Search-based DNN Testing and Retraining with GAN-enhanced Simulations

Mohammed Oualid Attaoui, Fabrizio Pastore, Lionel C. Briand

***Abstract*—In safety-critical systems (e.g., autonomous vehicles and robots), Deep Neural Networks (DNNs) are becoming a key component for computer vision tasks, particularly semantic segmentation. Further, since DNN behavior cannot be assessed through code inspection and analysis, test automation has become an essential activity to gain confidence in the reliability of DNNs. Unfortunately, state-of-the-art automated testing solutions largely rely on simulators, whose fidelity is always imperfect, thus affecting the validity of test results. To address such limitations, we propose to combine meta-heuristic search, used to explore the input space using simulators, with Generative Adversarial Networks (GANs), to transform the data generated by simulators into realistic input images. Such images can be used both to assess the DNN accuracy and to retrain the DNN more effectively. We applied our approach to a state-of-the-art DNN performing semantic segmentation, in two different case studies, and demonstrated that it outperforms a state-of-the-art GAN-based testing solution and several other baselines. Specifically, it leads to the largest number of diverse images leading to the worst DNN accuracy. Further, the images generated with our approach, lead to the highest improvement in DNN accuracy when used for retraining. In conclusion, we suggest to always integrate a trained GAN to transform test inputs when performing search-driven, simulator-based testing.**

***Index Terms*—GAN-based testing, Simulator-based testing, DNN-based systems testing**

I. Introduction

Deep Neural Networks (DNNs) are an effective solution to automate tasks that require vision capabilities. For this reason, several companies developing cyber-physical systems (CPS), are making large R&D investments to rely on DNNs to automate tasks that are currently manual. Examples in the automotive sector include driving assistance systems (e.g., obstacles detection, traffic sign detection, autonomous driving) and interior car monitoring (e.g., drowsiness detection or unsupervised child detection). In other sectors, key developments are observed in robotics and space, for example to develop autonomous vehicles capable of exploring caves and planets' surfaces.

When a DNN that processes camera images is used to drive a CPS, we must gain sufficient confidence that it can adequately respond to all foreseeable inputs. To achieve this objective, research has focused on combining meta-heuristic search with simulators [21], [23], [27], [28], [44], [69]. The rationale is that, under the assumption that simulators can accurately create images corresponding to a realistic scene, meta-heuristic search enables the cost-effective search of inputs leading to failures. What enables test automation is the fact that simulators, in addition to generate images according to the parameters selected by the search algorithm, can be used to automatically assess the quality of the DNN output for each generated input. For example, the performance of steering angle DNNs can be assessed by relying on the distance between the ego vehicle and the lane separator line, both provided by the simulator; indeed, the closer the vehicle gets to the separator line, the more likely it will cross the lane. Although such search-based approaches effectively generate images leading to DNN failures, the validity of such DNN assessment highly depends on the simulator fidelity. For example, recent work has demonstrated that the fidelity gap of simulators leads to different reliability results in simulation-based and real-world testing [28], [50].

The limited fidelity of a simulator may thus hinder the detection of DNN shortcomings. For example, the road landscapes generated by the AirSim simulator [47], which resemble residential areas in the USA, are very different from the landscape of European cities and towns. Based on our interactions with industry partners in the space and automotive sectors [5], [19], [20], we noticed that high-fidelity simulators are lacking for many environments in which DNNs can be applied, including for example car interiors (e.g., for in-cabin monitoring DNNs) and space landscapes (e.g., to drive Mars and Lunar rovers).

To accurately test DNNs, ensuring that we explore the input space relying on realistic input images, we propose *DNN testing with GAN-enhanced simulations and search (DESIGNATE)*. We rely on meta-heuristic search to effectively drive a simulator towards the generation of diverse inputs leading to failures but, instead of testing the DNN with simulator images, we leverage Generative Adversarial Networks (GANs) to transform the simulator output into a more realistic image that resembles real-world data distributions. GANs are a promising candidate to address the simulator fidelity gap because they have shown to be effective in image-to-image translation, that is mapping images from one domain (e.g., sketches) to images in another (e.g., photographs) [31], [57]. For testing, GANs have been mainly adopted to introduce realistic perturbations in input images [34], [58], [65] and to estimate sensory data [50], [63]. DESIGNATE is the first technique integrating GANs into search-based testing with simulators.

We assume that, because the realistic images are generated from the simulated ones, both the simulator and realistic images share the same ground truth (i.e., the DNN should generate similar outputs from them), thus enabling test au-

M. O. Attaoui and F. Pastore are with the SnT Centre, University of Luxembourg.
E-mail: mohammed.attaoui@uni.lu, fabrizio.pastore@uni.lu
L. Briand is with the Lero Research Ireland Centre and University of Limerick, Limerick, Ireland, and the School of EECS, University of Ottawa, Ottawa, Canada.
E-mail: lbriand@uottawa.ca

tomation by assessing the DNN output based on the information provided by the simulator. Further, realistic images leading to DNN failures can be used to retrain a DNN to improve its reliability in the field, something which may be more difficult to achieve with simulator images because of on their fidelity. However, to address this challenge, DESIGNATE can test a road scene segmentation DNN by relying on AirSim to generate landscapes of city roads with traffic and buildings, and then leverage a GAN to transform those landscapes into realistic images with European cities.

Inspired by work on DNN testing [28], [44], we rely on genetic algorithms to cost-effectively identify failure-inducing images with GANs; our objective is to identify simulator parameters leading to simulator outputs (e.g., an image of a simulated city road) that, after being transformed into realistic images using a GAN, lead to DNN failures. Since we ideally aim at identifying all the situations in which the DNN fails, to both identify failures and maximize input diversity, we combine two fitness functions. The first one measures the accuracy of the DNN output whereas the second one measures the similarity across generated realistic images. To test a DNN with diverse scenes being depicted in the input images, we are interested in generating images that differ in terms of the items being displayed and not just a few, scattered, unmatched pixels. For this reason, we introduce a feature-based distance metric that leverages transfer learning [5].

We conducted an empirical investigation with a state-of-the-art, third-party DNN for road segmentation. Our results show that DESIGNATE outperforms TACTIC [34] and DeepJanus [44], two state-of-the-art DNN testing approaches, and several baselines including random search, single objective search, and multi-objective search without GANs. DESIGNATE generates inputs leading to the worst DNN reliability and maximizes input diversity. Further, we demonstrate that the inputs generated by DESIGNATE better support DNN retraining, compared to competing approaches, with an increase in DNN accuracy of eight percentage points. In addition to our road segmentation case study, in the context of the Mars rover self-driving system, we also demonstrate the applicability and benefits of our approach to object detection on Mars' surface, using the AI4MARS dataset [51] and a custom-built simulator leveraging Unreal Engine [17] and Airsim. Based on our results across both case studies, we conclude that GANs should be combined with simulators when testing and improving vision-based DNNs as it leads to more effective and diverse testing, as well as better retraining.

Our paper proceeds as follows. Section II introduces background techniques. Section III describes DESIGNATE. Section IV reports on our empirical study. Section V discusses related work. Section VI concludes the paper.

II. Background

A. Terminology

In this paper, we rely on the following definitions:

*A **scene** describes a snapshot of the environment including the scenery and dynamic elements, as well as all actors' and observers' self-representations, and the relationships among those entities* [53]. In our context, where the DNN under test processes pictures taken by a camera, a scene is a single picture taken by a camera or generated by a simulator.

Fig. 1: Example of images from the cityscapes dataset showing the same situation (a turning road, buildings on the side along with parked cars).

*A **situation** is the entirety of circumstances, which are to be considered for the selection of an appropriate behavior pattern at a particular point of time [53].* For vision DNNs in autonomous driving (e.g., semantic segmentation, steering angle prediction), since the DNN prediction depends on the objects depicted in an image, a situation captures commonalities among similar images leading to the same DNN prediction. For example, pictures with a turning road, buildings on the side along with parked cars, may belong to the same situation (see Figure 1).

*A **scenario** describes the temporal development between several scenes in a sequence of scenes. Every scenario starts with an initial scene. Actions and events characterize the temporal development in a scenario. Other than a scene, a scenario spans a certain amount of time [53].* In our context, a scenario is a sequence of images generated by the simulator.

B. Semantic Segmentation

Semantic segmentation is a computer vision task that involves assigning a class label to each pixel in an image. Unlike other forms of image recognition or classification, semantic segmentation provides a fine-grained understanding of an image's content by labeling each pixel with a corresponding class, such as "road," "car," "person," and "building,".

Semantic segmentation is often performed with convolutional neural networks (CNNs) that learn to capture spatial relationships, object boundaries, and contextual information within an image, enabling them to differentiate between different objects and delineate their precise outlines. In our study, we consider the state-of-the-art DeepLabV3+ architecture [13]. DeepLabV3+ is renowned for its accuracy and efficiency in pixel-level semantic segmentation tasks [37], [56].

C. Simulated Environments

Simulators replicate real-world scenarios, allowing DNNs to be tested in a controlled and reproducible environment. In our work, we rely on two simulators:

1) AirSim: AirSim is an open-source simulator for autonomous vehicles and drones developed by Microsoft [47]. It provides a realistic environment for testing and training AI algorithms using a variety of sensors, including cameras, lidars, and GPS. AirSim can be controlled through a dedicated API, which enables its use for DNN testing. In our context, we rely on the features for controlling the positioning of the ego vehicle in the simulated environment and generating a picture of the landscape in front of the ego vehicle. Further, for each generated image, AirSim provides a semantic segmentation image where each pixel's color directly corresponds to a specific category or class, including roads, buildings, vehicles, pedestrians, or sky. We use these images as *ground truth* to assess the reliability of DNNs in road scene segmentation.

2) MarsSim: Focusing on extraterrestrial object detection, specifically on the Martian surface, we built a simulator that generate images resembling the ones in the AI4MARS dataset by combining Unreal Engine with the the same plugins that Airsim uses. This simulator creates realistic images of Mars terrain, which, when enhanced with a GAN, enables the generation of diverse training data for testing DNNs in Mars exploration contexts. Including the Airsim API enables us to control the simulator by changing the position of the camera. To create an authentic Martian environment, we modeled objects that match those found in the AI4MARS dataset: rocks, bedrock, sand, and Martian soil, each designed with textures and colors reflective of actual Mars imagery. We also incorporated variations in object size, shape, and placement to reflect the irregular distribution of terrain features typically observed on Mars. Additionally, we introduced terrain undulations and slopes to simulate the topographical challenges of Mars, such as craters, cliffs, and valleys, which pose unique navigational challenges for autonomous rovers.

D. GAN-based Image-to-image Translation

Image-to-image translation maps images from one domain (e.g., sketches, edges, or semantic labels) to images in another domain (e.g., photographs, paintings, or architectural layouts). This transformation process involves learning the complex and non-linear mappings between the two domains, allowing for a seamless transition while preserving essential attributes.

Generative Adversarial Networks (GANs) are a key technology for image-to-image translation. GANs consist of two interconnected DNNs - the generator and the discriminator. The generator is responsible for producing synthetic images, while the discriminator aims to differentiate between real images from the target domain and the generated images. Through a competitive process, the generator progressively improves its ability to produce more realistic images, and the discriminator becomes more adept at distinguishing real and generated images.

Example GANs for image-to-image translation are Pix2Pix [32], with its high definition extension Pix2PixHD [57], which are trained with pairs of images from the two domains, and CycleGAN [68], which does not require paired images. Other GANs are UNIT [36] and MUNIT [31]. UNIT includes two domain image encoders, two domain image generators, and two domain adversarial discriminators. MUNIT decomposes an image representation into domain-independent content code and domain-specific style code. The content code captures the underlying structure and semantics of the image that remain consistent across different domains. For example, in road scene images, the content code includes the road layout, lane positions, vehicles, traffic signs, pedestrians, and buildings. On the other hand, the style code captures the stylistic attributes unique to a particular domain. In the context of automotive images, style codes might include night-vision images, different weather conditions, and synthetic images from simulators. MUNIT translates images to other domains by recombining content code with style code sampled from the target domain. Related work leveraged UNIT and MUNIT to change weather conditions in real-world images [59], [65], which is a different objective than ours. CycleGAN has instead been used to automatically generate sensor data [50]. After a preliminary assessment of both CycleGAN and Pix2PixHD to generate realistic images from segmentation maps produced by the AirSim simulator, we selected Pix2PixHD for our work since it led to better results (i.e., higher-fidelity images). Figure 2 illustrates an example of image-to-image translation of a segmentation map produced by AirSim into a realistic image, using the Pix2PixHD model [32]

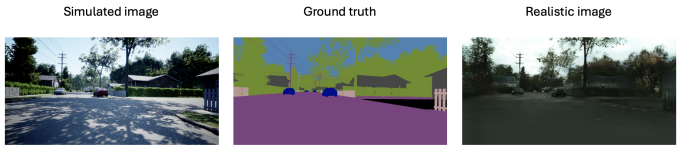

Fig. 2: Examples of simulator image, ground truth (i.e., segmentation map provided by the simulator), and realistic image generated by Pix2PixHD from the ground truth of AirSim.

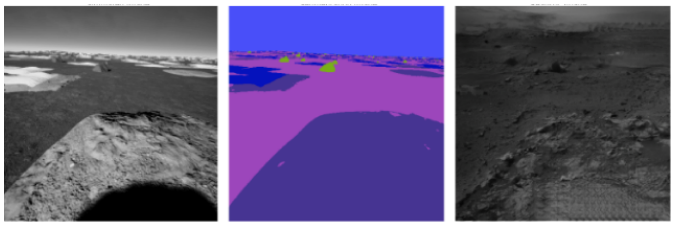

Fig. 3: Examples of a Mars simulator's simulated image, its ground truth, and the realistic image generated from it by Pix2PixHD.

III. The DESIGNATE Approach

DESIGNATE supports the testing and improvement of vision DNNs by generating realistic but synthetic inputs belonging to distinct situations. Such realistic inputs are generated by combining meta-heuristic search, simulators, and GANs. Simulators enable the inexpensive generation of input images and corresponding ground truth information (e.g., correct segmentation mask), while meta-heuristic search enables the exploration of the input space to exercise distinct situations

increasingly likely to lead to mispredictions. Finally, GANs enable the generation of high-fidelity images from the simulator data, thus maximizing the likelihood that testing results are representative of what can be observed with real-world inputs, thus ensuring that the mispredictions observed during testing are due to DNN limitations that can be experienced in the real-world and not the fidelity gap of simulators. Further, realistic images can be used to retrain the DNN under test to improve its reliability. Last, the images generated by DESIGNATE belong to distinct situations, thus enabling the identification of diverse DNN limitations and their improvement.

Figure 4 provides an overview of DESIGNATE, which consists of three components:

1) *GAN-based input generation component.* It receives as input the configuration parameter values to be used to generate, with a simulator, *scene data* (i.e., an image and the corresponding segmentation map). The scene data is then used to generate a *realistic image* with a GAN trained on a specific reference domain. For our automotive case study, we used the Pix2PixHD GAN trained on the CityScape dataset to generate realistic road scenes from the segmentation map produced by the simulator. For our Mars case study, we utilized the AI4MARS dataset to train the GAN, and the simulator was built using Unreal Engine and Airsim to generate Martian surface images.
2) *Fitness computation component.* Since we aim to identify situations that are both unsafe (i.e., leading to DNN mispredictions) and diverse, we apply two distinct fitness functions: $F_{accuracy}$, which measures how accurate is the DNN prediction on the generated realistic image (e.g., the accuracy of the semantic segmentation), and $F_{similarity}$, which measures how similar a generated image is from the already generated ones. These functions are used by a minimization algorithm to address our objectives, as described next.
3) *Search Algorithm.* We rely a multi-objective search algorithm to find a population of diverse situations where the DNN fares poorly, thereby suggesting these scenarios are not well captured by the real-world DNN training set. In our search algorithm, an individual is captured by the set of parameters used to deterministically generate, using a simulator, scene data, and to derive a realistic image from such data. For optimization purposes, for each individual, we store the simulator parameter values, the scene data, and the realistic image.

The output of DESIGNATE is a set of diverse and realistic images leading to poor DNN reliability. They should be inspected for safety-analysis purposes; specifically, they enable understanding the situations in which the DNN mispredicts, determine their likelihood based on domain knowledge, and identify countermeasures. Also, they can be used to retrain the DNN and improve its performance. The following sections provide details about our components.

A. GAN-based input generation

We generate scene data (i.e., images with simulated scenes and corresponding segmentation masks) by relying on the AirSim simulator (Section II-C); however, our approach can be applied with simulators having similar capabilities (e.g., Carla [10] and BeamNG [7]). However, one must ensure that the generated scenes are realistic. In our case, to assess DNNs that control a vehicle (e.g., segmentation of the images in front of the vehicle, or steering angle prediction), it is necessary to generate scenes that mimic what can be observed by a car on the road. Consequently, we control the generation of scene data through two parameters: orientation and position of the car. To test other DNNs (e.g., visual navigation for drones), a different set of parameters would be considered (e.g., altitude might be included).

We then generate realistic images from the ground truth information provided by the simulator. In our experiments, such ground truth is the segmentation mask, but could be different information in other contexts. To this end, we utilize a Pix2PixHD DNN [57] that we train using the same dataset used for the DNN under test; such dataset consists of pairs $\langle \text{image}, \text{expected output} \rangle$. For example, in our experiments, we considered a DNN trained on a dataset of pairs $\langle \text{European road image}, \text{segmentation mask} \rangle$, which we use to train Pix2PixHD to generate images of roads belonging to European cities from segmentation masks.

During testing, DESIGNATE provides the ground truth of the simulated images to the trained Pix2PixHD to generate a realistic images; in our experiments, DESIGNATE provides to Pix2PixHD the segmentation mask produced by AirSim in order to generate realistic images resembling real-world images from Cityscapes.

B. Fitness Functions

1) Accuracy fitness: The fitness function $F_{accuracy}(i)$, which computes the accuracy of the DNN on an individual, depends on the DNN under test. In general, we suggest relying on the metric used to measure the reliability of the DNN. For example, in our empirical assessment, we test a DNN for image segmentation that is assessed, for safety reasons, based on its capability to detect cars accurately. Specifically, it is assessed with the *Intersection over Union (IoU)* metric, (i.e., Jaccard Index [18]). It measures the overlap between the predicted segmentation mask and the ground truth mask for a class (e.g., car, road), as the ratio of the intersection area to the union area of the two masks. Therefore, we compute $F_{accuracy}(i)$ by relying on the IoU formula for car objects:

$$IoU_{car}(i) = \frac{TP_{car}(i)}{TP_{car}(i) + FP_{car}(i) + FN_{car}(i)} \quad (1)$$

where $TP_{car}(i)$ are the true positive car pixels in image i (pixels that belong to cars in both ground truth and prediction), $FP_{car}(i)$ are the false positive pixels for cars (pixels erroneously reported as belonging to a car in the predicted segmentation mask), and $FN_{car}(i)$ are the false negative pixels (pixels erroneously not reported as belonging to a car in the predicted segmentation mask).

For the Martian case study, we consider all the classes when computing the accuracy. Therefore, we use the mean IoU ($mIoU$) as $F_{accuracy}(i)$. The $mIoU$ for one image is

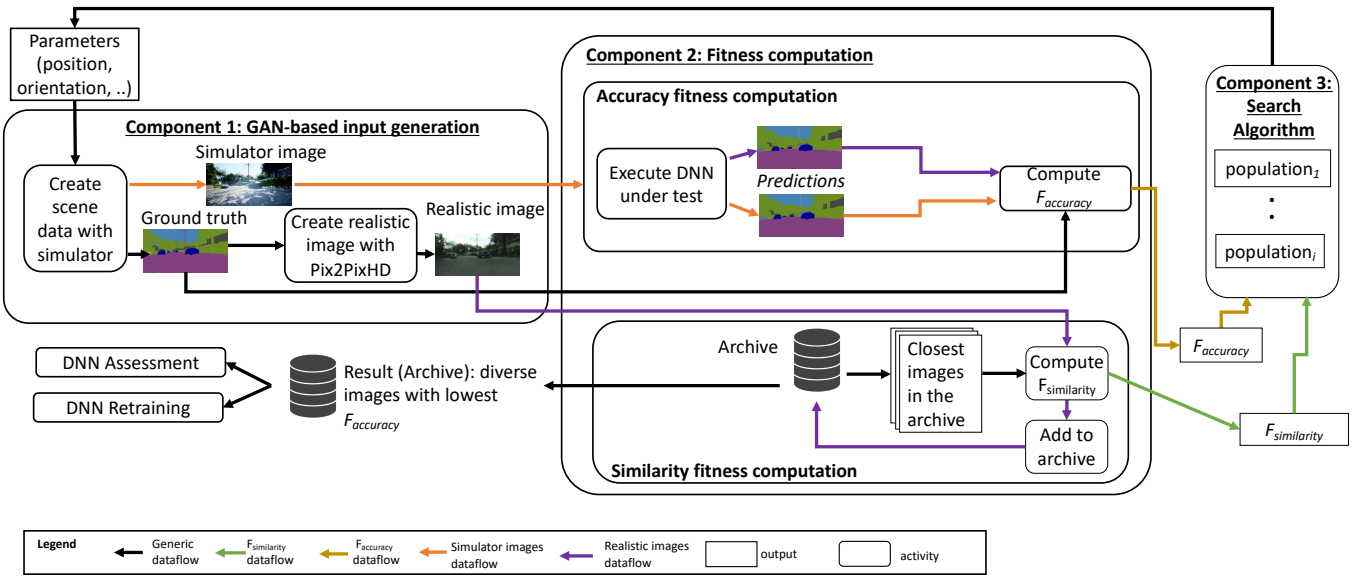

Fig. 4: Overview of DESIGNATE.

obtained by averaging the IoU scores across all classes and is computed as follows:

$$mIoU = \frac{1}{C}\sum_{c=1}^{C}\frac{TP_c}{TP_c + FP_c + FN_c} \tag{2}$$

Unfortunately, when test images are automatically generated using simulators and GANs, the DNN may badly perform because the generated images are not relevant for testing. Specifically, they may be out-of-domain (because of the simulator's control limitations) or may not match the ground truth (because of the GAN's limitations). Consequently, we defined a fitness function that distinguishes between non-relevant and relevant individuals, as follows:

$$F_{accuracy}(i) = \begin{cases} 2 & \text{if } i \text{ is not relevant} \\ performance(i) & \text{otherwise} \end{cases} \tag{3}$$

We assume the performance metric (i.e., $performance(i)$, based on $IoUcar(i)$) to be normalized in the range 0-1, with a perfect output leading to 1. Therefore, since in our search we aim to minimize the fitness function, we return a fitness of 2 for irrelevant individuals, so that they are discarded by the search process. Below, we describe how we determine irrelevant individuals.

How we determine *out-of-domain* inputs depends on the specific case study. When relying on AirSim, for example, we observe out-of-domain inputs when the ego vehicle is positioned next to another vehicle and perpendicular to it (e.g., facing the other vehicle's door), which is not interesting for our testing purposes since it is very easy to make a segmentation DNN fail in such case. Another case is that of images with no cars, which should be excluded because we are only interested in testing the DNN with images exhibiting traffic. Consequently, we consider an image to be relevant if the proportion of pixels belonging to cars (i.e., $carPixelsProportion$) in the input image is within a given range:

$$0 < carPixelsProportion < 0.4$$

The variable *carPixelsProportion* is computed on the segmentation ground truth. We set the upper bound to 0.4 to ensure that the car does not cover most of the image, which would result in unrealistic scenarios for evaluating DNN performance. Indeed, images where *carPixelsProportion* exceeds 0.4 often involve close-ups or unusual viewpoints, such as a vehicle occupying most of the camera's field of view—lidar-based sensors typically prevent autonomous cars from being in such situations. We set the lower bound above zero to ensure that the image contains at least one car. Similarly, for the Martian environment, we need to ensure that the sky does not cover most of the image and, therefore, we consider an image relevant if the proportion of pixels belonging to the sky (i.e., $skyPixelsProportion$) is within the range:

$$0 < skyPixelsProportion < 0.7$$

One may consider that 70% is a relatively large proportion of the image but it is a realistic situation based on our discussions with ESA representatives.

Also, to prevent the positioning of the car outside of the road (e.g., on a building), we ensure that the position of the car is within the road lanes, whose coordinates we mapped manually, by assigning a high fitness otherwise.

Detecting *GAN-generated realistic inputs not matching the ground truth* is hard to automate because only humans can determine if the image is a correct representation of the ground truth (e.g., cars really appear where car pixels are reported on the ground truth). However, to automate such verification, we propose relying on heuristics based on the difference in DNN reliability between images generated by the simulator and the corresponding GAN-generated realistic images. Our heuristic relies on the empirical observation that

the DNN under test performs better on a realistic image than on the corresponding simulator image, which is expected with DNNs trained on real-world images. To apply our heuristic, before testing, we consider a large set (1000) of simulator images generated with randomly selected parameter values, and generate corresponding realistic images. Each simulator and realistic image is processed by the DNN under test to generate a prediction, and the performance metric is computed for each image in each image pair, which enables computing the performance difference as:

$$delta_{performance}(i) = \\ performance_{simulated}(i) - performance_{realistic}(i) \tag{4}$$

where i indicates the i-th pair of simulator and corresponding realistic image.

After sorting the generated image pairs based on their delta$_{performance}(i)$, it is possible, through visual inspection, to determine a threshold ($T_{relevance}$) above which the realistic images do not correctly capture the ground truth (e.g., a car is missing where expected based on the ground truth). We selected $T_{relevance}$ as the third percentile of the $delta_{performance}$ distribution. During testing, we thus consider a realistic image to be relevant if $delta_{performance}(i) \leq T_{relevance}$. Regarding the segmentation DNN for urban environments considered in our empirical assessment:

$$delta_{performance}(i) = delta_{IoU}(i) = \\ IoU_{simulated}(i) - IoU_{realistic}(i) \tag{5}$$

In contrast, in the case of the martian environment, we did not observe GAN-generated images that are less realistic than those of the simulator and therefore we do not rely on $T_{relevance}$. We believe this is mainly due to our Mars simulator having limited fidelity and is thus representative of cases where limited effort was dedicated to simulator development[1].

In the segmentation DNN considered in our empirical assessment with the urban case study, $F_{accuracy}$ is therefore computed as follows:

$$F_{accuracy}(i) = \begin{cases} 2 & \text{if } 0 < carPixelsProportion < 0.4 \\ 2 & \text{if the ego vehicle is off the road} \\ 2 & \text{if } delta_{IoU}(i) > T_{relevance} \\ IoU_{car}(i) & \text{otherwise} \end{cases} \tag{6}$$

For the martian case study, based on the above discussion, $F_{accuracy}$ is instead computed as follows:

$$F_{accuracy}(i) = \begin{cases} 2 & \text{if } 0 < skyPixelsProportion < 0.7 \\ mIoU(i) & \text{otherwise} \end{cases} \tag{7}$$

2) Diversity fitness: Our diversity fitness should enable determining if a newly generated image captures a situation that was not observed before (e.g., no traffic on the road VS a lot of traffic on the road), which can be achieved by computing the difference between the newly generated image and the previously generated images (e.g., stored in an archive). However, the above-mentioned objective cannot simply rely on a simple computation of the the pixel-by pixel difference between two images; indeed, two images generated before and after slightly turning a car, may present pixels that are all shifted in one direction thus resulting into a high difference even if the two images capture the same situation.

To address the problem above, we suggest capturing the semantic difference between two images by computing the difference between the feature vectors derived from the two images. Based on our previous experience with clustering DNN mispredictions [5], we rely on a ResNet50 model trained on the ImageNet dataset to automatically derive feature vectors. The deep architecture of ResNet50 (50 layers) allows it to capture richer and more abstract features at different levels of the network. This makes it more powerful for feature extraction tasks where high-level semantic features are important [30], [48]. Image diversity can then be computed as the Euclidean distance between two feature vectors. Euclidean distance inherently measures the geometric distance between points in a multi-dimensional space; when applied to feature vectors representing images, it captures the differences in features between images. Images with similar features will have a shorter distance, indicating less diversity, while images with dissimilar features will have a longer distance, suggesting higher diversity.

In the presence of an archive with previously generated images, $F_{similarity}$, which captures how i is similar to previously generated images, can thus be computed as a function of the Euclidean distance from the closest image in the archive. However, in a search algorithm, preserving images that slightly differ from previously generated ones may lead to being stuck in local optima; therefore, inspired by previous work [44], we introduce a threshold for the diversity ($T_{diversity}$) below which an image is not considered different from a previously generated one and is thus assigned with high fitness:

$$F_{similarity}(i) = \begin{cases} 2 & \text{if } distanceFromClosest(i) < T_{diversity} \\ \frac{1}{1+distanceFromClosest(i))} & \text{otherwise} \end{cases} \tag{8}$$

The value of $distanceFromClosest(i)$ is identified by computing the Euclidean distance of the image i from each image in the archive and selecting the shortest one. Like in related work [44], we compute $T_{diversity}$ as the median of the pairwise distances in a set of 1000 randomly generated images. Finally, the term $\frac{1}{1+distanceFromClosest(i)}$ is used for normalization: it leads to a fitness close to zero for a very high distance, and to one for overlapping images.

C. Search algorithm

We aim to address a multi-objective problem [4], generating realistic images leading to minimal DNN accuracy and having minimal similarity among each other. We therefore rely on the NSGA-II algorithm, which demonstrated its effectiveness in multiple multi-objective software testing problems. However, inspired by related work aiming at maximizing the diversity

[1]Please note that state-of-the-art realistic simulators for space exist (e.g., PANGU [49]), but they do not generate the ground truth labels required to train segmentation DNNs.

Algorithm 1 Updating the archive with image i

Require: A: archive with individuals selected so far, i: individual to be added, $T_{similarity}$
Ensure: archive with individuals that are sparse and minimize $F_{accuracy}$
1: **if** $F_{accuracy}(i) == 2$ **then** ▷ ignore non-relevant
2: **return**
3: **if** $distanceFromClosest(i) > T_{diversity}$ **then**
4: add i to A
5: **else if** $\big(distanceFromClosest(i) \leq T_{diversity}$ **and** $F_{accuracy}(closestTo(i)) > F_{accuracy}(i)\big)$ **then**
6: put i in A, in place of i's closest individual already in A

among the generated individuals [44], we extend NSGA-II with an archive to store the best (non-dominated) individuals encountered during the search process. The archive plays a critical role in preventing the genetic algorithm from cycling, a phenomenon where the population moves from one area of the solution space to another and then back again without exploring other areas. By retaining the best individuals in the archive, and assigning high fitness values to individuals that are similar to the already generated ones, the algorithm ensures that these areas are not revisited, facilitating a more efficient exploration of the solution space.

Before delving into our search algorithm, we describe the procedure adopted to update the archive with an individual i, which is shown in Algorithm 1. Individuals that are not relevant for testing (i.e., $F_{accuracy}(i) == 2$) are not added to the archive (Line 1). Instead, we add individuals that are relevant and whose distance from the closest individual in the archive is above the predefined threshold (Line 4). Last, since individuals that are similar to others in the archive (i.e., $distanceFromClosest(i) < T_{diversity}$) may still be more effective for testing (i.e., lower $F_{accuracy}$), we put the individual i in place of the individual that is closest to i in the archive when the $F_{accuracy}$ of the former is lower than the accuracy of the latter (Line 6).

Algorithm 2 presents the pseudocode of the NSGA-II algorithm (with archive) employed in DESIGNATE. After creating an empty archive (Line 1), since the initial selection of seeds may affect the search results and inspired by our previous work [3], we generate multiple (5 in our experiments) populations of N randomly selected individuals (Line 2) and we select the population P_0 as the one with the individual having the lowest $F_{accuracy}$ (Line 4). This choice is based on empirical observations from prior studies [20], which demonstrated that using multiple seeds helps to reduce bias in the optimization process and ensures a broader exploration of the solution space. For instance, using five seeds strikes a balance between computational efficiency and the reliability of search results in related search-based testing tasks. All the individuals in P_0 are then used to update the archive based on Algorithm 1 (Line 5). In practice, the archive is populated with individuals whose distance is above $T_{diversity}$ and lead to worst accuracy.

As per NSGA-II, the population undergoes a non-dominated sorting process and crowding distance calculation (Lines 6-7).

The algorithm's core is a loop that iterates until the generation count reaches the specified maximum g_{max}. Within each generation g, we use NSGA-II's tournament selection method

Algorithm 2 Search algorithm used in DESIGNATE

Require: g_{max}: maximum number of generations, N: population size, P_c: crossover probability, P_m: mutation probability, r: number of populations for initial seed selection
Ensure: An archive of diverse individuals leading to minimal DNN accuracy
1: Initialize an empty archive A
2: Initialize r random populations $P_0^0..P_0^r$ with N individuals in each population
3: Evaluate $F_{accuracy}$ and $F_{similarity}$ of each individual in each population $P_0^0..P_0^r$ based on Equations 6 and 8
4: Select P_0 as the population with the individual having the lowest $F_{accuracy}$.
5: Update the archive A based on $F_{accuracy}$ and $F_{similarity}$ of each individual in P_0
6: Rank the individuals in using non-dominated sorting
7: Calculate crowding distance for each individual
8: $g \leftarrow 0$
9: **while** $g < g_{max}$ **do**
10: Select parents from the population P_g using binary tournament selection based on rank and crowding distance
11: Generate offspring Q_g using polynomial mutation and simulated binary crossover
12: Combine P_g and Q_g into R_g
13: Compute $F_{accuracy}$ for each individual in Q_g
14: Compute $F_{similarity}$ for each i in R_g
15: **for** i in R_g **do**
16: Update the archive A with i using on Algorithm 1
17: Update $F_{similarity}$ of each individual in R_g
18: Rank R_g using non-dominated sorting
19: Calculate crowding distance for each individual in R_g
20: Select the top N individuals from R_g to form new population P_{g+1} based on rank and crowding distance
21: $g \leftarrow g + 1$
22: **return** A

to select parent individuals for the next generation (Line 10). Offspring generation follows, employing polynomial mutation and simulated binary crossover (Line 11). Then, we compute $F_{accuracy}$ for each individual in Q_g (Line 13). We also compute $F_{similarity}$ for each individual in R_g, with R_g being the union of the current population P_g with the offspring population Q_g (Line 14). Recomputing $F_{similarity}$ for P_g is necessary because, in the previous iteration, new individuals have been added to the archive and, therefore, distances have changed. We then try to add each image to the archive based on Algorithm 1 (Line 16). Finally, since the archive changed, we update $F_{similarity}$ for the individuals in R_g (Line 17). R_g is then ranked using the non-dominated sorting process (Line 18) and the top individuals, based on rank and crowding distance, form the new population P_{t+1} for the subsequent generation (Lines 19-20).

The algorithm concludes by returning the populated archive, which includes diverse individuals leading to minimal DNN accuracy.

IV. Empirical Evaluation

We report on an empirical assessment of DESIGNATE aiming to address the following research questions:

RQ1: How does DESIGNATE compare to alternative DNN testing solutions in terms of test effectiveness? Alternatives include approaches generating test images by relying on GANs (but not simulators), random baselines, and alternative implementations of our approach (ablation study). We discuss what approach leads to inputs leading to the lowest DNN accuracy and having the highest diversity.

RQ2: How does DESIGNATE compare to alternative DNN testing solutions to improve the DNN? Images generated for DNN testing can also be used to retrain a DNN; we therefore aim to determine what approach generates images that, when used for DNN retraining, increase DNN accuracy the most.

A. Subjects of the Study

As subjects of our study, we consider two distinct subjects: 1) The DeeplabV3 DNN [11] from the Gluoncv library [24], which is a state-of-the-art semantic segmentation DNN that classifies each pixel image into a predefined class. This DNN focuses on urban environments relying on the Cityscapes dataset [46]. 2) A DeeplabV3 DNN trained for object detection on Mars, utilizing the AI4MARS dataset [51]. This DNN implements a key task for the navigation of Mars rovers; specifically, it detects objects such as rocks, sand, and bedrocks. This subject was developed in the context of a project with the European Space Agency.

For the urban environment case, DeeplabV3 was initially trained by its developers on the Cityscapes dataset, which contains a variety of urban street scenes from different European cities. The dataset consists of a variety of images captured at a resolution of 1024 by 2048 pixels in the streets of 50 different European cities, primarily in Germany. Each real-world image comes with a segmentation mask representing its ground truth. The DeeplabV3 training set consists of 2975 images; however, its test set comes without ground truth and, therefore, for testing, we rely on the Cityscapes validation set (500 images). The average IoU_{car} metric (see Section III) is particularly high: 71% (min=00%, first-quartile=62%, third-quartile=91%, max=97%).

For the Mars navigation subject, we trained the DNN on the AI4MARS dataset, which provides annotated images of the Martian surface. The simulator for generating synthetic Martian images was built using Unreal Engine [17] with the same plugins used by AirSim. The simulator was configured to simulate realistic Martian terrain and lighting conditions. The Airsim plugins are added to: simulate Vehicle dynamics (including interaction with sand and soil), add LIDAR for object detection, provide Python API for real-time control of vehicle, enabling communication between our search algorithm and the simulated vehicle. For segmentation on the AI4MARS dataset, DeepLabV3 achieved an average mIoU of 49% (min=0%, first-quartile=1%, third-quartile=77%, max=99%).

B. Assessed techniques

In our experiments, we assess a DESIGNATE prototype, that relies on the original pix2pixHD GAN [39] and AirSim 1.8.1 [38] for the urban case study, and Unreal Engine 5 for the Martian case study. We also extended the NSGA-II algorithm provided by the Pymoo library [8].

To address our RQs, we compare DESIGNATE with a random baseline, two variants of DESIGNATE (for the purpose of performing ablation studies), TACTIC [34], and DeepJanus [44]. We selected DeepJanus because it may lead to better results if a DNN tends to fail where model behaviour changes. For DeepJanus, in addition to the original implementation provided by its authors, we developed an extended version that integrates a GAN, as described below.

TABLE I: Configuration of DESIGNATE

Parameter	Value
Population size	12
Generations	100
Mutation probability	0.3
Crossover probability	0.7
$T_{similarity}$ (for DESIGNATE)	18
$T_{similarity}$ (for DESIGNATE$_{pix}$)	0.02
$T_{relevance}$	0.1

Our random baseline relies only on the GAN-based input generation component; specifically, it generates realistic images from simulator segmentation maps generated using randomly selected values for the simulator parameters (e.g., position of the car in Airsim). For brevity, we refer to the images from our random baseline as *random realistic images*. We generate as many random images as DESIGNATE (i.e., $12 * 100 = 1200$).

We defined three additional DESIGNATE variants (hereafter, DESIGNATE$_{single}$, DESIGNATE$_{pix}$, and DESIGNATE$_{NoGAN}$) to assess how $F_{similarity}$ and the feature-based distance contribute to DESIGNATE results. Specifically, DESIGNATE$_{single}$ employs a single objective search where our algorithm is configured with $F_{accuracy}$ as the sole fitness function and all the images are added to the archive. DESIGNATE$_{pix}$ relies on the same algorithms as DESIGNATE but, instead of relying on feature-based distance, it computes the distance between two realistic images as the percentage of pixels that do not match. Such distance metric matches the pixel accuracy metric, which is commonly used to assess semantic segmentation DNNs. DESIGNATE$_{NoGAN}$ uses the same process as DESIGNATE, but without the GAN component, therefore, both fitnesses are computed using the simulator images.

Table I provides the hyper-parameters used for DESIGNATE, DESIGNATE$_{single}$, DESIGNATE$_{pix}$, and DESIGNATE$_{NoGAN}$. For population size and number of generations, we selected the same values adopted by related search-based approaches with similar objectives (i.e., minimize accuracy and maximize diversity) [44]. We selected the mutation and crossover probabilities that proved to be effective in our previous work on simulator-based image generation [20]. Thresholds were determined based on 1000 randomly generated images, as described in previous sections. The two different $T_{similarity}$ thresholds refer to the two different distance metrics used in DESIGNATE and DESIGNATE$_{pix}$.

TACTIC is a state-of-the-art DNN testing approach that leverages a MUNIT GAN trained to alter input images by changing their weather conditions (night, sunshine, rain, snow); further, it integrates a search-based approach which alters the latent space vector in such a way that the generated images lead to the worst accuracy for the DNN under test. To apply TACTIC, we randomly select 220 images from the Cityscapes validation set and apply the five weather conditions supported by TACTIC (night, rain, snow night, snow daytime,

and sunny conditions), each leading to a distinct new test image. This process generates a total of 1100 images. Since TACTIC was originally trained to generate driving conditions, we consider it for the Urban environment only; indeed, it makes little sense to add snow and rain effects on Mars images, as well as, changing illumination to make it similar to Earth.

DeepJanus is multi-objective search-based approach that searches for the boundary of the input space that is properly processed by the DNN; specifically, it looks for pair of images that are similar but leads to different DNN results, one of which is erroneous. We integrate DeepJanus with our simulators with the same configurations as in the original paper. Furher, since DeepJanus exclusively produces simulated images, we explore the impact of using a GAN to generate more realistic images (hereafter, DeepJanus$_{GAN}$). To achieve this, we incorporated the Pix2pixHD model, as utilized in DESIGNATE, into the DeepJanus workflow to transform simulated images into realistic ones; precisely, DeepJanus$_{GAN}$, during search, still generates simulator images and ground truth as in DeepJanus, but, instead of providing a simulator image to the DNN under test, it first generates a realistic image using Pix2pixHD and then feeds it to the DNN under test.

C. RQ1: Effectiveness

1) Design and measurements: For each assessed technique, we compute the IoU_{car}, for the urban case study, and $mIoU$ for the Martian case study (see Section III-B1 for definitions), obtained by the DeepLabV3 model with the generated images. Since we aim to discover critical DNN limitations, the most effective technique is the one leading to the lowest median, minimum, and first quartile.

However, since we also wish to obtain a set of diverse unsafe situations, we also assess diversity across the generated images by relying on the two distance metrics employed by DESIGNATE: feature-based distance and pixel-based distance. Specifically, we compute the distance between every possible pair of images generated by the selected techniques and compare their distributions. An approach with higher diversity values is preferable to identify the different situations under which segmentations are erroneous.

To cope with the non-determinism characterizing all the considered techniques, we apply each of the selected techniques ten times.

We discuss the significance of the differences by relying on a non-parametric Mann-Whitney U-test (the U statistics is computed considering all the datapoints generated by the ten executions). Further, we address effect size by relying Vargha and Delaney's $\hat{A}_{12}$. The A_{12} statistic, given observations (e.g., IoU) obtained with two techniques X and Y, indicates the probability that technique X leads to higher values than technique Y. Based on A_{12}, effect size is considered small when $0.56 \leq A_{12} < 0.64$, medium when $0.64 \leq A_{12} < 0.71$, and large when $A_{12} \geq 0.71$. Otherwise the two populations are considered equivalent [54]. In contrast, when A_{12} is below 0.50, it is more likely that treatment X leads to lower values than treatment Y; effect size is small when $0.36 < A_{12} \leq 0.44$, medium when $0.29 < A_{12} \leq 0.36$, and large when $A_{12} \leq 0.29$.

2) Results: ***Accuracy.*** Table II and IV provide, for each of the selected approaches, details on the performance of the subject DNNs for the urban and the Martian environment, respectively. Column *images* provides the average number of test images selected over the ten executions; for DESIGNATE and its variants we report the images in the archive, for Random and TACTIC we report all the generated images, excluding, for the former, an average of 15 and 169 images, for the urban and Martin environments, respectively, that do not satisfy '$0 < pixelsProportion < 0.4$'. The other columns provide descriptive statistics for IoU_{car} and $mIoU$. Table VII and Table V report differences between each pair of techniques by providing p-values and A_{12} for the urban and martian environments, respectively. We discuss the results observed with the urban environment first.

DESIGNATE$_{single}$ is the approach leading to the largest number of images with low IoU_{car} values; indeed, it shows the lowest average (0.48) and median (0.67) values. However, although their difference is always statistically significant (p-value < 0.05, Table VII), based on A_{12}, DESIGNATE$_{single}$, DESIGNATE, DESIGNATE$_{pix}$, and DeepJanus$_{GAN}$ perform similarly and do not lead to practically significant differences.

Except for DESIGNATE$_{NoGAN}$, all the DESIGNATE variants perform significantly better (lower IoU_{car}) than Random, and TACTIC (Table VII), which indicates that combining simulators driven by search with GAN is necessary for effective testing. For example, DESIGNATE leads to a median IoU_{car} of 0.77 compared to 0.85 and 0.82 for Random and TACTIC, respectively. Further, the first quartile of all the GAN-based DESIGNATE variants is zero, indicating that 25% of the generated images do not correctly identify a single pixel belonging to cars. For Random, less than 25% of the images lead to $IoU_{car} = 0$ (first quartile is 0.58), while TACTIC tends to lead to a higher IoU_{car} (average $IoU_{car} = 0.81$), with IoU_{car} never being equal to 0. Also, all the DESIGNATE variants fare better than DeepJanus$_{GAN}$, thus showing that an appropriate search strategy is necessary to identify test inputs leading to failures. As for DeepJanus, its IoU is similar in terms of median to that of DESIGNATE$_{single}$ and worse in terms of average.

Also, Table III shows that the difference between DESIGNATE and both TACTIC and Random is practically significant, according to A_{12}. DESIGNATE$_{NoGAN}$, instead, leads to median results (0.84) that are in-between those of TACTIC and Random, with A_{12} indicating that DESIGNATE$_{NoGAN}$ and TACTIC perform similarly, while Random performs slightly better than DESIGNATE$_{NoGAN}$ ($A_{12} = 0.43$). These results suggest that relying on simulator images only is not sufficient to drive the search towards cases where the DNN performs worse.

Concerning TACTIC, we further checked how the different weather conditions (rain, snow day, sunny, and night) perform. Figure 5 illustrates an example of a generated image for each weather condition, alongside the median IoU_{car} of the generated images for the respective conditions. Our observations indicate that three conditions — Sunny (median IoU_{car} = 80%, min = 50%, max = 97%, average = 78%), Night (median IoU_{car} = 90%, min = 50%, max = 94%, average = 80%), and

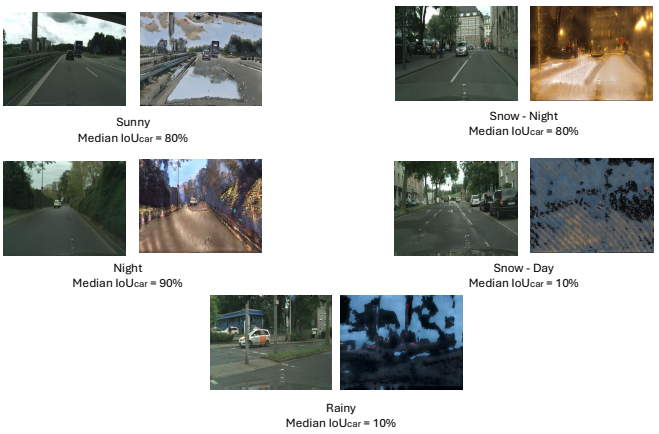

Fig. 5: Examples of images generated by TACTIC for various weather conditions (right), along with the original cityscapes image (left) and the median IoU_{car} of the generated images for each respective weather condition.

Snow-Night (median IoU_{car} = 80%, min = 50%, max = 91%, average = 80%) — yield high median IoU_{car} values, with the generated images appearing realistic. Conversely, the Rainy (median IoU_{car} = 10%, min = 1%, max = 32%, average = 10%) and Snow-Day (median IoU_{car} = 10%, min = 2%, max = 36%, average = 10%) conditions result in lower median IoU_{car} values, with the generated images being unrealistic, with no observable cars, and therefore not matching the ground truth. Such images, as illustrated in Figure 5, are not useful for testing purposes.

The slightly better performance of DESIGNATE$_{single}$ can be explained by the fact that DESIGNATE$_{single}$, being single-objective, may be stuck in a particular region of the solution space, leading to outputs that are less diverse but all leading to minimal IoU_{car}. In contrast, DESIGNATE adopts a multi-objective strategy, aiming to balance both low IoU_{car} and diversity, which may lead to a higher IoU_{car} (e.g., there might be regions of the solution space where the DNN performs poorly but not as bad as in the only region detected by DESIGNATE$_{single}$).

The random baseline appears to be less effective, with a high median IoU_{car} of 85%, which shows that the DNN under test performs generally well in the sampled input space and a search-based approach is needed to comprehensively uncover its limitations.

DeepJanus yields slightly better accuracy compared to DESIGNATE and DESIGNATE$_{pix}$ with A_{12} suggesting that DeepJanus, DESIGNATE, and DESIGNATE$_{pix}$ perform similarly. However, DESIGNATE$_{single}$ outperforms DeepJanus, which suggests that the single-objective search used by DESIGNATE$_{single}$ leads to better results than the one used by DeepJanus. Further, after visual inspection, we realized that most the failures triggered by DeepJanus are due to borderline cases with cars appearing far away and occupying few pixels in the image. Though such sceneries can be generated by simulators, they are likely to be irrelevant in practice as the position of cars that are far away are unlikely to affect the behaviour of the ego vehicle and thus be relevant for testing. More precisely, because DeepJanus looks for frontier behaviour, with pairs of images that are very similar but where one of the two leads to failures, it tends to generate images where cars are far from the ego vehicle.

The higher accuracy of DeepJanus$_{GAN}$, which performs worse and significantly different from DESIGNATE, is likely due to the fact that, when applying GANs, far away cars disappear from the picture. Indeed, the training set does not contain cars occupying only few pixels, because such far-away cars are irrelevant for a segmentation DNN in an automotive system and thus not important for testing purposes.

For the Martian environment, DESIGNATE achieved a median $mIoU$ of 0.49, outperforming DESIGNATE$_{single}$ (0.54), DESIGNATE$_{pix}$ (0.52), Random (0.64), and DeepJanus$_{GAN}$ (0.60). Similar to the urban environment, the A_{12} values in Table V indicate that DESIGNATE$_{single}$, DESIGNATE, and DESIGNATE$_{pix}$ perform similarly, although their differences are statistically significant ($p < 0.05$).

The key difference between the urban and Martian environments is that the lowest accuracy values are observed with DESIGNATE$_{NoGAN}$ and DeepJanus: 0.32 and 0.48 in the latter, as opposed to DESIGNATE$_{single}$ in the former. We attribute this result to DESIGNATE$_{NoGAN}$ and DeepJanus relying on simulated images that are often unrealistic, thereby increasing the likelihood of mispredictions. This can be explained by the low fidelity of the Mars simulator when compared to AirSim, as depicted in Fig. 3. To conclude, as reported for the urban environment, the use of GAN prevents the identification of failures that will not occur in real usage settings. In other words, when not using a GAN and relying on a low-fidelity simulator, many unrealistic and therefore irrelevant test inputs are generated.

Diversity. Table VI and Table VIII provide descriptive statistics for the diversity metrics collected for the selected techniques for the Urban and Martian environments, respectively. Additionally, Table VII and Table IX provide p-values and $\hat{A}_{12}$ resulting from the comparisons of diversity across techniques for the Urban and Martian environments, respectively. We discuss the urban environment first.

Considering feature-based distance, DESIGNATE stands out with the highest median (14.97) and average (15.25) values. This result is likely due to DESIGNATE integrating feature-based distance in $F_{similarity}$. Further, DESIGNATE performs significantly better than other DESIGNATE variants ($p-value < 0.01$) and has a significantly higher probability of performing better than other DESIGNATE variants (medium effect size, based on $\hat{A}_{12}$). Expectedly, the lack of a diversity objective in DESIGNATE$_{single}$ leads to images that are less diverse than those of DESIGNATE. The lower diversity of DESIGNATE$_{pix}$ compared to DESIGNATE confirms that a fitness based on pixel distance is not effective to obtain feature-based diversity. Last, the lower median obtained by DESIGNATE$_{NoGAN}$ (12.90) is likely due to the lower fidelity of simulator images compared to the ones generated by the GAN (e.g., simulator images lack details, leading to images that are more similar to each other).

DESIGNATE performs significantly better, with high effect

TABLE II: RQ1-Accuracy assessment. Descriptive statistics for IoU_{car} obtained with the generated images for the Urban environment.

	#images	$\mathbf{IoU_{car}}$ distribution						
		min	max	median	5th percentile	1st quartile	3rd quartile	average
DESIGNATE$_{single}$	893	**0.00**	0.99	**0.67**	**0.00**	**0.00**	0.89	**0.48**
DESIGNATE	721	**0.00**	0.99	0.77	**0.00**	**0.00**	0.91	0.54
DESIGNATE$_{pix}$	783	**0.00**	0.99	0.76	**0.00**	**0.00**	0.92	0.52
DESIGNATE$_{NoGAN}$	1190	**0.00**	1.00	0.84	0.21	0.68	1.00	0.78
Random	1185	**0.00**	1.00	0.85	**0.00**	0.58	0.95	0.69
TACTIC	1100	0.02	**0.97**	0.82	0.69	0.75	0.88	0.81
DeepJanus	943	**0.00**	0.99	0.74	0.26	0.59	**0.82**	0.68
DeepJanus$_{GAN}$	954	0.01	0.99	0.87	0.40	0.76	0.94	0.81

Note: best (i.e., lowest) results (per column) in bold.

TABLE III: RQ1-Accuracy assessment. p-values and A_{12} for the data in Table II.

	DESIGNATE$_{single}$		DESIGNATE		DESIGNATE$_{pix}$		DESIGNATE$_{NoGAN}$		Random		TACTIC		DeepJanus		DeepJanus$_{GAN}$	
	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value
DESIGNATE$_{single}$			0.45	5.00E-28	0.47	8.00E-13	**0.28**	0.00E+00	**0.34**	3.00E-298	**0.35**	1.00E-64	0.47	2.19E-09	**0.33**	7.53E-301
DESIGNATE	0.55	5.00E-28			0.52	1.00E-03	**0.33**	0.00E+00	**0.39**	1.14E-139	**0.41**	9.00E-21	0.53	9.24E-09	**0.38**	1.36E-131
DESIGNATE$_{pix}$	0.53	8.00E-13	0.48	2.00E-03			**0.31**	0.00E+00	**0.38**	8.00E-161	**0.41**	1.00E-22	0.52	4.69E-03	**0.38**	2.41E-150
DESIGNATE$_{NoGAN}$	**0.72**	0.00E+00	**0.67**	0.00E+00	**0.69**	0.00E+00			**0.57**	0.00E+00	0.55	0.00E+00	**0.73**	1.88E-133	**0.61**	1.19E-33
Random	**0.66**	3.00E-298	**0.61**	1.14E-135	**0.62**	8.00E-161	**0.43**	0.00E+00			**0.54**	5.10E-05	**0.66**	2.23E-210	0.50	6.95E-01
TACTIC	**0.64**	1.00E-64	**0.58**	9.00E-21	**0.59**	1.00E-22	0.45	0.00E+00	0.46	5.10E-05			**0.73**	3.40E-55	0.45	2.15E-04
DeepJanus	0.53	2.19E-09	0.47	9.24E-09	0.48	4.69E-03	**0.27**	1.88E-133	**0.34**	2.23E-210	**0.27**	3.40E-55			**0.29**	2.18E-297
DeepJanus$_{GAN}$	**0.67**	7.53E-301	**0.62**	1.36E-131	**0.62**	2.41E-150	**0.39**	1.19E-33	0.50	6.95E-01	0.55	2.15E-04	**0.71**	2.18E-297		

Note: tangible (based on A_{12}) differences in bold; for each pair of approaches, the best approach is the one with $A_{12} < 0.50$ and its name on the row.

TABLE IV: RQ1-Accuracy assessment. Descriptive statistics for $mIoU$ obtained with the generated images for the Martian environment.

	#images	mIoU_distribution						
		min	max	median	5th percentile	1st quartile	3rd quartile	average
DESIGNATE$_{single}$	802	**0.01**	0.96	0.54	0.09	0.35	0.73	0.52
DESIGNATE	764	**0.01**	0.96	0.49	0.09	0.33	0.69	0.50
DESIGNATE$_{pix}$	718	**0.01**	0.95	0.52	0.20	0.39	0.70	0.53
DESIGNATE$_{NoGAN}$	1151	**0.01**	0.97	**0.32**	**0.03**	**0.12**	0.70	**0.39**
Random	1031	**0.01**	0.96	0.64	0.24	0.44	0.83	0.62
DeepJanus	986	**0.01**	**0.93**	0.48	**0.03**	0.20	**0.67**	0.44
DeepJanus$_{GAN}$	954	**0.01**	0.98	0.60	0.20	0.42	0.85	0.61

Note: best (i.e., lowest) results (per column) in bold.

TABLE V: RQ1-Accuracy assessment for the Martian case study. p-values and A_{12} for the data in Table IV.

	DESIGNATE$_{single}$		DESIGNATE		DESIGNATE$_{pix}$		DESIGNATE$_{NoGAN}$		Random		DeepJanus		DeepJanus$_{GAN}$	
	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value
DESIGNATE$_{single}$			0.53	1.10E-11	0.50	4.41E-01	**0.63**	4.55E-197	**0.38**	3.11E-153	**0.59**	1.80E-81	**0.40**	1.45E-116
DESIGNATE	0.47	1.10E-11			**0.46**	5.22E-16	**0.61**	2.20E-136	**0.36**	4.16E-228	**0.56**	2.89E-34	**0.37**	5.03E-178
DESIGNATE$_{pix}$	0.50	0.440865115	0.54	5.22E-16			**0.65**	2.54E-239	**0.38**	6.00E-146	**0.59**	2.76E-85	**0.40**	1.90E-103
DESIGNATE$_{NoGAN}$	**0.37**	4.55E-197	**0.39**	2.20E-136	**0.35**	2.54E-239			**0.28**	0.00E+00	0.46	3.18E-24	**0.28**	0.00E+00
Random	**0.62**	3.11E-153	**0.64**	4.16E-228	**0.62**	6.00E-146	**0.72**	0.00E+00			**0.69**	0.00E+00	0.50	3.97E-01
DeepJanus	**0.41**	1.80E-81	**0.44**	2.89E-34	**0.41**	2.76E-85	0.54	3.18E-24	**0.31**	0.00E+00			**0.33**	0.00E+00
DeepJanus$_{GAN}$	**0.60**	1.45E-116	**0.63**	5.03E-178	**0.60**	1.90E-103	**0.72**	0.00E+00	0.50	3.97E-01	**0.67**	0.00E+00		

Note: tangible (based on A_{12}) differences in bold; for each pair of approaches, the best approach is the one with $A_{12} < 0.50$ and its name on the row.

size, than TACTIC and Random. TACTIC has a feature-based distance that is similar across pairs of images, between 8.09 and 10.49 (avg. is 10.06). Since in our experiments TACTIC works by modifying a set of 220 randomly selected images, and since these images present a feature-based distance of at least 8.09, it seems that the degree of diversity introduced by TACTIC through changes in weather conditions is limited. DESIGNATE, instead is more effective in creating diverse images as 75% (1st quartile) of the pairs of images generated by DESIGNATE have a distance higher than 12.29, which is higher than the best results achieved by TACTIC. Further 25% of DESIGNATE images yield a distance higher than 18.08.

The random baseline leads to images with low feature-based distance, with a median of 10.99 and a maximum of 12.34; it shows that the mere sampling of the input space does not lead to identifying diverse images and that search guidance is needed.

DESIGNATE significantly outperforms DeepJanus and DeepJanus$_{GAN}$. Their lower diversity suggests that their search algorithm, which focuses on identifying the boundaries of correctly processed inputs, is unexpectedly less effective than NSGA-II in exploring the search landscape. DeepJanus$_{GAN}$ achieves higher diversity values (29.71) than DeepJanus thus confirming that GANs, by avoiding unrealistic cases (e.g., cars with few visible pixels, as discussed above), prevents the search from getting stuck in unrealistic, similar images.

Considering pixel-based distance, all the DESIGNATE variants have the same average distance but DESIGNATE still performs better than the others, with a small to high effect size. Such results confirm our observations above for DESIGNATE$_{single}$ and DESIGNATE$_{NoGAN}$. Further, the comparison with DESIGNATE$_{pix}$ indicates that a fitness driven by pixel-based distance is less effective than a feature-based distance fitness to maximize pixel-based diversity across images. This indicates that, for the search algorithm, it is easier to generate pixel-wise diverse images when driven by a feature-based fitness. Note that augmenting feature-based

TABLE VI: RQ1-Diversity assessment. Descriptive statistics for diversity across the generated images for the Urban environment.

	Feature-based distance						
	min	**max**	**median**	**5th percentile**	**1st quartile**	**3rd quartile**	**Average**
DESIGNATE$_{single}$	1.15	27.97	12.83	9.00	11.16	14.71	13.04
DESIGNATE	0.48	28.80	**14.97**	9.37	**12.29**	**18.08**	**15.25**
DESIGNATE$_{pix}$	0.43	27.58	12.67	8.33	10.90	14.87	13.05
DESIGNATE$_{NoGAN}$	0.38	15.01	12.90	5.22	11.30	13.98	12.08
Random	0.01	12.34	10.99	8.56	10.06	11.70	10.77
TACTIC	**8.09**	10.49	10.11	9.20	9.80	10.32	10.06
DeepJanus	2.26	29.03	13.21	**10.01**	11.85	14.78	13.45
DeepJanus$_{GAN}$	2.66	**29.71**	12.11	8.71	10.60	13.89	12.40
	Pixel-based distance						
	min	**max**	**median**	**5th percentile**	**1st quartile**	**3rd quartile**	**Average**
DESIGNATE$_{single}$	0.30	0.99	0.97	**0.94**	0.96	0.98	**0.97**
DESIGNATE	0.14	**1.00**	**0.98**	**0.94**	**0.97**	**0.99**	**0.97**
DESIGNATE$_{pix}$	0.10	**1.00**	0.97	0.93	0.96	0.98	**0.97**
DESIGNATE$_{NoGAN}$	0.26	0.99	0.92	0.83	0.89	0.94	0.91
Random	0.30	0.76	0.73	0.62	0.70	0.75	0.72
TACTIC	0.43	0.83	0.64	0.51	0.59	0.68	0.63
DeepJanus	**0.66**	**1.00**	0.94	0.89	0.92	0.96	0.94
DeepJanus$_{GAN}$	0.65	**1.00**	0.97	0.93	0.95	0.98	0.96

Note: best (i.e., highest) results (per column) in bold.

TABLE VII: RQ1-Diversity assessment. p-values and A_{12} for the data in Table VI.

Feature-based distance																
	DESIGNATE$_{single}$		DESIGNATE		DESIGNATE$_{pix}$		DESIGNATE$_{NoGAN}$		Random		TACTIC		DeepJanus		DeepJanus$_{GAN}$	
	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value
DESIGNATE$_{single}$			**0.33**	0.00E+00	0.51	0.00E+00	0.55	0.00E+00	**0.78**	0.00E+00	**0.87**	0.00E+00	0.45	0.00E+00	**0.57**	0.00E+00
DESIGNATE	**0.67**	0.00E+00			**0.67**	0.00E+00	**0.72**	0.00E+00	**0.86**	0.00E+00	**0.92**	0.00E+00	**0.64**	0.00E+00	**0.72**	0.00E+00
DESIGNATE$_{pix}$	0.49	0.00E+00	**0.33**	0.00E+00			0.53	0.00E+00	**0.75**	0.00E+00	**0.84**	0.00E+00	**0.44**	0.00E+00	**0.56**	0.00E+00
DESIGNATE$_{NoGAN}$	0.45	0.00E+00	**0.28**	0.00E+00	0.47	0.00E+00			**0.77**	0.00E+00	**0.84**	0.00E+00	**0.40**	0.00E+00	0.54	0.00E+00
Random	**0.22**	0.00E+00	**0.14**	0.00E+00	**0.25**	0.00E+00	**0.23**	0.00E+00			**0.74**	5.55E-169	**0.14**	0.00E+00	**0.30**	0.00E+00
TACTIC	**0.13**	0.00E+00	**0.08**	0.00E+00	**0.16**	0.00E+00	**0.16**	0.00E+00	**0.26**	5.55E-169			**0.06**	0.00E+00	**0.18**	7.40E-299
DeepJanus	0.55	0.00E+00	**0.36**	0.00E+00	**0.56**	0.00E+00	**0.60**	0.00E+00	**0.86**	0.00E+00	**0.94**	0.00E+00			**0.63**	0.00E+00
DeepJanus$_{GAN}$	**0.43**	0.00E+00	**0.28**	0.00E+00	**0.44**	0.00E+00	0.46	0.00E+00	**0.70**	0.00E+00	**0.82**	7.40E-299	**0.37**	0.00E+00		
Pixel-based distance																
	DESIGNATE$_{single}$		DESIGNATE		DESIGNATE$_{pix}$		DESIGNATE$_{NoGAN}$		Random		TACTIC		DeepJanus		DeepJanus$_{GAN}$	
	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value	A_{12}	p-value
DESIGNATE$_{single}$			**0.39**	0.00E+00	0.48	0.00E+00	**0.95**	0.00E+00	0.50	1.00E+00	**0.44**	3.77E-12	**0.86**	0.00E+00	**0.57**	0.00E+00
DESIGNATE	**0.61**	0.00E+00			**0.58**	0.00E+00	**0.96**	0.00E+00	**0.61**	0.00E+00	**0.59**	2.74E-22	**0.90**	0.00E+00	**0.67**	0.00E+00
DESIGNATE$_{pix}$	0.52	0.00E+00	**0.42**	0.00E+00			0.94	0.00E+00	0.52	0.00E+00	0.46	3.76E-05	**0.86**	0.00E+00	**0.59**	0.00E+00
DESIGNATE$_{NoGAN}$	**0.05**	0.00E+00	**0.04**	0.00E+00	**0.06**	0.00E+00			**0.05**	0.00E+00	**0.01**	0.00E+00	**0.29**	0.00E+00	**0.08**	0.00E+00
Random	0.50	1.00E+00	**0.39**	0.00E+00	0.48	0.00E+00	**0.95**	0.00E+00			**0.44**	3.77E-12	**0.86**	0.00E+00	**0.57**	0.00E+00
TACTIC	**0.56**	3.77E-12	**0.41**	2.74E-22	**0.54**	3.76E-05	**0.99**	0.00E+00	**0.56**	3.77E-12			**0.96**	0.00E+00	**0.65**	2.11E-64
DeepJanus	**0.14**	0.00E+00	**0.10**	0.00E+00	**0.14**	0.00E+00	**0.71**	0.00E+00	**0.14**	0.00E+00	**0.04**	0.00E+00			**0.19**	0.00E+00
DeepJanus$_{GAN}$	**0.43**	0.00E+00	**0.33**	0.00E+00	**0.41**	0.00E+00	**0.92**	0.00E+00	**0.43**	0.00E+00	**0.35**	2.11E-64	**0.81**	0.00E+00		

Note: tangible (based on A_{12}) differences in bold; for each pair of approaches, the best approach is the one with $A_{12} > 0.50$ and its name on the row.

distance implies changing the elements depicted in the images and, consequently, the pixel distance. This result may thus be explained by the fact that most of the changes in simulator inputs lead to pixel-based distances that are significant (i.e., above $T_{diversity}$), whereas only pixel changes that have a strong impact on the elements in the images can result into a significant feature-based distance. Consequently, feature-based distance simplifies the task of the search algorithm to explore the input space towards the search objectives. Also, DESIGNATE performs significantly better than TACTIC and Random, with small and high effect size, respectively. As per feature-based distance, Random presents lower diversity than DESIGNATE (ranges are [0.3,0.76] for Random vs. [0.14,1.00] for DESIGNATE), thus confirming that random sampling is not sufficient to identify the unsafe parts of the input space. TACTIC's third quartile shows that only 25% of TACTIC's image pairs have a pixel-based distance higher than 0.68 (i.e., 68% of the pixels differ) while 95% (5th percentile) of the image pairs generated by DESIGNATE present a distance higher than 0.94 (almost all the pixels differ). Last, the pixel-based distance of DESIGNATE images is significantly larger than that of DeepJanus and DeepJanus$_{GAN}$ images, with medium to large effect sizes, thus confirming the differences discussed above.

In the Martian environment, DESIGNATE achieved a median diversity of 14.75 for feature-based distance that is significantly higher than DESIGNATE$_{single}$ (10.59), DESIGNATE$_{pix}$ (11.92), the random baseline (11.36), DeepJanus (12.81), and DeepJanus$_{GAN}$ (9.52). However, DESIGNATE$_{NoGAN}$ outperformed DESIGNATE in terms of feature-based diversity, which we attribute to DESIGNATE$_{NoGAN}$ relying on the lower-fidelity outputs of the simulator; indeed, such simulator images are likely to cause erroneous outputs and therefore DESIGNATE$_{NoGAN}$ is likely to generate images that are spread throughout the whole input space, thus leading to larger diversity. In all likelihood, this phenomenon is not observed with DeepJanus because, by generating images on the boundaries of the input space, it likely leads to more similar images (e.g., images containing roads where cars can be positioned far away).

For pixel-based diversity, DESIGNATE outperformed all the other approaches with a median pixel distance of 1.00. Instead, with feature-based distance, DESIGNATE$_{NoGAN}$ outperformed DESIGNATE, such a difference is likely due to the nature of the two distance metrics; indeed, the pixel distance may capture dissimilarities that are practically irrelevant. For example, two images may have slightly different luminosity,

TABLE VIII: RQ1-Diversity assessment. Descriptive statistics for diversity across the generated images for the Martian environment.

	Feature Distance Diversity						
	min	**max**	**median**	**5th percentile**	**1st quartile**	**3rd quartile**	**Average**
DESIGNATE$_{single}$	0.00	27.17	10.59	5.67	8.70	12.35	10.58
DESIGNATE	0.00	26.24	14.75	8.09	11.22	18.58	14.83
DESIGNATE$_{pix}$	0.82	25.43	11.92	7.79	9.99	14.38	12.37
DESIGNATE$_{NoGAN}$	1.01	**32.49**	**16.79**	**10.23**	**14.35**	**19.54**	**16.83**
Random	0.00	28.70	11.36	7.51	9.44	13.66	11.89
DeepJanus	**2.18**	31.67	12.81	7.16	10.58	18.64	14.48
DeepJanus$_{GAN}$	0.00	25.97	9.52	6.13	7.80	12.25	10.54
	Pixel Distance Diversity						
	min	**max**	**median**	**5th percentile**	**1st quartile**	**3rd quartile**	**Average**
DESIGNATE$_{single}$	**0.96**	**1.00**	0.99	0.97	0.98	0.99	0.99
DESIGNATE	0.95	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**1.00**
DESIGNATE$_{pix}$	0.95	**1.00**	0.98	0.96	0.97	0.99	0.98
DESIGNATE$_{NoGAN}$	0.95	**1.00**	0.98	0.96	0.97	0.99	0.98
Random	**0.96**	**1.00**	0.98	0.97	0.98	0.99	0.98
DeepJanus	0.95	0.99	0.96	0.95	0.96	0.97	0.96
DeepJanus$_{GAN}$	0.95	**1.00**	0.99	0.97	0.98	0.99	0.98

Note: best (i.e., highest or most diverse) results (per column) in bold.

TABLE IX: RQ1-Diversity assessment for the Martian case study. p-values and A_{12} for the data in Table VIII.

Feature-based distance														
	DESIGNATE$_{single}$		**DESIGNATE**		**DESIGNATE**$_{pix}$		**DESIGNATE**$_{NoGAN}$		**Random**		**DeepJanus**		**DeepJanus**$_{GAN}$	
	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**
DESIGNATE$_{single}$			0.43	0.00E+00	0.52	0.00E+00	0.15	0.00E+00	0.51	0.00E+00	0.34	0.00E+00	0.62	0.00E+00
DESIGNATE	**0.57**	0.00E+00			**0.59**	0.00E+00	**0.19**	0.00E+00	**0.59**	0.00E+00	**0.40**	0.00E+00	**0.68**	0.00E+00
DESIGNATE$_{pix}$	0.48	0.00E+00	**0.41**	0.00E+00			**0.13**	0.00E+00	0.50	0.00E+00	**0.33**	0.00E+00	**0.62**	0.00E+00
DESIGNATE$_{NoGAN}$	**0.85**	0.00E+00	**0.81**	0.00E+00	**0.87**	0.00E+00			**0.86**	0.00E+00	**0.70**	0.00E+00	**0.88**	0.00E+00
Random	0.49	0.00E+00	**0.41**	0.00E+00	0.50	0.00E+00	**0.14**	0.00E+00			**0.33**	0.00E+00	**0.62**	0.00E+00
DeepJanus	**0.66**	0.00E+00	**0.60**	0.00E+00	**0.67**	0.00E+00	**0.30**	0.00E+00	**0.67**	0.00E+00			**0.74**	0.00E+00
DeepJanus$_{GAN}$	**0.38**	0.00E+00	**0.32**	0.00E+00	**0.38**	0.00E+00	**0.12**	0.00E+00	**0.38**	0.00E+00	**0.26**	0.00E+00		
Pixel-based distance														
	DESIGNATE$_{single}$		**DESIGNATE**		**DESIGNATE**$_{pix}$		**DESIGNATE**$_{NoGAN}$		**Random**		**DeepJanus**		**DeepJanus**$_{GAN}$	
	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**
DESIGNATE$_{single}$			**0.36**	2.23E-211	**0.60**	3.56E-106	**0.81**	0.00E+00	0.55	1.08E-26	**0.99**	0.00E+00	0.55	1.32E-35
DESIGNATE	**0.64**	2.23E-211			**0.71**	0.00E+00	**0.83**	0.00E+00	**0.68**	0.00E+00	**0.93**	0.00E+00	**0.68**	0.00E+00
DESIGNATE$_{pix}$	**0.40**	3.56E-106	**0.29**	0.00E+00			**0.77**	0.00E+00	**0.44**	4.51E-40	**0.98**	0.00E+00	0.45	0.00E+00
DESIGNATE$_{NoGAN}$	**0.19**	0.00E+00	**0.17**	0.00E+00	**0.23**	0.00E+00			**0.20**	0.00E+00	**0.88**	0.00E+00	**0.21**	0.00E+00
Random	0.45	1.08E-26	**0.32**	0.00E+00	**0.56**	4.51E-40	**0.80**	0.00E+00			**0.99**	0.00E+00	0.51	0.00E+00
DeepJanus	**0.01**	0.00E+00	**0.07**	0.00E+00	**0.02**	0.00E+00	**0.12**	0.00E+00	**0.01**	0.00E+00			**0.01**	0.00E+00
DeepJanus$_{GAN}$	0.45	1.32E-35	**0.32**	0.00E+00	0.55	1.11E-34	**0.79**	0.00E+00	0.49	1.19E-01	**0.99**	0.00E+00		

Note: tangible (based on A_{12}) differences in bold; for each pair of approaches, the best approach is the one with $A_{12} < 0.50$ and its name on the row.

thus leading to high pixel-based distance, but contain the same objects thus yielding low feature-based distance.

To summarize, DESIGNATE$_{single}$ fares better than DESIGNATE and DESIGNATE$_{pix}$ in terms of accuracy. However, DESIGNATE yields more diversity (both in terms of features and pixels) than DESIGNATE$_{single}$ and DESIGNATE$_{pix}$. In other words, DESIGNATE and DESIGNATE$_{single}$ represent two different trade-offs and which one to adopt depends on whether accuracy or diversity is prioritized. Since DESIGNATE$_{single}$ focuses solely on minimizing DNN accuracy, without explicitly considering diversity, it is highly effective at generating images that cause DNN failures, but at the cost of producing less diverse inputs. We should note, however, that DESIGNATE is nevertheless able to detect a large number of cases where the DNN predicts poorly, as demonstrated by the zero accuracy cases representing more that 25% of generated inputs (first quartile of the IoU_{car} distribution in Table II). Last, there is no particular reason to rely on DESIGNATE$_{pix}$ as it provides no advantages in terms of diversity or accuracy. With respect to other alternative techniques, as presented above, they tend to either not be effective at triggering failures or generate many unrealistic and irrelevant test inputs.

D. RQ2: Retraining

1) Design and measurements: We use the images generated by the approaches considered for RQ1 to retrain our case-study subject DNN. Further, to account for non-determinism, we repeat the process 10 times. For each approach and each retraining iteration, we use the images generated in one run of the approach to augment the Cityscapes training set (2975 images), for the urban environment, and the AI4MARS training set (16064 images), for the Martian environment. We then retrain the DNN using the augmented dataset, maintaining the same configuration settings (100 epochs) as those used by the DeeplabV3 developers [12]. To avoid introducing bias due to a different number of images in the augmented datasets, for each approach, we randomly select up to 900 images from the set of images generated in RQ1. In addition to the approaches presented in RQ1 (TACTIC, random baseline, DeepJanus, and DeepJanus$_{GAN}$), we compare DESIGNATE to the training of the DeeplabV3 model for 100 additional epochs using the whole original training set, to determine the degree of improvement achievable with a longer training time, without generating additional training images.

We measure the IoU_{car} obtained by all the retrained DNNs on the Cityscapes validation set, which enables us to assess if retraining the DNN with test images improves its performance with real-world inputs, the target of the DNN under test. We analyze on the IoU_{car} distribution and, as per RQ1, we discuss

significance of the differences and effect sizes by relying on the results of Mann-Whitney U-test and Vargha and Delaney's $A12$, respectively.

For the Martian environment, we measure the $mIoU$ obtained by all the retrained DNNs on the AI4MARS test set (966 images).

2) Results: Tables X and XII provide descriptive statistics for IoU_{car} and $mIoU$ from testing the retrained models in the urban and Martian environments, respectively. These tables summarize the performance improvements achieved through retraining with outputs generated by different approaches. Tables XI and XIII present the p-values and $\hat{A}_{12}$ values from statistical comparisons of IoU_{car} and $mIoU$ across retrained models, capturing the significance and effect sizes of the observed differences.

For the urban environment, the DNN retrained with the output of DESIGNATE has the highest median performance at 0.90, followed by DeepJanus (0.87), DESIGNATE$_{single}$ (0.86), DeepJanus$_{GAN}$ (0.86), DESIGNATE$_{NoGAN}$ (0.85), and DESIGNATE$_{pix}$ (0.85). DESIGNATE not only improves ($+0.08$) over the original pre-trained DeeplabV3 model (median $IoU_{car} = 0.82$), but fare much better than the DeeplabV3 retrained with the cityscapes training set (0.84, $+0.06$), a set of random realistic images (0.85, $+0.05$), TACTIC (0.84, $+0.06$), DeepJanus (0.87, $+0.05$) and DeepJanus$_{GAN}$ (0.84, $+0.04$) outputs.

The difference between DESIGNATE and its variants is statistically ($p - value < 0.05$) and practically ($\hat{A}_{12} \geq 0.56$) significant, which indicates that integrating a diversity objective and feature distance as fitness functions leads to better results.

Also the difference between DESIGNATE and the two DeepJanus variants is statistically and practically significant. The fact that DESIGNATE outperforms DeepJanus$_{GAN}$ indicates that focusing on corner cases (i.e., the frontier of behaviours found by DeepJanus$_{GAN}$) is not effective to retrain a DNN. Further, the likelihood of DeepJanus and DeepJanus$_{GAN}$ performing better than random is very low (0.55 and 0.52).

In general, we can conclude that relying on simulator images only (i.e., DESIGNATE$_{NoGAN}$ and DeepJanus) leads to worse results compared to retraining with GAN-generated images, thus justifying our choice.

Further, models retrained relying on search to drive the generation of scenarios (DESIGNATE and DeepJanus variants) outperform the other approaches. Such results suggest that relying on changes in weather conditions (i.e., TACTIC) limits retraining and does not yield better results than what is achievable with additional training epochs. Further, TACTIC leads to results that do not significantly differ from the ones obtained with the original DeeplabV3 model, with a model retrained with additional epochs, or with a set of random realistic images ($0.46 < A_{12} < 0.54$).

For the Martian environment, DESIGNATE again demonstrated superior performance, achieving the highest median $mIoU$ (0.53, +0.09 over the original model). These results highlight the effectiveness of DESIGNATE in generating both failure-inducing and diverse inputs, which, when used for retraining, improves DNN robustness in challenging environments. The difference between DESIGNATE and its variants is statistically ($p - value < 0.05$) and practically ($\hat{A}_{12} \geq 0.56$) significant, thus confirming the effectiveness of integrating a diversity objective and feature distance as fitness functions.

Like in the urban environment, DESIGNATE ($mIoU = 0.53$) outperforms DeepJanus (0.50) and DeepJanus$_{GAN}$ (0.50), thus confirming to be a better choice for retraining. Also, DESIGNATE outperforms a random approach ($mIoU = 0.49, +0.05$), with a practically significant difference ($A_{1,2} = 0.61$), while DeepJanus and DeepJanus$_{GAN}$ perform similar to random ($A_{1,2} = 0.50$).

Concluding, our results show that DESIGNATE leads to the best retraining results, thus demonstrating that (1) simulator images are not as effective as realistic images to retrain DNNs, (2) relying on images belonging to the frontier of DNN behaviours is not the best approach for DNN retraining, and (3) relying on GANs to generate images that simply change weather conditions is not as effective as combining GAN with a simulator to generate realistic images for DNN retraining.

E. Threats to validity

To address *internal validity* threats, we carefully tested our implementation of DESIGNATE and its variants, and further relied on the original version of TACTIC and DeepJanus provided by their authors. Still related to internal validity, to prevent the generation of unrealistic images with GANs, our approach integrates a strategy to eliminate images that are likely to be unrealistic (i.e., images where the DNN performs better with the simulated image). Such heuristic seems effective since RQ2 results show that retraining the DNN with DESIGNATE images leads to improved results with a real-world test set. If DESIGNATE images were not realistic, the performance of the DNN would have shown limited or no improvement because it would have learned from images that are unrealistic, which is what happens with approaches without GAN. However, it is still possible that our filter may not be fully effective. In the future, we aim to extend our approach with additional methods such as histogram analysis [55] and other state-of-the-art approaches [45], [60] for the identification of out-of-domain and out-of-distribution images.

To address *conclusion validity* threats, we repeated our experiments 10 times and relied on non-parametric statistical analysis (Mann Whitney U-Test) and effect size (Vargha and Delaney's) tests.

To address *generalizability threats*, we considered two instances of a state-of-the-art DNN implementing a computer vision task that is key for safety-critical systems (i.e., image segmentation), for two very different environments (urban and Martian). We therefore expect our results to be applicable across many industrial contexts, from automotive to robotics.

F. Data availability

Our replication package is available online [6].

TABLE X: RQ2. IoU_{car} results obtained with the DeeplabV3 model retrained using the outputs of the different approaches.

	$\mathbf{IoU_{car}}$ distribution						
Retraining set	**min**	**max**	**median**	**5th percentile**	**1st quartile**	**3rd quartile**	**average**
None (original DeeplabV3)	0.00	0.97	0.82	0.00	0.62	0.91	0.71
Cityscapes (Retrained DNN)	0.00 (+0.00)	0.97 (+0.00)	0.84 (+0.02)	0.00 (+0.00)	0.70 (+0.08)	0.91 (+0.00)	0.74 (+0.03)
DESIGNATE$_{single}$	0.00 (+0.00)	**0.98 (+0.01)**	0.86 (+0.04)	0.00 (+0.00)	0.73 (+0.11)	0.92 (+0.01)	0.76 (+0.05)
DESIGNATE	0.00 (+0.00)	**0.98 (+0.01)**	**0.90 (+0.08)**	**0.34 (+0.34)**	**0.80 (+0.18)**	**0.94 (+0.03)**	**0.82 (+0.11)**
DESIGNATE$_{pix}$	0.00 (+0.00)	**0.98 (+0.01)**	0.85 (+0.03)	0.00 (+0.00)	0.70 (+0.08)	0.92 (+0.01)	0.75 (+0.04)
DESIGNATE$_{NoGAN}$	0.00 (+0.00)	0.97 (+0.00)	0.86 (+0.04)	0.00 (+0.00)	0.73 (+0.1)	0.92 (+0.01)	0.76 (+0.05)
Random	0.00 (+0.00)	**0.98 (+0.01)**	0.85 (+0.03)	0.00 (+0.00)	0.70 (+0.08)	0.92 (+0.01)	0.74 (+0.03)
TACTIC	0.00 (+0.00)	0.97 (+0.00)	0.84 (+0.02)	0.00 (+0.00)	0.69 (+0.07)	0.92 (+0.01)	0.74 (+0.03)
DeepJanus	0.00 (+0.00)	**0.98 (+0.01)**	0.87 (+0.05)	0.00 (+0.00)	0.75 (+0.13)	0.93 (+0.02)	0.77 (+0.06)
DeepJanus$_{GAN}$	0.00 (+0.00)	0.97 (+0.00)	0.86 (+0.04)	0.00 (+0.00)	0.73 (+0.11)	0.92 (+0.01)	0.76 (+0.05)

Note: best (i.e., highest or most diverse) results (per column) in bold.

TABLE XI: RQ2. p-values and A_{12} for the data in Table X.

	Pre-trained DeeplabV3		**Retrained DeeplabV3**		**DESIGNATE**$_{single}$		**DESIGNATE**		**DESIGNATE**$_{pix}$		**DESIGNATE**$_{NoGAN}$		**Random**		**TACTIC**		**DeepJanus**		**DeepJanus**$_{GAN}$	
	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	$\mathbf{A_{12}}$	p-value	VDA	p-value	VDA	p-value	VDA	p-value
Pre-trained DeeplabV3			0.47	0.06	**0.44**	0.00E+00	**0.35**	0.00E+00	0.46	0.00E+00	**0.44**	0.00E+00	0.46	0.00E+00	0.47	2.00E-02	0.42	0.00E+00	**0.44**	0.00E+00
Retrained DeeplabV3	0.53	6.00E-02			0.47	0.00E+00	**0.37**	0.00E+00	0.49	5.00E-02	**0.46**	0.00E+00	0.49	6,00E-02	0.49	5.00E-01	**0.44**	0.00E+00	0.47	0.00E+00
DESIGNATE$_{single}$	**0.56**	0.00E+00	0.53	0.00			**0.40**	0.00E+00	0.52	0.00E+00	0.49	0.00E+00	0.52	0,00E+00	0.53	0.00E+00	0.47	0.00E+00	0.50	7.00E-01
DESIGNATE	**0.65**	0.00E+00	**0.63**	0.00	**0.60**	0.00E+00			**0.61**	0.00E+00	**0.59**	0.00E+00	**0.61**	0,00E+00	**0.62**	0.00E+00	**0.57**	0.00E+00	**0.59**	0.00E+00
DESIGNATE$_{pix}$	0.54	0.00E+00	0.51	0.05	0.48	0.00E+00	**0.39**	0.00E+00			0.48	0.00E+00	0.50	8,40E-01	0.51	1.40E-01	0.45	0.00E+00	0.48	0.00E+00
DESIGNATE$_{NoGAN}$	**0.56**	0.00E+00	0.54	0.00E+00	0.51	1.00E-01	**0.41**	0.00E+00	0.52	0.00E+00			0.53	0.00E+00	0.53	0.00E+00	0.48	0.00E+00	0.51	2.70E-01
Random	0.54	0.00E+00	0.51	6.00E-02	0.48	0.00E+00	**0.39**	0,00E+00	0.50	8,40E-01	0.47	0.00E+00			0.51	1.90E-01	0.45	0.00E+00	0.48	0.00E+00
TACTIC	0.53	2.00E-02	0.51	0.50	0.47	0.00E+00	**0.38**	0.00E+00	0.49	1.40E-01	0.47	0.00E+00	0.49	1,90E-01			**0.44**	0.00E+00	0.47	0.00E+00
DeepJanus	**0.58**	0.00E+00	**0.56**	0.00E+00	0.53	0.00E+00	**0.43**	0.00E+00	0.55	0.00E+00	0.52	0.00E+00	0.55	0.00E+00	**0.56**	0.00E+00			0.53	0.00E+00
DeepJanus$_{GAN}$	**0.56**	0.00E+00	0.53	0.00E+00	0.50	7.00E-01	**0.41**	0.00E+00	0.52	0.00E+00	0.49	2.70E-01	0.52	0.00E+00	0.53	0.00E+00	0.47	0.00E+00		

Note: tangible (based on A_{12}) differences in bold; for each pair of approaches, the best approach is the one with $A_{12} > 0.50$ and its name on the row.

TABLE XII: RQ2. $mIoU$ results obtained with the DeeplabV3 model retrained using the outputs of the different approaches for the Martian environment.

Retraining set	**min**	**max**	**median**	**5th percentile**	**1st quartile**	**3rd quartile**	**Average**
None (Original DeeplabV3)	0.00	0.99	0.44	0.01	0.26	**0.77**	0.49
AI4Mars training (Retrained DNN)	**0.01 (+0.01)**	1.00 (+0.01)	0.47 (+0.03)	0.17 (+0.16)	0.33 (+0.07)	0.63 (-0.14)	0.49 (+0.00)
DESIGNATE$_{single}$	0.00 (+0.00)	1.00 (+0.01)	0.50 (+0.06)	0.19 (+0.18)	0.34 (+0.08)	0.67 (-0.10)	0.52 (+0.03)
DESIGNATE	0.00 (+0.00)	1.00 (+0.01)	**0.53 (+0.09)**	0.19 (+0.18)	**0.38 (+0.12)**	0.69 (-0.08)	**0.53 (+0.04)**
DESIGNATE$_{pix}$	0.00 (+0.00)	1.00 (+0.01)	0.50 (+0.06)	0.15 (+0.14)	0.36 (+0.09)	0.68 (-0.09)	0.52 (+0.02)
DESIGNATE$_{NoGAN}$	0.00 (+0.00)	1.00 (+0.01)	0.46 (+0.02)	0.07 (+0.06)	0.30 (+0.04)	0.64 (-0.13)	0.47 (-0.02)
Random	0.00 (+0.00)	1.00 (+0.01)	0.49 (+0.05)	0.12 (+0.10)	0.33 (+0.07)	0.68 (-0.09)	0.51 (+0.02)
DeepJanus	0.00 (+0.00)	1.00 (+0.01)	0.50 (+0.06)	0.19 (+0.18)	0.36 (+0.09)	0.66 (-0.10)	0.52 (+0.03)
DeepJanus$_{GAN}$	0.00 (+0.00)	1.00 (+0.01)	0.50 (+0.06)	**0.20 (+0.19)**	0.36 (+0.09)	0.67 (-0.10)	0.52 (+0.03)

Note: best (i.e., highest or most diverse) results (per column) in bold.

TABLE XIII: RQ2. p-values and A_{12} for the data in Table XII

	Original		**Original_retrained**		**DESIGNATE**$_{single}$		**DESIGNATE**		**DESIGNATE**$_{pix}$		**DESIGNATE**$_{NoGAN}$		**Random**		**DeepJanus**		**DeepJanus**$_{GAN}$	
	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**	$\mathbf{A_{12}}$	**p-value**		
Original			0.47	1.60E-01	0.47	1.00E-02	0.39	0.00E+00	0.43	0.00E+00	0.51	5.70E-01	0.48	1.60E-01	**0.44**	0.00E+00	**0.44**	0.00E+00
Retrained DNN	0.53	1.60E-01			0.50	7.40E-01	0.44	0.00E+00	0.48	2.30E-01	0.54	0.00E+00	0.52	1.80E-01	0.49	3.90E-01	0.49	3.10E-01
DESIGNATE$_{single}$	0.53	1.00E-02	0.50	7.40E-01			0.43	0.00E+00	0.47	0.00E+00	0.54	0.00E+00	0.51	3.00E-02	0.48	0.00E+00	0.48	0.00E+00
DESIGNATE	**0.61**	0.00E+00	**0.56**	0.00E+00	**0.57**	0.00E+00			0.54	0.00E+00	**0.61**	0.00E+00	**0.58**	0.00E+00	0.55	0.00E+00	0.55	0.00E+00
DESIGNATE$_{pix}$	**0.57**	0.00E+00	0.52	2.30E-01	0.53	0.00E+00	0.46	0.00E+00			**0.57**	0.00E+00	0.54	0.00E+00	0.51	1.00E-02	0.51	5.00E-02
DESIGNATE$_{NoGAN}$	0.49	5.70E-01	0.46	0.00E+00	0.46	0.00E+00	**0.39**	0.00E+00	**0.43**	0.00E+00			0.47	0.00E+00	**0.44**	0.00E+00	**0.44**	0.00E+00
Random	0.52	1.60E-01	0.48	1.80E-01	0.49	3.00E-02	**0.42**	0.00E+00	0.46	0.00E+00	0.53	0.00E+00			0.47	0.00E+00	0.47	0.00E+00
DeepJanus	**0.56**	0.00E+00	0.51	3.90E-01	0.52	0.00E+00	0.45	0.00E+00	0.49	1.00E-02	**0.56**	0.00E+00	0.53	0.00E+00			0.50	6.30E-01
DeepJanus$_{GAN}$	**0.56**	0.00E+00	0.51	3.10E-01	0.52	0.00E+00	0.45	0.00E+00	0.49	5.00E-02	**0.56**	0.00E+00	0.53	0.00E+00	0.50	6.30E-01		

Note: tangible (based on A_{12}) differences in bold; for each pair of approaches, the best approach is the one with $A_{12} > 0.50$ and its name on the row.

V. Related work

Our work relates to DNN testing approaches leveraging generative neural networks, which are mainly adopted to perform adversarial testing or generating test images by altering image elements.

Adversarial testing technique have a different purpose than DESIGNATE; indeed, they aim to assess whether DNNs provide correct predictions when attackers purposely alter inputs. DESIGNATE, instead, aims to ensure correct DNN behaviour within its operational design domain [67]. The state of the art technique is FuzzGan [25], which relies on an auxiliary classifier GAN [40] that is trained to learn the representation of images belonging to specific classes and triggering specific neuron activations. During testing, the GAN is used to generate inputs that likely triggers certain neuron activations; testing terminates when the desired neuron coverage is achieved [29]. FuzzGan outperforms CAGFuzz [66], an approach relying on CycleGAN for the same purpose. An alternative approach to adversarial testing has been recently proposed by Yuan et al.; it consists of performing search-based exploration on a manifold obtained through a GAN [64]. Yuan's approach generates valid images, as opposed to approaches mutating the latent space captured by variational autoencoders [15], [33]; further, it outperforms other DNN testing approaches (DeepHunter [62], TensorFuzz [41], DeepTest [52]) in terms of number of reported failures, and it is the only approach not reducing accuracy after retraining but slightly increasing it. Such results show that inputs generated by adversarial attacks

tend to reduce accuracy, thus not being an appropriate solution to our problem. DESIGNATE, in contrast to Yuan's approach, does not focus on adversarial testing, but aims at improving DNN accuracy. Finally, DESIGNATE can be applied to classifier and regression DNNs, while Yuan's approach can only be applied to the former because, for regression tasks, it is not possible to automatically derive a ground truth for the generated images.

Inspired by metamorphic testing approaches relying on image modifications [42], [52], several techniques rely on GANs to change the weather conditions in input images, and report failures when the DNN output differs from the original inputs. For example, AdversarialStyle relies on MUNIT [59], while DeepRoad relies on UNIT [65]. The state-of-the-art approach is TACTIC [34], which relies on MUNIT to introduce environmental changes (see Section IV-B) and, in addition, relies on a multi-objective search approach to further alter the MUNIT style vector to generate images that are diverse (using a fitness based on neuron coverage) and leading to the worst prediction error. Unfortunately, TACTIC can alter only the environmental conditions of existing images, which limits input diversity, while DESIGNATE can leverage a simulator to generate inputs that cannot be obtained by altering a given image (e.g., rotating the view). Such difference between DESIGNATE and TACTIC leads to a much better accuracy obtained by DESIGNATE during retraining, as demonstrated by our empirical assessment.

Without leveraging GANs, *semInFuzz* [61] relies on the segmentation masks provided by Cityscapes to identify objects (e.g., a car) to be extracted from seed images and paste them into the images to be used as test inputs. Being based on simple transformations (e.g., cut and paste), it may lead to unrealistic images (e.g., proportions, perspective, or object blending).

A large number of approaches rely on meta-heuristic search to drive simulators for cost effective testing [1], [20], [22], [27], [28], [43], with some recent solutions maximizing input diversity to maximize the number of diverse fault being detected [44], [69]–[71]. Recently, reinforcement learning has also shown to be an effective solution for DNN testing [26]. However, all these approaches do not lead to realistic images because they do not rely on GANs; instead, we demonstrated that the combination of GANs with meta-heuristic search, leveraging simulators, leads to better results for both testing and retraining of vision DNNs.

To the best of our knowledge, DESIGNATE is the first technique combining meta-heuristic search, simulation, and GANs, to test and retrain DNNs for safety-critical tasks, based on realistic images. In other contexts, preliminary works report on the feasibility of relying on CycleGAN to generate, from simulator outputs, realistic sea images [16], sonar data [35], and landscapes [2]; other work proposes testing object detection DNNs by applying GANs to maritime simulator outputs but do empirically assess the idea [9]. In the software engineering context, the closer work is that of Stocco et al. [50], who evaluated the differences in testing results obtained in a simulated environment versus those from a physical setting, with a small-size vehicle (i.e., Donkeycar [14]). They use CycleGAN to generate realistic images from simulated driving scenes and train a telemetry predictor. In line with our findings, their results show that images generated by GANs can be used to successfully train a DNN used in the physical world. However, because of the nature of their study, they do not demonstrate that simulators and GANs can be integrated within a search-based process; instead, our work demonstrates how to perform such integration in an effective way, without leading to the generation of images that fail simply because unrealistic.

VI. Conclusion

In this paper, we proposed DESIGNATE, a novel approach that integrates GANs with meta-heuristic search and simulators to enhance the testing and retraining of DNNs. By leveraging GANs, DESIGNATE transforms simulator-generated images into high-fidelity, realistic inputs that closely resemble real-world data distributions. This transformation addresses the fidelity gap inherent in simulator-based testing, ensuring that the observed DNN failures are indicative of real-world performance limitations.

Our empirical investigation, conducted using a state-of-the-art DNN for road and martian terrains segmentation, demonstrates the effectiveness of DESIGNATE in identifying failure-inducing inputs and maximizing input diversity. DESIGNATE consistently outperformed the baseline approaches, including the state-of-the-art methods like TACTIC and DeepJanus, by generating inputs that led to worse DNN performance and more diverse failure scenarios. Furthermore, the realistic images generated by DESIGNATE proved to be valuable for DNN retraining, leading to improved performance on real-world images, thus outperforming alternative solutions and demonstrating that: (1) simulator images are not as effective as realistic images for DNN retraining (2) relying on GANs to generate images that simply change weather conditions is not as effective as combining GAN with a simulator to generate realistic images for DNN retraining. Based on our findings, we suggest that future work on DNN testing, instead of simply relying on simulators, should integrate GAN components transforming simulator outputs into more realistic ones.

Acknowledgments

The experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg (see http://hpc.uni.lu). This work has been partially supported by the ESA contract RFP/3-17931/22/NL/GLC/my, TIA (Test, Improve, Assure). Lionel Briand was partly supported by the Canada Research Chair and Discovery Grant programs of the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Science Foundation Ireland under Grant 13/RC/2094-2.

References

[1] Raja Ben Abdessalem, Shiva Nejati, Lionel C Briand, and Thomas Stifter. Testing vision-based control systems using learnable evolutionary algorithms. In *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*, pages 1016–1026. IEEE, 2018.

[2] Silvia Arellano, Beatriz Otero, Tomasz Piotr Kucner, and Ramon Canal. A 3d terrain generator: Enhancing robotics simulations with gans. In Giuseppe Nicosia, Varun Ojha, Emanuele La Malfa, Gabriele La Malfa, Panos M. Pardalos, and Renato Umeton, editors, *Machine Learning, Optimization, and Data Science*, pages 212–226, Cham, 2024. Springer Nature Switzerland.

[3] Mohammed Attaoui, Hazem Fahmy, Fabrizio Pastore, and Lionel Briand. Black-box safety analysis and retraining of dnns based on feature extraction and clustering. *ACM Trans. Softw. Eng. Methodol.*, 32(3), apr 2023.

[4] Mohammed Oualid Attaoui, Hanene Azzag, Mustapha Lebbah, and Nabil Keskes. Multi-objective data stream clustering. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 113–114, 2020.

[5] Mohammed Oualid Attaoui, Hazem Fahmy, Fabrizio Pastore, and Lionel Briand. Supporting safety analysis of image-processing dnns through clustering-based approaches. *ACM Trans. Softw. Eng. Methodol.*, 33(5), jun 2024.

[6] Mohammed Oualid Attaoui, Fabrizio Pastore, and Lionel Briand. Replication package, 2024. https://doi.org/10.5281/zenodo.14535408.

[7] BeamNG GmbH. Beamng.drive. https://www.beamng.com, 2015.

[8] J. Blank and K. Deb. pymoo: Multi-objective optimization in python. *IEEE Access*, 8:89497–89509, 2020.

[9] Andreas Brandsæter and Ottar L Osen. Assessing autonomous ship navigation using bridge simulators enhanced by cycle-consistent adversarial networks. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 237(2):508–517, 2023.

[10] CARLA Simulator Team. Carla: An open urban driving simulator. https://github.com/carla-simulator/carla, 2017.

[11] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 40(04):834–848, April 2018.

[12] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[13] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[14] Donkeycar community. Donkeycar. https://www.donkeycar.com/, 2024.

[15] Swaroopa Dola, Matthew B. Dwyer, and Mary Lou Soffa. Distribution-aware testing of neural networks using generative models. *Proceedings - International Conference on Software Engineering*, pages 226–237, 2021.

[16] Yuxuan Dong, Peng Wu, Sen Wang, and Yuanchang Liu. Shipgan: Generative adversarial network based simulation-to-real image translation for ships. *Applied Ocean Research*, 131:103456, 2023.

[17] Epic Games. Unreal engine.

[18] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[19] Hazem Fahmy, Fabrizio Pastore, Mojtaba Bagherzadeh, and Lionel Briand. Supporting deep neural network safety analysis and retraining through heatmap-based unsupervised learning. *IEEE Transactions on Reliability*, pages 1–17, 2021.

[20] Hazem Fahmy, Fabrizio Pastore, Lionel Briand, and Thomas Stifter. Simulator-based explanation and debugging of hazard-triggering events in dnn-based safety-critical systems. *ACM Trans. Softw. Eng. Methodol.*, 32(4), may 2023.

[21] A. Gambi, M. Mueller, and G. Fraser. Asfault: Testing self-driving car software using search-based procedural content generation. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pages 27–30, May 2019.

[22] Alessio Gambi, Marc Mueller, and Gordon Fraser. Automatically testing self-driving cars with search-based procedural content generation. *ISSTA 2019 - Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 273–283, 2019.

[23] Christoph Gladisch, Thomas Heinz, Christian Heinzemann, Jens Oehlerking, Anne von Vietinghoff, and Tim Pfitzer. Experience paper: search-based testing in automated driving control applications. In *Proceedings of the 34th IEEE/ACM International Conference on Automated Software Engineering*, ASE '19, page 26–37. IEEE Press, 2020.

[24] Jian Guo, He He, Tong He, Leonard Lausen, Mu Li, Haibin Lin, Xingjian Shi, Chenguang Wang, Junyuan Xie, Sheng Zha, Aston Zhang, Hang Zhang, Zhi Zhang, Zhongyue Zhang, Shuai Zheng, and Yi Zhu. Gluoncv and gluonnlp: Deep learning in computer vision and natural language processing. *Journal of Machine Learning Research*, 21(23):1–7, 2020.

[25] Ge Han, Zheng Li, Peng Tang, Chengyu Hu, and Shanqing Guo. Fuzzgan: A generation-based fuzzing framework for testing deep neural networks. In *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, pages 1601–1608. IEEE, 2022.

[26] F. Ul Haq, D. Shin, and L. C. Briand. Many-objective reinforcement learning for online testing of dnn-enabled systems. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1814–1826, Los Alamitos, CA, USA, may 2023. IEEE Computer Society.

[27] Fitash Ul Haq, Donghwan Shin, Lionel C. Briand, Thomas Stifter, and Jun 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*, ISSTA 2021, page 91–102, New York, NY, USA, 2021. Association for Computing Machinery.

[28] Fitash Ul Haq, Donghwan Shin, Shiva Nejati, and Lionel Briand. Can offline testing of deep neural networks replace their online testing? a case study of automated driving systems. *Empirical Softw. Engg.*, 26(5), sep 2021.

[29] Fabrice Harel-Canada, Lingxiao Wang, Muhammad Ali Gulzar, Quanquan Gu, and Miryung Kim. Is neuron coverage a meaningful measure for testing deep neural networks? In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2020, page 851–862, New York, NY, USA, 2020. Association for Computing Machinery.

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[31] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

[32] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[33] Sungmin Kang, Robert Feldt, and Shin Yoo. Sinvad: Search-based image space navigation for dnn image classifier test input generation. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ICSEW'20, page 521–528, New York, NY, USA, 2020. Association for Computing Machinery.

[34] Zhong Li, Minxue Pan, Tian Zhang, and Xuandong Li. Testing dnn-based autonomous driving systems under critical environmental conditions. In *International Conference on Machine Learning*, pages 6471–6482. PMLR, 2021.

[35] Dingyu Liu, Yusheng Wang, Yonghoon Ji, Hiroshi Tsuchiya, Atsushi Yamashita, and Hajime Asama. Cyclegan-based realistic image dataset generation for forward-looking sonar. *Advanced Robotics*, 35(3-4):242–254, 2021.

[36] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 700–708, Red Hook, NY, USA, 2017. Curran Associates Inc.

[37] Yanyan Liu, Xiaotian Bai, Jiafei Wang, Guoning Li, Jin Li, and Zengming Lv. Image semantic segmentation approach based on deeplabv3 plus network with an attention mechanism. *Engineering Applications of Artificial Intelligence*, 127:107260, 2024.

[38] Microsoft. Airsim. https://github.com/microsoft/AirSim, 2024. Version 1.8.1.

[39] NVIDIA. pix2pixhd: High-resolution image synthesis with conditional gans. https://github.com/NVIDIA/pix2pixHD, 2017. 2023.

[40] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2642–2651. JMLR.org, 2017.

[41] Augustus Odena, Catherine Olsson, David Andersen, and Ian Goodfellow. TensorFuzz: Debugging neural networks with coverage-guided fuzzing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4901–4911. PMLR, 09–15 Jun 2019.

[42] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *proceedings of the 26th Symposium on Operating Systems Principles*, pages 1–18, 2017.

[43] Vincenzo Riccio, Nargiz Humbatova, Gunel Jahangirova, and Paolo Tonella. DeepMetis: Augmenting a Deep Learning Test Set to Increase its Mutation Score. *Proceedings - 2021 36th IEEE/ACM International Conference on Automated Software Engineering, ASE 2021*, pages 355–367, 2021.

[44] Vincenzo Riccio and Paolo Tonella. Model-based exploration of the frontier of behaviours for deep learning system testing. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, number 2, pages 876–888, New York, NY, USA, nov 2020. ACM.

[45] Vincenzo Riccio and Paolo Tonella. When and why test generators for deep learning produce invalid inputs: an empirical study. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1161–1173, 2023.

[46] Abhinav Sagar and RajKumar Soundrapandiyan. Semantic segmentation with multi scale spatial attention for self driving cars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2650–2656, 2021.

[47] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics: Results of the 11th International Conference*, pages 621–635. Springer, 2018.

[48] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2015.

[49] Star-Dundee. Planet and asteroid natural scene generation utility. https://pangu.software/, 2024. 2024.

[50] Andrea Stocco, Brian Pulfer, and Paolo 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.

[51] R Michael Swan, Deegan Atha, Henry A Leopold, Matthew Gildner, Stephanie Oij, Cindy Chiu, and Masahiro Ono. Ai4mars: A dataset for terrain-aware autonomous driving on mars. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1982–1991, 2021.

[52] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*, pages 303–314. IEEE, 2017.

[53] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. Defining and substantiating the terms scene, situation, and scenario for automated driving. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 982–988, 2015.

[54] András Vargha and Harold D. Delaney. A critique and improvement of the cl common language effect size statistics of mcgraw and wong. *Journal of Educational and Behavioral Statistics*, 25(2):101–132, 2000.

[55] Anton Vasiliuk, Daria Frolova, Mikhail Belyaev, and Boris Shirokikh. Limitations of out-of-distribution detection in 3d medical image segmentation. *Journal of Imaging*, 9(9), 2023.

[56] Jun Wang, Xiaolin Zhang, Tianhong Yan, and Aihong Tan. Dpnet: Dual-pyramid semantic segmentation network based on improved deeplabv3 plus. *Electronics*, 12(14):3161, 2023.

[57] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, Los Alamitos, CA, USA, jun 2018. IEEE Computer Society.

[58] Jiefei Wei and Qinggang Meng. Adversarialstyle: Gan based style guided verification framework for deep learning systems. In *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)*, volume 1, pages 641–648, 2020.

[59] Jiefei Wei and Qinggang Meng. Adversarialstyle: Gan based style guided verification framework for deep learning systems. In *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)*, volume 1, pages 641–648, 2020.

[60] Michael Weiss, André García Gómez, and Paolo Tonella. Generating and detecting true ambiguity: a forgotten danger in dnn supervision testing. *Empirical Softw. Engg.*, 28(6), November 2023.

[61] Trey Woodlief, Sebastian Elbaum, and Kevin Sullivan. Semantic image fuzzing of ai perception systems. In *Proceedings of the 44th International Conference on Software Engineering*, pages 1958–1969, 2022.

[62] Xiaofei Xie, Lei Ma, Felix Juefei-Xu, Minhui Xue, Hongxu Chen, Yang Liu, Jianjun Zhao, Bo Li, Jianxiong Yin, and Simon See. Deephunter: a coverage-guided fuzz testing framework for deep neural networks. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 146–157, 2019.

[63] Z. Yang, Y. Chai, D. Anguelov, Y. Zhou, P. Sun, D. Erhan, S. Rafferty, and H. Kretzschmar. Surfelgan: Synthesizing realistic sensor data for autonomous driving. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11115–11124, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society.

[64] Yuanyuan Yuan, Qi Pang, and Shuai Wang. Provably valid and diverse mutations of real-world media data for dnn testing. *IEEE Transactions on Software Engineering*, 50(5):1040–1064, 2024.

[65] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz 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*, pages 132–142, 2018.

[66] Pengcheng Zhang, Bin Ren, Hai Dong, and Qiyin Dai. Cagfuzz: coverage-guided adversarial generative fuzzing testing for image-based deep learning systems. *IEEE Transactions on Software Engineering*, 48(11):4630–4646, 2021.

[67] Xinhai Zhang, Jianbo Tao, Kaige Tan, Martin Törngren, José Manuel Gaspar Sánchez, Muhammad Rusyadi Ramli, Xin Tao, Magnus Gyllenhammar, Franz Wotawa, Naveen Mohan, Mihai Nica, and Hermann Felbinger. Finding Critical Scenarios for Automated Driving Systems: A Systematic Mapping Study. *IEEE Transactions on Software Engineering*, 49(3):991–1026, mar 2023.

[68] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017.

[69] Tahereh Zohdinasab, Vincenzo Riccio, Alessio Gambi, and Paolo Tonella. Deephyperion: exploring the feature space of deep learning-based systems through illumination search. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 79–90, 2021.

[70] Tahereh Zohdinasab, Vincenzo Riccio, Alessio Gambi, and Paolo Tonella. Efficient and effective feature space exploration for testing deep learning systems. *ACM Trans. Softw. Eng. Methodol.*, 32(2), mar 2023.

[71] Tahereh Zohdinasab, Vincenzo Riccio, and Paolo Tonella. An Empirical Study on Low- and High-Level Explanations of Deep Learning Misbehaviours. *International Symposium on Empirical Software Engineering and Measurement*, pages 1–11, 2023.