 21448 Safety of the Intended Function (SOTIF) standard [24] or the UN Regulation No 157 (2021/389) concerning the approval of vehicles with regards to Automated Lane Keeping Systems [25], demand extensive coverage of the Operational Design Domain (ODD) conditions. The ODD should describe the conditions under which the automated vehicle is intended to drive autonomously, such as roadway types; geographic area; speed range; environmental conditions (weather as well as day/night time); and other domain constraints. In this work, we focus on the *ODD* conditions that *visually* impact the environment and the DNNs of the ADS. Specifically, we used the conditions described in the standard ISO 34505 [2] "Scenery Elements (Section 9)" and "Environmental Conditions (Section 10)", some examples being different geographic areas (e.g., European cities or coastal areas) and weather conditions (e.g., cloudy or rainy weather as well as day/night).

The safe deployment of ADS requires a thorough exploration of ODDs through simulated and in-field testing. Due to significant time, space, and cost constraints associated with in-field testing (i.e., real-world testing with physical vehicles), simulation-based testing has become the standard option for system-level testing of ADS [8]. Driving simulators can generate data and conditions that closely mimic those encountered in real-world scenarios [26]. To test the limits of the ADS, a simulator produces a vast amount of highly consistent data through synthetic input generation using a 3D image rendering engine. However, a comprehensive test dataset must not only be statistically significant in volume but also adequately represent the diverse ODD conditions. This is a major limitation of current driving simulators, which often have restricted ODD coverage, which is essential for comprehensive testing and fault exposure [1].

B. Vision Generative AI

Generative AI has significantly advanced various vision tasks by enabling the creation of realistic and diverse data [27].

In this paper, we consider techniques that allow one to *control the content of the augmentation*. We experiment with Diffusion Models [28], a class of (Vision) Generative AI techniques that achieved state-of-the-art performance in image generation tasks [29]. These models operate by reversing a gradual noising process, starting from a simple distribution and iteratively refining it to generate high-quality images. For example, given an initial noisy image, the model denoises it step by step to produce a coherent image. Different randomly sampled noise seeds lead to variations in the generated images, allowing these models to create diverse and unique images.

Conditional Diffusion Models allow for further control over image generation using different types of input conditioning. For example, Stable Diffusion [18] and DALL-E [30] use *single conditioning* through a textual description to guide the denoising process, as a form of requirement specification (e.g., "*generate a sunny driving image scenario.*"). This guidance concept aims to align the generated images closely with the conditions or descriptions provided. Other techniques use *multiple conditioning*. For example, InstructPix2Pix [16] takes as input an image and a textual editing instruction that describes the modification to be applied to the image, whereas ControlNet [31] facilitates the addition of arbitrary conditioning inputs to a pre-trained Stable Diffusion model.

III. SOLUTION

Our methodology seamlessly integrates into the standard system-level testing loop without requiring major modifications to the simulator or the ADS. The key idea lies in manipulating the environment perceived by the ADS through diffusion-based augmentation, while the actual driving commands are executed on the original simulator. Our methodology consists of three main phases, namely Domain Augmentation, Semantic Validation, and Knowledge Distillation.

The first phase (Domain Augmentation) involves intercepting the images captured by the car cameras. These images are then processed using diffusion models to generate a new image depicting the same road structure but a different ODD condition (such as background and weather conditions). The main reason is that a well-trained lane-keeping ADS should focus on the foreground features that characterize the road scenario, instead of the features in the background [32].

The second phase (Semantic Validation) involves a validation step of the generated image to assess the validity and semantic and label preservation between the original and the augmented image. Particularly, our approach aims to generate road images devoid of visual artifacts, distortions, inconsistencies, or hallucinations and that are *semantically equivalent* to the original one in terms of geometrical features (e.g., direction, lanes, length, width). The semantic validation process is fundamental to maintaining the validity of the augmentation process, as significant changes to the road structure introduced during augmentation could cause the ADS to make decisions based on misleading information. Our approach addresses visual and semantic consistency, whereas physics-related factors such as changes in friction and traction (e.g., due to snowy conditions) are not considered.

The third and last phase (Knowledge Distillation) enables the rendering of new ODDs online, during the execution of a simulation. The diffusion models used in the first phase for domain augmentation are not suitable for online usage because they are too slow at inference time. Hence, we train a faster cycle-consistent generative neural network [21], using the output images produced by the first phase as the training set. At each simulation step, this network transforms the input image to reproduce the domain augmentation of the diffusion models. If the augmented image passes the semantic validation, it is forwarded to the ADS for processing. The ADS processes the image and predicts the appropriate driving commands based on the augmented ODD. The predicted driving commands are sent to the simulator, which actuates them, completing the feedback loop. The simulator then modifies the virtual environment and provides the vehicle with updated sensor data, thus preparing for the next iteration of the testing loop. In the next sections, we describe each step of each phase.

A. Domain Augmentation

We analyze three alternative controllable augmentation strategies based on categories of diffusion models to introduce environmental ODD changes in driving images: *Instruction-editing*, *Inpainting*, and *Inpainting with Refinement*.
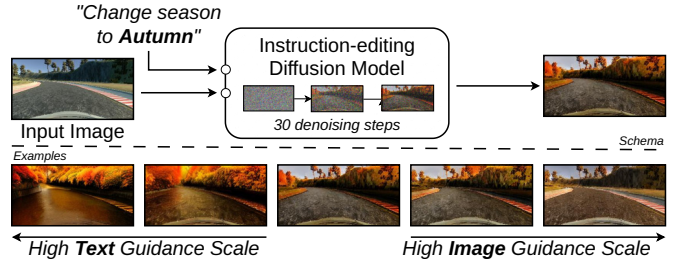

Fig. 1: Instruction-edited Domain Augmentation Strategy.

Instruction-editing. This category takes two inputs: the image to be modified and an editing instruction (e.g., "add trees" or "change season to autumn"), and produces an output image with the editing instruction applied. Instances of Instruction-editing models are InstructPix2Pix [16] and SDEdit [17]. Instruction-editing models can be configured with two parameters, *image* and *text guidance scale*, that represent how much the two inputs influence output generation. The first parameter *image* dictates how much of the structure and spatial details of the input image should be preserved. The *text guidance scale* determines the strength to use when applying the editing instruction. Figure 1 reports a schema of the strategy (top) and how the two guidance scales can influence the augmentation process (bottom). Specifically, we can observe that an excessively high text guidance scale can compromise the road semantics, while overly increasing the image guidance scale too much may result in the insufficient application of the desired edit.

Inpainting. This category employs a text-to-image diffusion model that performs inpainting. Instances of Inpainting models are Stable Diffusion [18], DALL-E [30], and Pixart-α [33].

We customized the inpainting pipeline to preserve the parts of the images related to driving actions (e.g., the road), while the rest of the image can be "repainted" by the diffusion model. We identify the road automatically using a semantic mask that describes which pixels in the image belong to the semantic class *road*. As the semantic mask is provided by the simulator, it ensures perfect semantic segmentation.

The inpainting text-to-image model takes three inputs: an input image, a mask, and a textual prompt that describes the desired image. The model generates an image by preserving only the content selected by the mask while guiding the entire image to align as closely as possible with the textual prompt. In our setting, this process ensures that only the parts outside the road are replaced with new content, thus maintaining the shape of the road since it is not modified by the inpainting strategy. Note that the road in the inpainted image is the same as the one in the input image. For instance, in Figure 2 (top left), the surrounding environment has been transformed, while the road structure, markings, and position are unchanged.

Inpainting with Refinement. This model category adds a step to the *Inpainting* strategy by performing a refinement of the entire image. The goal is to improve the visual coherence between the preserved and the generated parts of the inpainted
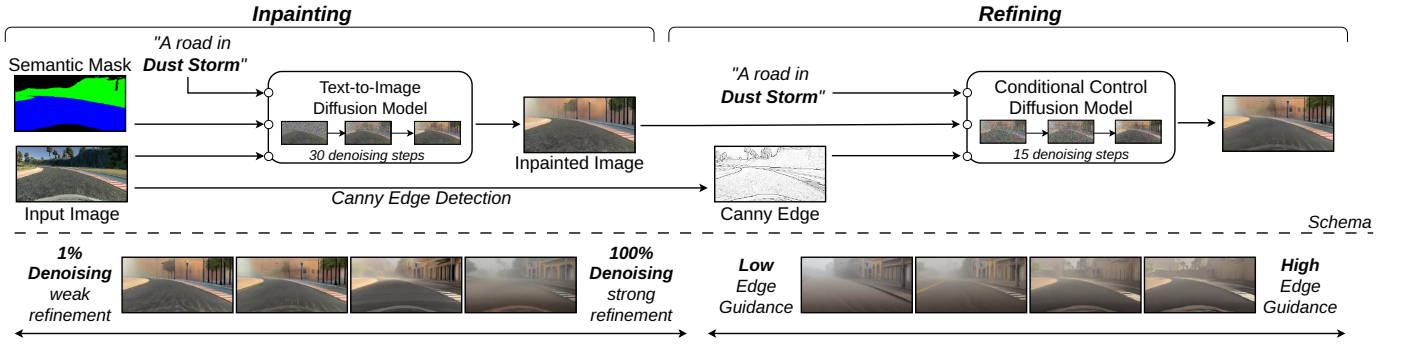
Fig. 2: Inpainting and Inpainting with Refinement Domain Augmentation Strategies.

image. While the traditional denoising process of diffusion models starts from fully-noised images, the refinement step starts from a partially-noised (inpainted) image. This makes the refined image more similar to the initial inpainted one. The noise level removed during denoising determines the difference between the inpainted and final image; higher noise removal leads to greater differences.

The refinement step employs a different type of diffusion model compared to the inpainting step. This is because image generation with a text-to-image model (used during inpainting) can be guided using only text. This is sufficient for the inpainting step as the road semantics are ensured by the semantic mask that preserves the road. However, during the refinement step, the entire image is modified, and thus the shape of the road might change. For this reason, in our study, we evaluated a category of diffusion model with conditional controls that allow to better guide the augmentation process, such as ControlNet [31] with Canny edge [34] conditioning, or T2I Adapter [35]. This model takes three inputs: the edge map derived from the original input image (captured in the simulator), a partially noisy inpainted image, and a textual prompt that describes the desired image. The model takes the noisy image and refines it using both the edge map and the textual prompt. The edge map ensures that the refined image retains edge structures similar to the original, while the textual prompt directs the overall content.

Figure 2 provides an overview of the Inpainting with Refinement process (top), and reports some augmentations obtained with different levels of denoising and guidance scales for the same input image (bottom). A stronger refinement process (more denoising) results in significant differences from the initial image, but can also lead to inferior preservation of road semantics. Similarly, higher edge guidance scale values preserve road semantics more, while lower values encourage greater freedom, diversity, and realism in the generated image.

B. Semantic Validation

Diffusion model categories aim to produce visually appealing outputs, but they may still generate invalid outputs, failing to preserve the semantics of the road during enhancement, for example, by widening the road or introducing new intersec-
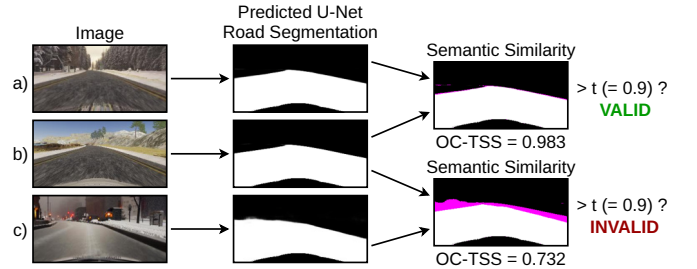

Fig. 3: Semantic Validation using OC-TSS [19].

tions. To mitigate this, our methodology includes a semantic validation step to minimize incorrect augmentations.

The main objective of this phase is to check that the generated augmentation is characterized by a road that is semantically equivalent to the road in the image captured in the simulator. To this end, we filter out augmentations that do not preserve the original road semantics using the OC-TSS metric (One Class - Targeted Semantic Segmentation) [19].

OC-TSS is a similarity metric that measures semantic details and structural differences between two images by focusing on a single task-relevant class within the semantic masks predicted by a fine-tuned segmentation model. This metric ranges from 0 to 1, where 1 indicates perfect semantic equivalence between the original and augmented images, and 0 suggests complete dissimilarity. OC-TSS has been applied in previous work [19] to assess the accuracy of Generative AI models in translating images across domains for a lane-keeping ADS. In line with this study, our analysis focuses on the semantic class "road", as road lanes represent the relevant image characteristics for a lane-keeping ADS. Unlike the original work, we employ a U-Net architecture [36] for semantic segmentation, rather than the SegFormer architecture [37], for its computational efficiency.

Figure 3 illustrates our semantic validation process. The top row of Figure 3 (a) shows a semantically valid augmentation of the input image (Figure 3 (b)), while the bottom row reports an incorrect augmentation (Figure 3 (c)). Specifically, the augmentation in the bottom figure is considered semantically invalid because the road's orientation changes to the left instead of continuing straight as in the original image. The middle

column presents the semantic segmentation masks computed by the U-Net model for each image. In these masks, the road is represented in white, and the background is depicted in black. The final step of our process, represented in the right column, involves measuring the distance between the semantic masks of the augmented and original images, the differences highlighted in pink. Augmentations with a similarity score below a threshold are discarded. Higher threshold values ensure validity, but they may discard valid images and increase the time required to generate a semantically valid augmentation. However, lower thresholds could compromise testing by allowing too many invalid images.

In this study, we determine the threshold based on empirical observations to balance filtering out invalid augmentations and retaining enough variability for thorough ADS testing (Section IV-A).

C. Knowledge Distillation

The final phase involves creating a fast and consistent neural rendering engine in the simulator, using outputs from the previous phases. The diffusion-based models, while effective for diversity, are computationally expensive and can produce potentially inconsistent augmentations, which may be problematic for the temporal coherence of a simulation.

Therefore, we adopt a technique known as knowledge distillation [20], where a smaller model (*student*) is trained to replicate the behavior of a larger, more complex model (*teacher*, i.e., the diffusion model). In particular, for the student model, we use a cycle-consistent generative network [21] that can map images from the original domain (virtual images from the simulator) to another domain (augmented images by the diffusion models). Instances of these architectures are Cycle-GAN [21] or UNIT [38]. This technique is widely used for image-to-image translation tasks, including the autonomous driving domain, due to its low computational overhead, which makes it suitable for runtime usage in simulators [8], [12]. Moreover, training a separate student model for each domain allows effective learning of the key aspects of the teacher model's output, enhancing rendering consistency.

This strategy involves first training the cycle-consistent network to learn the mapping between the original and augmented image domains produced by the diffusion models. This process, while computationally intensive, is performed only once for each domain. Then, during the online system-level testing of the ADS, the trained cycle-consistent network model generator translates images at runtime, i.e., during the execution of the simulation.

An important advantage of this approach is that it does not require collecting new data to train the model, as it leverages existing pre-trained diffusion models. This means that the process can be easily automated and does not require human intervention (e.g., collecting and labeling data), making it an efficient solution for rapidly generating consistent and high-quality domain augmentations online, during ADS simulation.

IV. EVALUATION

We evaluated the proposed approach through the following research questions:

RQ₁ (semantic validity and realism): *Do diffusion models generate images that are semantically valid and realistic? How is the semantic validator at detecting invalid augmentations?*

RQ₂ (effectiveness): *How effective are augmented images in exposing faulty system-level misbehaviors of ADS?*

RQ₃ (efficiency): *What is the overhead introduced by diffusion model techniques in simulation-based testing? Does the knowledge-distilled model speed up computation?*

RQ₄ (generalizability): *Does the approach generalize to complex urban scenarios and multi-modal ADS?*

The first research question aims to assess the semantic validity of the augmentations generated by our methodology. Specifically, the focus is on whether diffusion models can transform images while preserving road semantics, and whether the proposed semantic validator can effectively identify roads with different semantics. The second research question aims to check the utility of the proposed approach in identifying potential faults in ADS that may not be detected using only the simulator's capabilities. The third research question evaluates the computational cost of our approach, which is crucial to understanding scalability in real-world ADS testing scenarios. The fourth research question studies the generalizability of our approach to test multi-modal ADS in complex urban environments with pedestrians, vehicles, and traffic lights.

A. Experimental Setup

In this section we describe the experimental setup used for RQ_1, RQ_2, RQ_3, while the setup changes required for RQ_4 are described in Section IV-F, before presenting the results.

Simulation Platform. We carried out our evaluation using the Udacity simulator with behavioral cloning ADS models [39], a widely adopted platform in the literature [40], [41]. The simulator supports various closed-loop tracks (divided into 40 sectors) to test behavioral cloning ADS, including a pre-defined set of ODDs, such as different times of day/night and three weather conditions (rainy, snowy, and foggy) [42]. We extended the simulator to generate semantic segmentation masks for vehicle camera images of size 160×320 (height $\times$ width) to identify the regions for inpainting. In addition, we developed a synchronous simulation mechanism that pauses the simulation during image augmentation and resumes once the new image is generated. This ensures that the augmentation process is transparent to the system-level testing process.

Lane-keeping ADS. We evaluated four different single-camera, lane-keeping DNN-based ADS as systems under test to assess the performance of the proposed methodology. In particular, we selected Nvidia DAVE-2 [43], Chauffeur [44] and Epoch [45] since they have been often used in multiple testing works [42], [46]–[48]. Finally, we also included a recent architecture based on Vision Transformer (ViT-based) [49] that achieved state-of-the-art performance in lane-keeping tasks.

TABLE I: ODD Domains.

Category	Domains
Weathers	cloudy, dust storm, foggy, lightnings, overcast, smoke, sunny
Seasons	autumn, spring, summer, winter
Daytimes	afternoon, dawn, dusk, evening, morning, night, sunset
Locations	coast, desert, forest, lake, mountain, plains, rivers, rural, seaside
Cities	beijing, berlin, chicago, el cairo, london, new york, paris, rome, san francisco, sidney, tokyo, toronto
Countries	australia, brazil, canada, china, england, france, germany, italy, japan, mexico, morocco, usa

Operational Design Domains Selection. We selected ODDs that encompass diverse conditions from existing standards (see Section II-A). We filtered out domains that do not preserve the driving action when applied for domain augmentation. For example, when converting a sunny road image to a snowy condition, it might require the prediction of a different steering angle that accounts for the different friction, despite the road being the same. Overall, we identified 6 domain categories and 52 distinct label-preserving domains (Table I).

To provide further insight into the difficulty of these domains, we measured the challenge they pose by computing the distance between augmented domains and the training data. This was done using the reconstruction error of a Variational Autoencoder (VAE) [50] to categorize the domains into three clusters: in-distribution domains (closer to the training distribution, e.g., familiar domain or road conditions), in-between domains (moderately different from the training distribution), and out-of-distribution domains (significantly different from the training distribution). In-distribution domains are useful for testing the robustness of the ADS by simulating scenarios similar to those that the model has previously encountered. On the other hand, out-of-distribution domains challenge the ADS's ability to generalize to new, unfamiliar conditions that are not present or rarely available in training data.

To determine the classification of these domains, we first trained the VAE to reconstruct the training data. The lower the reconstruction error, the closer the domain is to the training distribution. We generated $2,000$ augmented images for each domain using the three domain augmentation strategies and measured the reconstruction error for each. The domains were then sorted by reconstruction error and categorized as in-distribution, in-between, or out-of-distribution. Finally, we selected three domains that were categorized in the same group for all three augmentation techniques. The final selections were as follows: for in-distribution domains, we included sunny, summer, and afternoon conditions; for in-between domains, we chose autumn, desert, and winter; and for out-of-distribution domains, we selected dust storm, forest, and night scenarios.
Diffusion Models Calibration. We used three state-of-the-art pre-trained diffusion models for our categories: Instruct-Pix2Pix [16] for Instruction-editing, Stable Diffusion [18] for Inpainting, and ControlNet [31] with Canny edge [34] conditioning for Inpainting with Refinement. We fine-tuned the

hyperparameters of each considered model before answering the research questions. We prioritized image fidelity, adherence to instructions, and preservation of essential road features, which will be evaluated in our first research question. In particular, we configured all diffusion models to use the UNIPC multistep scheduler [51] with 30 inference denoising steps as *noise sampling strategy*. We chose UNIPC because it focuses on generating good images with a few denoising steps. In our exploratory experiments, a higher number of steps did not lead to significantly better images, but only introduced additional computational overhead.

For InstructPix2Pix, we set the image guidance scale to 2 and the text guidance scale to 10. These settings were found to be a good balance between image and instruction inputs, preserving key features from both sources. In the Stable Diffusion inpainting pipeline, we used a text guidance scale of 10, which maintained a high level of control over the generated content without compromising the quality of the inpainting process. ControlNet refining was configured with a text guidance scale of 10 and a noise level of 50%. This configuration preserved the structural integrity of the road while still allowing for meaningful and diverse augmentations. Higher noise levels were found to risk excessive alteration of images and the potential loss of essential road semantics.
Semantic Validator Configuration. To determine an appropriate threshold for the OC-TSS metric, we collected 150 images from the simulator with different semantics, which were manually assigned to three categories: images of straight roads, right turns, and left turns. We computed the OC-TSS for all images and evaluated the similarity between images within the same category and across different categories.

We aimed for a threshold that considers images within the same category as semantically similar and those in different categories as distinct, giving higher priority to filtering out invalid images that belong to another category, rather than including as many valid images from the same category as possible. Consequently, after manual inspection of a sample of included/excluded images, we chose a conservative threshold of 0.9, which minimizes the inclusion of semantically incorrect augmentations, while at the same time avoiding the exclusion of too many valid images. Thus, augmentations with an OC-TSS ≥ 0.9 are considered semantically consistent with the original layout of the road by our automated semantic validator, while those below 0.9 are considered invalid and discarded.
Knowledge Distillation Configuration. We used CycleGAN as the student model to distill knowledge from the pre-trained diffusion models. The CycleGAN architecture followed the recommendations in [52] to reduce droplet artifacts.

We trained one CycleGAN for each of the nine selected ODDs and the three augmentation strategies, resulting in 27 models. Each model was trained for 10 epochs using $2,000$ pairs of images (from the simulator and augmented), with checkpoints saved at the end of every epoch. The best checkpoint was selected according to the Fréchet Inception Distance [53], a metric that measures the distance between

two sets of images (the ones generated by CycleGAN and the ones generated by the diffusion models) by comparing their feature distributions.

Hardware and Software. All experiments were executed on a server with an AMD 5950X CPU, 64 GB RAM, and two Nvidia 4090 GPUs (24 GB VRAM each). The software environment includes Python 3.10, CUDA 12.1 for GPU acceleration, Pytorch 2.3.0 for ADS implementations, and Huggingface `diffusers` 0.27.2 for the diffusion models.

Our evaluation required more than $1,000$ GPU hours and involved 2.5 million image pairs generated over 52 ODD domains using 3 augmentation techniques. This process included filtering the domains to keep the experiment manageable within a reasonable time frame, as training 36 CycleGAN models for 10 epochs each took more than 150 GPU hours.

B. Metrics

Semantic Validator Effectiveness. We consider valid augmentations as the positive class and invalid augmentations as the negative class. Correspondingly, a True (resp. False) Positive TP (resp. FP) is an image regarded as a valid augmentation by our semantic validator, which is valid (resp. invalid) according to the ground truth. Similarly, a True (resp. False) Negative TN (resp. FN) is an image regarded as an invalid augmentation by our semantic validator, which is invalid (resp. valid) according to the ground truth. To assess the effectiveness of our semantic validation methodology, we utilize the confusion matrix [[TP, FP], [FN, TN]], either with absolute or percentage values.

Testing Effectiveness. We utilize two categories of metrics to evaluate ADS performance at the system level: one for measuring misbehavior and another for assessing driving quality. The first category directly quantifies errors, including incidents in which the vehicle deviates from the lane boundaries (Out-of-Bounds, OOB) or collides with obstacles (C). We use Failure Track Coverage (FTC) to capture the spatial distribution of errors, which identifies the percentage of track sectors where misbehaviors occur. This metric indicates whether errors are concentrated in specific challenging areas or spread over the entire track. In this way, we can determine whether errors arise primarily from the complexity of specific track sections or are induced more broadly by the new domain.

We assess two key metrics for driving quality relative to the nominal behavior of the ADS. The first is the Relative Cross-Track Error (RCTE), which measures the ratio of the average distance from the lane center in the test domain compared to the nominal domain. An RCTE value greater than 1 indicates degraded performance (the vehicle is closer to the edge of the road), while a value less than 1 indicates improved position accuracy (the vehicle is closer to the center of the road). The second metric is Relative Steering Jerk (RSJ), which calculates the difference in the rate of change of steering angle between the test and nominal domains. Higher RSJ values suggest more abrupt steering adjustments, while lower values indicate smoother driving. Although these metrics do

TABLE II: RQ$_1$: Semantic Validity Confusion Matrix.

	Valid Augmentation	Invalid Augmentation
Predicted Valid Augmentation	48 (55%)	3 (3%)
Predicted Invalid Augmentation	16 (18%)	20 (23%)

not directly indicate errors, they provide valuable insight into potential performance degradation caused by specific domains. **Computational Overhead.** We evaluate the computational overhead of our domain augmentation strategies by measuring the average time required to generate the augmented image and the average time required to complete our experiments, including both the augmentation process and the subsequent testing of the ADS. We compare these timings with a baseline where no augmentation is applied, allowing us to quantify the additional computational load introduced by each strategy.

C. RQ$_1$: Semantic Validity and Realism

We conducted two surveys with human assessors to evaluate the semantic validity of the images generated by the diffusion models, as well as their degree of realism. We recruited participants from Amazon Mechanical Turk (MTurk) and personal contacts using convenience sampling [54]. Each MTurk participant answered 200 questions, while the others answered 100 questions. In the two studies, we collected $5,300$ responses, of which only $4,500$ were retained due to failure to answer the control questions of our surveys. Ultimately, we retained responses from 35 participants, of which 10 from MTurk and 25 from personal contacts.

1) Semantic Validity: Participants were shown two randomly ordered images and asked whether the images represented the same semantics of the road, focusing on the shape of the road and the direction of turn. For this study, we randomly selected 36 pairs of images for each of the three domain augmentation strategies (Instruction-editing, Inpainting, and Inpainting with Refinement). This resulted in a total of 108 pairs, with each strategy contributing 18 semantically valid transformations and 18 invalid transformations, according to our semantic validator. In addition, we included two control questions to filter out low-quality responses: one where the road was the same and one where the road was entirely different. In total, the first questionnaire contained 110 questions, of which two were used for quality checks.

We considered a road semantically equal (or different) when at least $\frac{2}{3}$ of the participants agreed on the outcome. Overall, the participants reached a consensus on 80.6% of the pairs (87 out of 108). Our study revealed a positive correlation between OC-TSS similarity scores and user opinions, with a Pearson correlation coefficient of 0.63 (p-value=$2.63 \cdot 10^{-13}$).

Table II presents the results as a confusion matrix, where columns represent the judgments of human participants and rows show the results of the semantic validator, with valid augmentations considered the positive class. Our semantic validator failed to filter out invalid generations in only 3% of the
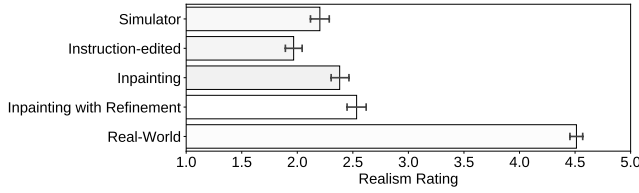
Fig. 4: RQ$_1$: Realism.

cases (3 FPs), while rejecting 18% of the valid transformations (16 FNs). The former error can affect the effectiveness of the proposed methodology as it could lead to testing ADS on images with different semantics, potentially compromising the validity of ADS testing. The latter, while less severe, may increase the time required to find valid augmentations.

2) GenAI Realism: In the second study, we evaluated the realism of the augmented images. Participants were presented with individual images and asked to rate their realism on a 5-point scale ranging from 1 (not realistic) to 5 (very realistic). We selected 18 semantically valid transformations for each of the three domain augmentation strategies. We also included 18 images from the simulator and 18 real-world driving images for a better comparison. In total, the second questionnaire contained 90 questions. We computed the average realism score for each category of images (augmented, simulator, and real-world) based on the participants' ratings.

Figure 4 shows the results. The images generated by *Inpainting* and *Inpainting with Refinement* strategies are perceived as more realistic compared to those generated by the *Instruction-editing* strategy. The *Mann-Whitney U test* with an alpha of 0.05 indicates statistically significant differences in realism scores between strategies, with a medium effect size (0.5 and 0.6, respectively). Furthermore, *Inpainting with Refinement* was found to produce more realistic images than *Inpainting* (p-value=0.02), with a small effect size (0.17).

RQ$_1$: *The output of our automated semantic validator matched human judgment, with only 3% of the augmented images incorrectly regarded as valid by the validator (and 18% valid augmentations incorrectly discarded by the validator). Inpainting with Refinement is the augmentation approach that produces the most realistic images.*

D. RQ$_2$: Effectiveness

This experiment evaluates the effectiveness of the domain augmentation approaches in discovering errors in lane-keeping ADS. The experiment aims to determine how many errors each approach could find and the nature of these errors. For each strategy, we ran $2,000$ ADS simulation steps for each augmented domain, collecting the failures and driving quality metrics (Section IV-B). As a baseline, we used the predefined set of domains available within the simulator (Section IV-A).

Table III reports the results with the augmented domains, and Table IV shows the domains available in the simulator. The findings indicate that the proposed methodology can identify

misbehaviors in all four ADS. As expected, domains similar to training conditions (in-distribution) showed fewer errors and lower failure track coverage than in-between or out-of-distribution domains, reflecting the increased difficulty due to domain shifts. However, even for *in-distribution* domains, our approach revealed failures, from a total of 6 OOB incidents (DAVE-2) to 5 collisions and 22 OOB incidents (ViT-based). The maximum failure track coverage with in-distribution domains was 25%. Failure track coverage increased as the domains deviated further from the training set. For example, when Epoch was tested with in-between domains, new errors were seen in up to 75% of the sectors, rising to 87.5% with out-of-distribution domains.

Simulated domains revealed errors in up to 17.5% of the track, while augmented domains reached 87.5%. In-distribution domains generated by the augmentation strategies did not show more misbehaviors than those in the simulator. In contrast, in-between and out-of-distribution domains triggered more errors, especially with the Instruction-editing strategy. Although these scenarios occur less often in training, the generated roads remain semantically validated and sufficiently representative for valid testing.

Furthermore, we found that the Instruction-editing strategy was the most effective across all three domain sets (in-distribution, in-between, and out-of-distribution). This effectiveness can likely be attributed to two main factors. First, the domains generated using instruction-editing were perceived as less realistic in our human study (Section IV-C). Despite being semantically validated by our automated validator, these less realistic images may still mislead the ADS into making incorrect decisions. Second, the average distance from these domains to the training domains was higher compared to that generated by the other two domain augmentation strategies. Specifically, the average reconstruction errors for the Instruction-editing strategy ranged from 0.074 (in-distribution) to 0.218 (out-of-distribution), while the errors for the Inpainting strategy ranged from 0.067 to 0.078, and for the Inpainting with Refinement strategy, they ranged from 0.070 to 0.082.

We observed different behaviors between ADS models based on Convolutional DNNs (DAVE-2, Chauffeur, and Epoch) and the one based on Vision Transformer. For the first ADSs, our augmented domains generally led to lower RSJ, indicating smoother steering responses. In contrast, the ViT-based ADS showed an increase in RSJ as domain distance increased, suggesting more abrupt steering adjustments in response to unfamiliar scenarios. These differences can likely be attributed to how these two types of neural network architectures process data. Convolutional DNNs, with their hierarchical structure and localized receptive fields, tend to focus on local features, which may allow for more stable and incremental responses. In contrast, ViTs, which utilize global attention mechanisms, can capture broader contextual information but may also be more sensitive to domain shifts, leading to more pronounced reactions to unfamiliar data.

To assess whether the augmented data can enhance the robustness of ADS, we retrained the ADS using a combination

TABLE III: RQ$_2$: Effectiveness results for system-level testing on augmented domains.

	In-distribution domains					In-between domains					Out-of-distribution domains				
	C	OOB	FTC	RCTE	RSJ	C	OOB	FTC	RCTE	RSJ	C	OOB	FTC	RCTE	RSJ
DAVE-2															
Instruction-editing	0	4	5.0%	1.13	0.85	10	19	45.0%	1.62	0.61	8	66	82.5%	1.90	0.70
Inpainting	0	2	2.5%	1.10	1.06	0	0	0.0%	1.06	0.95	3	7	12.5%	1.21	1.06
Inpainting with Refinement	0	0	0.0%	0.75	2.37	0	0	0.0%	0.90	0.68	0	4	5.0%	0.96	0.80
Chauffeur															
Instruction-editing	0	7	12.5%	1.32	0.68	4	19	35.0%	1.60	0.63	7	67	70.0%	1.60	0.57
Inpainting	0	0	0.0%	0.93	0.82	1	2	7.5%	1.23	0.80	1	7	12.5%	1.05	0.80
Inpainting with Refinement	1	3	7.5%	1.03	0.71	0	0	0.0%	0.93	0.68	4	6	20.0%	1.39	0.80
Epoch															
Instruction-editing	3	11	25.0%	2.24	0.67	3	54	75.0%	2.28	0.52	10	70	87.5%	2.18	0.41
Inpainting	0	3	7.5%	1.62	0.86	0	5	7.5%	1.65	0.71	3	30	52.5%	1.98	0.71
Inpainting with Refinement	0	8	12.5%	1.55	0.67	4	2	12.5%	1.71	0.67	4	10	15.0%	1.80	0.62
ViT-based															
Instruction-editing	2	8	12.5%	1.15	1.30	2	13	27.5%	1.23	2.45	3	21	45.0%	1.77	3.40
Inpainting	1	6	12.5%	1.21	1.74	1	15	25.0%	1.21	1.85	0	10	17.5%	1.07	2.02
Inpainting with Refinement	2	8	12.5%	1.05	1.79	2	12	25.0%	1.17	1.77	2	20	22.5%	1.02	1.02

TABLE IV: RQ$_2$: Effectiveness results for system-level testing on domains available in the Udacity simulator.

	Simulator Domains				
	C	OOB	FTC	RCTE	RSJ
DAVE-2	1	6	12.5%	1.19	0.96
Chauffeur	2	9	17.5%	1.20	0.99
Epoch	0	5	10.0%	1.68	1.01
ViT-based	1	7	17.5%	1.21	1.22

TABLE V: RQ$_3$: Performance Overhead.

	Augmentation Time (ms)	Testing Run Time (min)
Baseline (no augmentation)		15.7 ± 0.0
Instruction-editing	894.7 ± 0.8	76.9 ± 5.1
Inpainting	1245.2 ± 25.7	53.2 ± 1.2
Inpainting with Refinement	2172.6 ± 29.1	74.1 ± 1.7
Knowledge Distillation	12.3 ± 0.7	16.0 ± 0.1

of the original training data and the augmented ones. We fine-tuned the driving models for 50 additional epochs, using early stopping with a patience of 20 epochs based on improvement in validation loss. Empirical results show that, across the simulator domains, retrained ADS models showed, on average, a 35.2% misbehavior reduction, up to 81.7% in specific scenarios (i.e., foggy weather conditions).

> **RQ$_2$**: *The proposed augmentation technique has been able to expose failures of four existing ADS models, even in domains close to the training one. It represents a valuable complement to the execution of tests in domains supported by the simulator, as it was able to discover failures in sectors that the simulator deemed failure-free.*

E. RQ$_3$: Efficiency

In this experiment, we assessed the overhead introduced by domain augmentation strategies, particularly focusing on the impact of large diffusion models and the potential efficiency gains from a knowledge-distilled model.

Table V presents the results of testing DAVE-2, with similar results observed for other lane-keeping ADS systems. A test run of DAVE-2, consisting of 2,000 simulation steps without augmentation, took approximately 15.7 minutes (about 471.0ms per simulation step). DAVE-2 required only 1.2ms per prediction, while the remaining time was consumed by infrastructure tasks such as communication between the simulator and the agent, image processing, persistent logging, and

simulation management. Other ADS systems showed inference times ranging from 1.1 to 2.0 ms.

The use of diffusion models significantly increased the test duration. Specifically, the Inpainting strategy extended the testing time to 53.2 minutes (more than three times longer than the baseline), while the Instruction-editing and Inpainting with Refinement strategies increased it beyond 70 minutes (more than four times longer). Although the Instruction-editing strategy had a faster per-augmentation time (894.7ms), its overall testing duration was longer because our semantic validator detected a higher percentage of semantically invalid images and needed to be regenerated. Specifically, the semantic validator filtered out about 48% of images augmented by Instruction-editing, less than 1% generated by Inpainting, and 12% generated with Inpainting with Refinement.

In contrast, the knowledge-distilled model based on a CycleGAN architecture significantly reduced the augmentation overhead. It generated images in just 12.3 ms on average, resulting in a total testing time of approximately 16 min, less than a 2% increase over the baseline without augmentation.

> **RQ$_3$**: *Knowledge distillation is an essential component of our approach to achieve high simulation efficiency. The augmentation overhead without knowledge distillation is 470% for Inpainting with Refinement, the technique that produces more valid and more realistic images, which becomes just 2% with knowledge distillation.*

TABLE VI: RQ$_4$: System-level testing results with CARLA.

CARLA Metrics	DS (%)	RC (%)	CP (#)	CV (#)	ORI (#)	RLI (#)	SSI (#)
InterFuser							
Baseline (no augmentation)	84.26	84.26	0	0	0	0	0
Augmented Domains	69.70	77.56	0	3	0	25	0

TABLE VII: RQ$_4$: Overhead in CARLA simulator.

	Augmentation Time (ms)	Testing Run Time (min)
Baseline (no augmentation)		16.6 ± 0.1
Instruction-editing	3377.7 ± 28.9	150.7 ± 9.2
Inpainting	8442.0 ± 58.5	358.5 ± 16.2
Inpainting with Refinement	$11\,350.5 \pm 137.8$	474.5 ± 15.9
Knowledge Distillation	82.0 ± 5.7	19.9 ± 0.2

F. RQ$_4$: Generalizability

To assess how well our methodology can generalize to complex and realistic scenarios, we performed a preliminary evaluation using the CARLA simulator [55], a widely used high-fidelity platform for autonomous driving research [56]–[58]. We focused on scenarios from the CARLA Leaderboard (sensors track) [59] that require the ADS to perform multiple tasks such as lane-keeping, overtaking, obeying traffic signals, stop signs, and detecting and avoiding pedestrians.

We used InterFuser [22] as the system under test, along with the pre-trained model provided in the original paper. InterFuser is an end-to-end ADS for urban driving with one of the highest scores in the CARLA leaderboard [60] and was widely used in prior work [61]–[63]. Technically, InterFuser processes multi-modal sensory inputs: three RGB camera views (front, left, and right) and LiDAR point cloud data, and outputs driving commands such as throttle, brake, and steering.

To assess driving quality and failures, we used several metrics from the CARLA leaderboard. They measure both the ability of the ADS to complete tasks and its adherence to traffic regulations: Driving Score (DS) reflects the overall ADS performance that combines achievements and penalties. Route Completion (RC) measures the percentage of the route completed. Penalties include Collisions with Pedestrians (CP) or Vehicles (CV), Off-Road Infractions (ORI), Red Light Infractions (RLI), and Stop Sign Infractions (SSI).

To handle the multi-camera setup, we applied our methodology to three camera views and ensured consistency across them by using the same distilled model configured to augment each image. Although our methodology does not operate on LiDAR point clouds, preserving the semantic consistency of the visual representations ensures that the LiDAR data by the simulator remain consistent with the augmented camera views.

Urban driving tasks, such as overtaking and collision avoidance, require the consideration of four additional semantic classes: pedestrians, vehicles, traffic signs, and traffic lights. These additional classes were addressed during both augmentation and validation. During augmentation, we configured a strategy based on inpainting to also preserve those parts of the image. During validation, we applied OC-TSS to each semantic class and considered an augmentation valid only if all semantic classes were preserved within the threshold (0.9).

We replicated the experiments on the effectiveness of the augmented domains for failure exposure (RQ$_2$) and measured the associated overhead (RQ$_3$), using the same augmented domains as in the experiments with Udacity. CARLA supports various closed-loop urban maps for testing ADS. We considered Town05, one of the default maps provided by

CARLA, with its default environmental configuration (e.g., sunny weather). Within Town05, we considered ten different scenarios (details are provided in the replication package).

Table VI reports the average effectiveness results in all scenarios and in all domains. As those scenarios are already designed to challenge the ADS, we provide the results with and without augmentation. The results show that our approach successfully exposed new misbehaviors of the ADS, even in complex urban scenarios.

Regarding Route Completion (RC), InterFuser with no augmentation did not consistently achieve 100%. Completion rates varied, with some runs at 98.5% and others at 13.2%. Manual inspection of the logs revealed that the ego-vehicle gets "stuck" due to traffic jams caused by other vehicles that block intersections. With the baseline simulator (no augmentation), no infractions and collisions were detected. With our neural augmentations, we discovered 25 previously unknown red light infractions and 3 previously unknown collisions with vehicles.

Concerning the overhead (see Table VII), our methodology provides significant benefits also in CARLA. First, we measured the time taken to augment the three camera views. The results show that the domain augmentation approaches required from 3.4s (Inpainting) to 11.4s (Inpainting with Refinement), while the knowledge-distilled model took only 82.0ms (up to 138× faster). Then, we measured the time required to execute a test scenario to evaluate the overall testing overhead introduced by the augmented domains. To ensure that the duration of the test was not influenced by external factors, such as vehicles blocking intersections, pedestrian crossings, or red lights, we used a controlled scenario without such elements. With the diffusion models, the test runtime increased significantly, with observed test runs of 150.7 minutes (Instruction-editing), 358.5 minutes (Inpainting), and 474.5 minutes (Inpainting with Refinement) for a single test run. In contrast, using the lighter knowledge-distilled model, the runtime increased by only 19.9 minutes, a 20.0% increase over the baseline without augmentation. We believe that the overhead is higher than single-image ADS due to the higher number of images to be augmented (three) and their larger size (600×800 height×width).

> **RQ$_4$**: *Our approach generalizes to complex urban driving environments in CARLA and InterFuser, a multi-modal ADS. We discovered 25 red light infractions and 3 vehicle collisions, which were not detected in the original simulator. The knowledge-distilled model reduced the augmentation time to 82ms, resulting in an overhead of only 20.0%.*

G. Threats to Validity

Internal validity. We utilized widely used model architectures and simulators from the literature. The selection of the semantic validity threshold poses another potential threat. In this study, we adopted a conservative threshold to minimize the inclusion of semantically invalid images. We also assumed that domain augmentations preserve driving action labels. Although similar work has made this assumption [9], [11], we explicitly excluded domains that are unlikely to maintain the integrity of the label, such as snow or rain, as these conditions can alter the dynamics and driving style of the vehicle due to changes in friction or traction.

External validity. We considered limited instances of diffusion models. To address this threat, we selected state-of-the-art diffusion models of different types that consistently improved the simulator across ODDs.

Reproducibility. To support reproducibility, all of our data, including the code of the diffusion models and our enhanced simulator, are available in our replication package [23].

V. RELATED WORK

A. Test Generation for Autonomous Driving

Existing work leverages the ability of driving simulators to create diverse driving scenes for *scenario-based testing* of ADS [64]–[66]. Generated scenarios [67] include a wide range of driving conditions, such as sudden lane changes, adverse weather, or interactions with other vehicles and pedestrians [67]. Majumdar et al. [68] propose Paracosm, a tool that allows users to programmatically define complex driving scenarios. Woodlief et al. [69] propose a framework that abstracts sensor inputs to coverage domains that account for the spatial semantics of a scene. A new technique called Instance Space Analysis was recently proposed to identify the significant features of test scenarios that affect the ability to reveal the unsafe behavior of ADS [70].

All of these test generators operate within a confined range of predefined ODD scenarios, including specific weather conditions, background locations, and times of day, to maximize the number of failures *within* these predefined scenarios. Our approach seeks to considerably broaden the range of ODD conditions *beyond* those currently available. Our methodology is complementary and can be integrated with existing test generators to enhance their effectiveness without modifications.

B. Offline Testing with Generative AI

Approaches based on GenAI focus on augmenting existing image datasets by introducing variations like adverse weather or other visual elements [71], [72]. For example, Zhang et al. [9] propose DeepRoad, a solution that utilizes UNIT [38] to generate test images by altering the weather from sunny to foggy or snowy. Pan et al. [10] present a method that leverages CycleGAN [21] combined with techniques to synthesize different fog levels with controllable intensity and direction in driving images. Li et al. [11] propose TACTICS, an ADS testing framework that uses search-based strategies to identify critical environmental conditions and employs MUNIT [73] to

reproduce these conditions in existing driving images. Attaoui et al. [74] combine GenAI and search-based testing to test the semantic segmentation module of an ADS. Other approaches augment existing test images with diffusion models [75]. Zhao et al. [76] exploit semantic segmentation maps and a conditional generative model, ControlNet [31], to generate high-quality synthetic images. Xu et al. [77] employed a fine-tuned Stable Diffusion [18] to create controllable traffic signs.

Although these approaches assess the behavior of the ADS, they target *model-level testing* and measure the discrepancy between predicted and ground truth values [78]. In contrast, we focus on *system-level testing*. Our application of GenAI as a rendering engine within a physics-based simulator constitutes a novel contribution to the state-of-the-art in ADS testing.

C. Data-driven Simulation

Neural simulators [79] consist of data-driven approaches in which GenAI is used to produce a continuous stream of driving images. Unlike traditional simulators [80], which rely on game-based 3D rendering and physics models, neural simulators employ a learnable world model [81] to represent the environment of the ADS and target novel view synthesis (e.g., bird-eye's view) [82]. For example, DriveGAN [15] utilizes GANs to create driving scenarios with controllable weather conditions, traffic objects, and backgrounds. DriveDreamer [83] and GAIA-1 [84] employ diffusion models to generate real-world driving scenarios. UniSim [85] is a neural closed-loop sensor simulator that transforms a single recorded log from an ADS into a realistic multi-sensor simulation.

While neural simulators offer an improvement in generating novel and realistic training data, their lack of a physical representation limits their applicability for testing. This deficiency can produce inaccurate failure simulations (e.g., collisions), resulting in false positives. Thus, neural simulators are not the best choice for testing ADS systems at the system level.

To address this limitation, our approach integrates neural rendering and GenAI techniques with a physics simulator. This combination enables effective testing with precise failure determination while expanding the ODD conditions for testing.

VI. CONCLUSIONS AND FUTURE WORK

We have generated new ODD scenarios for ADS testing using diffusion models and instruction editing operations. We addressed the validity of the augmented images by creating an automated semantic validator, which was found to be extremely accurate in a human study, with as few as 3% invalid images incorrectly regarded as valid. We have considered the realism of the augmented images by conducting a human study that indicated the Inpainting with Refinement strategy as the technique that generates the most realistic images. We have reduced the simulation overhead by introducing a CycleGAN model that takes advantage of knowledge distillation. Most importantly, we have shown that our approach can expose ADS failures even in domains close to the training domain and in track sectors that were deemed error-free when considering only simulator-generated test scenarios.

REFERENCES

[1] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[2] "Iso 34504 — road vehicles — test scenarios for automated driving systems — scenario categorization," Standard, February 2024.

[3] X. Hu, S. Li, T. Huang, B. Tang, R. Huai, and L. Chen, "How simulation helps autonomous driving: A survey of sim2real, digital twins, and parallel intelligence," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, 2024.

[4] "Unity3d." https://unity.com, 2024.

[5] EpicGames, "Unreal engine." https://www.unrealengine.com/en-US, 2024.

[6] A. Gambi, M. Mueller, and G. Fraser, "Automatically testing self-driving cars with search-based procedural content generation," in *Proceedings of ISSTA 2019*. ACM, 2019, pp. 318–328.

[7] V. Riccio and P. Tonella, "Model-Based Exploration of the Frontier of Behaviours for Deep Learning System Testing," in *Proceedings of ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020.

[8] A. Stocco, B. Pulfer, and P. Tonella, "Mind the Gap! A Study on the Transferability of Virtual vs Physical-world Testing of Autonomous Driving Systems," *IEEE Transactions on Software Engineering*, 2022.

[9] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, "Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems," in *Proceedings of the ACM/IEEE International Conference on Automated Software Engineering*. ACM, 2018, pp. 132–142.

[10] Y. Pan, H. Ao, and Y. Fan, "Metamorphic testing for autonomous driving systems in fog based on quantitative measurement," in *Proceedings of the IEEE International Conference on Software Quality, Reliability and Security*, 2021, pp. 30–37.

[11] Z. Li, M. Pan, T. Zhang, and X. Li, "Testing dnn-based autonomous driving systems under critical environmental conditions," in *Proceedings of the International Conference on Machine Learning*, vol. 139. PMLR, 2021, pp. 6471–6482.

[12] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, "Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. ACM, 2018, pp. 132–142. [Online]. Available: http://doi.acm.org/10.1145/3238147.3238187

[13] M. Biagiola, A. Stocco, V. Riccio, and P. Tonella, "Two is better than one: Digital siblings to improve autonomous driving testing," 2023.

[14] A. Stocco, B. Pulfer, and P. Tonella, "Model vs system level testing of autonomous driving systems: a replication and extension study," *Empir. Softw. Eng.*, vol. 28, no. 3, p. 73, 2023.

[15] S. W. Kim, J. Philion, A. Torralba, and S. Fidler, "Drivegan: Towards a controllable high-quality neural simulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2021, pp. 5820–5829.

[16] T. Brooks, A. Holynski, and A. A. Efros, "Instructpix2pix: Learning to follow image editing instructions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2023, pp. 18 392–18 402.

[17] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J. Zhu, and S. Ermon, "Sdedit: Guided image synthesis and editing with stochastic differential equations," in *Proceedings of the International Conference on Learning Representations*, 2022.

[18] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2022, pp. 10 674–10 685.

[19] 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*. IEEE, 2024, p. 12 pages.

[20] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, 2015.

[21] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2242–2251.

[22] H. Shao, L. Wang, R. Chen, H. Li, and Y. Liu, "Safety-enhanced autonomous driving using interpretable sensor fusion transformer," in *Proceedings of the Conference on Robot Learning*, vol. 205. PMLR, 2022, pp. 726–737.

[23] "Replication package." https://tinyurl.com/replication-package-icse-2025, 2025.

[24] T. R. I. . International Organization for Standardization, "Road vehicles - safety of the intended functionality," 2019.

[25] E. Union, "Un regulation no 157 – uniform provisions concerning the approval of vehicles with regards to automated lane keeping systems [2021/389]," 2021.

[26] F. U. Haq, D. Shin, S. Nejati, and L. C. Briand, "Comparing offline and online testing of deep neural networks: An autonomous car case study," in *Proceedings IEEE International Conference on Software Testing, Validation and Verification*. IEEE, 2020, pp. 85–95.

[27] J. Li, C. Zhang, W. Zhu, and Y. Ren, "A comprehensive survey of image generation models based on deep learning," *Annals of Data Science*, 2024.

[28] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, 2020.

[29] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion models in vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 9, pp. 10 850–10 869, 2023.

[30] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," in *Proceedings of the International Conference on Machine Learning*, vol. 139. PMLR, 2021, pp. 8821–8831.

[31] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*. IEEE, 2023, pp. 3813–3824.

[32] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars." *CoRR*, vol. abs/1604.07316, 2016.

[33] J. Chen, J. Yu, C. Ge, L. Yao, E. Xie, Z. Wang, J. Kwok, P. Luo, H. Lu, and Z. Li, "Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis," in *Proceedings of the International Conference on Learning Representations*, 2024.

[34] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

[35] C. Mou, X. Wang, L. Xie, Y. Wu, J. Zhang, Z. Qi, and Y. Shan, "T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models," in *Proceedings of the Conference on Artificial Intelligence, AAAI 2024*, 2024, pp. 4296–4304.

[36] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 9351. Springer, 2015, pp. 234–241.

[37] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Álvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *CoRR*, vol. abs/2105.15203, 2021.

[38] M. Liu, T. M. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 700–708.

[39] Udacity, "A self-driving car simulator built with Unity," https://github.com/udacity/self-driving-car-sim, 2017, online; accessed 25 October 2023.

[40] A. Stocco, M. Weiss, M. Calzana, and P. Tonella, "Misbehaviour prediction for autonomous driving systems," in *Proceedings of the International Conference on Software Engineering*. ACM, 2020, pp. 359–371.

[41] Udacity, "Udacity self-driving car's challenge," https://github.com/udacity/self-driving-car/, 2017, online; accessed 18 August 2019.

[42] A. Stocco, M. Weiss, M. Calzana, and P. Tonella, "Misbehaviour prediction for autonomous driving systems," in *Proceedings of 42nd International Conference on Software Engineering*. ACM, 2020, p. 12 pages.

[43] M. B. et al., "End to end learning for self-driving cars," *Arxiv*, vol. abs/1604.07316, 2016.

[44] Team Chauffeur, "Steering angle model: Chauffeur," https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/chauffeur, 2016, online; accessed 18 August 2019.

[45] Team Epoch, "Steering angle model: Epoch," https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/cg23, 2016, online; accessed 18 August 2019.

[46] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the International Conference on Software Engineering*. ACM, 2018.

[47] M. Hussain, N. Ali, and J.-E. Hong, "Deepguard: A framework for safeguarding autonomous driving systems from inconsistent behaviour," *Automated Software Engg.*, vol. 29, no. 1, may 2022.

[48] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM, 2017, pp. 1–18. [Online]. Available: http://doi.acm.org/10.1145/3132747.3132785

[49] I. Sonata, Y. Heryadi, A. Wibowo, and W. Budiharto, "End-to-end steering angle prediction for autonomous car using vision transformer," *CommIT (Communication and Information Technology) Journal*, vol. 17, pp. 221–234, 09 2023.

[50] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proceedings of the International Conference on Learning Representationss*, 2014.

[51] W. Zhao, L. Bai, Y. Rao, J. Zhou, and J. Lu, "Unipc: A unified predictor-corrector framework for fast sampling of diffusion models," in *Advances in Neural Information Processing Systems*, 2023.

[52] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8107–8116.

[53] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *NIPS*, 2017.

[54] S. J. Stratton, "Population research: Convenience sampling strategies," *Prehospital and Disaster Medicine*, vol. 36, no. 4, p. 373–374, 2021.

[55] A. Dosovitskiy, G. Ros, F. Codevilla, A. López, and V. Koltun, "CARLA: an open urban driving simulator," *CoRR*, vol. abs/1711.03938, 2017. [Online]. Available: http://arxiv.org/abs/1711.03938

[56] S. Kim, M. Liu, J. J. Rhee, Y. Jeon, Y. Kwon, and C. H. Kim, "DriveFuzz," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. ACM, nov 2022. [Online]. Available: https://doi.org/10.1145/3548606.3560558

[57] M. Cheng, Y. Zhou, and X. Xie, "Behavexplor: Behavior diversity guided testing for autonomous driving systems," in *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*. Association for Computing Machinery, 2023, p. 488–500. [Online]. Available: https://doi.org/10.1145/3597926.3598072

[58] G. Li, Y. Li, S. Jha, T. Tsai, M. Sullivan, S. K. S. Hari, Z. Kalbarczyk, and R. Iyer, "Av-fuzzer: Finding safety violations in autonomous driving systems," in *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*, 2020, pp. 25–36.

[59] W. Zhang, M. Elmahgiubi, K. Rezaee, B. Khamidehi, H. Mirkhani, F. Arasteh, C. Li, M. A. Kaleem, E. R. Corral-Soto, D. Sharma, and T. Cao, "Analysis of a modular autonomous driving architecture: The top submission to CARLA leaderboard 2.0 challenge," *CoRR*, vol. abs/2405.01394, 2024.

[60] CARLA team, "CARLA: leaderboard," 2023. [Online]. Available: https://leaderboard.carla.org/leaderboard/

[61] X. Jia, P. Wu, L. Chen, J. Xie, C. He, J. Yan, and H. Li, "Think twice before driving: Towards scalable decoders for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 983–21 994.

[62] X. Jia, Y. Gao, L. Chen, J. Yan, P. L. Liu, and H. Li, "Driveadapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 7953–7963.

[63] B. Jaeger, K. Chitta, and A. Geiger, "Hidden biases of end-to-end driving models," *arXiv preprint arXiv:2306.07957*, 2023.

[64] S. T. et al., "A survey on automated driving system testing: Landscapes and trends," *ACM Transactions on Software Engineering and Methodologies*, vol. 32, no. 5, 2023.

[65] G. Lou, Y. Deng, X. Zheng, M. Zhang, and T. Zhang, "Testing of autonomous driving systems: where are we and where should we go?" in *Proceedings of the ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 2022, pp. 31–43.

[66] Z. Zhong, Y. Tang, Y. Zhou, V. de Oliveira Neves, Y. Liu, and B. Ray, "A survey on scenario-based testing for automated driving systems in high-fidelity simulation," *Arxiv*, 2021.

[67] J. Jullien, C. Martel, L. Vignollet, and M. Wentland, "Openscenario: A flexible integrated environment to develop educational activities based on pedagogical scenarios," in *Proceedings of the IEEE International Conference on Advanced Learning Technologies*. IEEE Computer Society, 2009, pp. 509–513.

[68] R. Majumdar, A. S. Mathur, M. Pirron, L. Stegner, and D. Zufferey, "Paracosm: A test framework for autonomous driving simulations," in *Proceedings of the International Conference*, vol. 12649. Springer, 2021, pp. 172–195.

[69] T. Woodlief, F. Toledo, S. Elbaum, and M. B. Dwyer, "S3c: Spatial semantic scene coverage for autonomous vehicles," in *ICSE*. Association for Computing Machinery, 2024. [Online]. Available: https://doi.org/10.1145/3597503.3639178

[70] N. Neelofar and A. Aleti, "Identifying and explaining safety-critical scenarios for autonomous vehicles via key features," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 4, apr 2024. [Online]. Available: https://doi.org/10.1145/3640335

[71] V. Ostankovich, R. Yagfarov, M. Rassabin, and S. Gafurov, "Application of cyclegan-based augmentation for autonomous driving at night," in *Proceedings of the IEEE International Conference on Nonlinearity, Information and Robotics*, 2020, pp. 1–5.

[72] K. Gao, J. Wang, B. Wang, R. Wang, and J. Jia, "UAV test data generation method based on cyclegan," in *Proceedings of the International Conference on Dependable Systems and Their Applications*. IEEE, 2021, pp. 338–343.

[73] X. Huang, M. Liu, S. J. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European Conference on Computer Vision*, vol. 11207. Springer, 2018.

[74] M. O. Attaoui, F. Pastore, and L. Briand, "Search-based dnn testing and retraining with gan-enhanced simulations," 2024. [Online]. Available: https://arxiv.org/abs/2406.13359

[75] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M. Yang, "Diffusion models: A comprehensive survey of methods and applications," *ACM Comput. Surv.*, vol. 56, no. 4, pp. 105:1–105:39, 2024.

[76] H. Zhao, Y. Wang, T. Bashford-Rogers, V. Donzella, and K. Debattista, "Exploring generative AI for sim2real in driving data synthesis," *Arxiv*, 2024.

[77] M. Xu, D. Niyato, J. Chen, H. Zhang, J. Kang, Z. Xiong, S. Mao, and Z. Han, "Generative ai-empowered simulation for autonomous driving in vehicular mixed reality metaverses," *IEEE J. Sel. Top. Signal Process.*, vol. 17, no. 5, pp. 1064–1079, 2023.

[78] A. Stocco, B. Pulfer, and P. Tonella, "Model vs system level testing of autonomous driving systems: A replication and extension study," *Empirical Softw. Engg.*, vol. 28, no. 3, may 2023. [Online]. Available: https://doi.org/10.1007/s10664-023-10306-x

[79] Z. Zhu, X. Wang, W. Zhao, C. Min, N. Deng, M. Dou, Y. Wang, B. Shi, K. Wang, C. Zhang, Y. You, Z. Zhang, D. Zhao, L. Xiao, J. Zhao, J. Lu, and G. Huang, "Is sora a world simulator? A comprehensive survey on general world models and beyond," *Arxiv*, 2024.

[80] A. Dosovitskiy, G. Ros, F. Codevilla, A. M. López, and V. Koltun, "CARLA: an open urban driving simulator," in *Proceedings of the Annual Conference on Robot Learning*, vol. 78. PMLR, 2017.

[81] Y. Guan, H. Liao, Z. Li, G. Zhang, and C. Xu, "World models for autonomous driving: An initial survey," *Arxiv*, 2024.

[82] W. Xia and J. Xue, "A survey on deep generative 3d-aware image synthesis," *ACM Comput. Surv.*, vol. 56, no. 4, pp. 90:1–90:34, 2024.

[83] X. Wang, Z. Zhu, G. Huang, X. Chen, and J. Lu, "Drivedreamer: Towards real-world-driven world models for autonomous driving," *Arxiv*, 2023.

[84] A. Hu, L. Russell, H. Yeo, Z. Murez, G. Fedoseev, A. Kendall, J. Shotton, and G. Corrado, "GAIA-1: A generative world model for autonomous driving," *Arxiv*, 2023.

[85] Z. Yang, Y. Chen, J. Wang, S. Manivasagam, W. Ma, A. J. Yang, and R. Urtasun, "Unisim: A neural closed-loop sensor simulator," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2023, pp. 1389–1399.