CARLA-GEAR: A Dataset Generator for a Systematic Evaluation of Adversarial Robustness of Deep Learning Vision Models

Federico Nesti, Giulio Rossolini, *Member, IEEE*, Gianluca D'Amico, Alessandro Biondi, *Member, IEEE*, and Giorgio Buttazzo, *Fellow, IEEE*

***Abstract*— Adversarial examples represent a serious threat for deep neural networks in several application domains and a huge amount of work has been produced to investigate them and mitigate their effects. Nevertheless, no much work has been devoted to the generation of datasets specifically designed to evaluate the adversarial robustness of neural models. This paper presents CARLA-GEAR, a tool for the automatic generation of photo-realistic synthetic datasets related to driving scenarios that can be used for a systematic evaluation of the adversarial robustness of neural models against physical adversarial patches, as well as for comparing the performance of different adversarial defense/detection methods. The tool is built on the CARLA simulator, using its Python API, and allows the generation of datasets for several vision tasks in the context of autonomous driving. The adversarial patches included in the generated datasets are attached to billboards or the back of a truck and are crafted by using state-of-the-art white-box attack strategies to maximize the prediction error of the model under test. Finally, the paper presents an experimental study to evaluate the performance of some defense methods against such attacks, showing how the datasets generated with CARLA-GEAR might be used in future work as a benchmark for adversarial defense in the real world. All the code and datasets used in this paper are available at http://carlagear.retis.santannapisa.it.**

***Index Terms*— Adversarial robustness, autonomous driving, CARLA simulator, adversarial defenses.**

I. INTRODUCTION

ADVERSARIAL examples represent a serious threat in several application domains, ranging from natural language processing to tabular data and computer vision [2]. Adversarial attacks can induce a convolutional neural network (CNN) to produce a wrong output by modifying the input with malicious perturbations that are typically imperceptible to humans. In the last years, an impressive amount of literature has been produced on this topic.

While the majority of the published papers are focused on digital adversarial examples (i.e., images whose digital representation is modified directly), only a few works tried to tackle the problem of *physical* adversarial attacks. The latter are special types of attacks carried out by means of physical objects, which are crafted to induce an adversarial behavior when captured by a camera whose images are processed by a CNN. They are even more subtle than those acting on the digital representation of images, as they can fool a CNN without directly accessing the vision system, but acting on the external environment only. This type of attacks represents a realistic threat for safety-critical systems relying on vision-based perception, such as self-driving cars, and a general countermeasure has yet to be found. Hence, an extensive evaluation of the adversarial robustness of CNNs is a crucial step for increasing the security of such systems.

In spite of the high relevance of such a step, no much attention was devoted in the literature to the development of datasets or benchmarks for a systematic evaluation of the adversarial robustness of CNNs against physical adversarial attacks, nor for assessing the performance of adversarial defense methods in a unified framework, in particular for autonomous driving scenarios [3], [4], [5]. This might be due to the practical difficulties posed by such an evaluation in the autonomous driving domain, which could result incomplete and/or dangerous. However, the rise of high-definition simulators is paving the way for fully-controllable, photo-realistic driving scenarios that allow for an extensive evaluation of potentially dangerous situations.

This lack in the literature motivates the quest for the design of datasets and benchmarks for the evaluation of the adversarial robustness of CNNs and the performance of defense methods in the physical world. To this end, this paper presents CARLA-GEAR (**Ge**neration of datasets for **A**dversarial **R**obustness evaluation with CARLA), a tool built on top of the Python API of the CARLA simulator [1], which allows constructing photo-realistic synthetic datasets for four vision tasks, namely semantic segmentation, 2D and 3D stereo object detection, and monocular depth estimation.

Manuscript received 29 January 2023; revised 22 November 2023 and 2 May 2024; accepted 3 June 2024. This work was supported in part by the Project Security and Rights in the Cyberspace (SERICS) under the Ministero dell'Università e della Ricerca (MUR) National Recovery and Resilience Plan funded by European Union-NextGenerationEU under Grant PE00000014. The Associate Editor for this article was Q. Ye. *(Federico Nesti and Giulio Rossolini contributed equally to this work.) (Corresponding author: Giulio Rossolini.)*

The authors are with the Department of Excellence in Robotics and AI, Scuola Superiore Sant'Anna, 56127 Pisa, Italy (e-mail: giulio.rossolini@santannapisa.it).

This article has supplementary downloadable material available at https://doi.org/10.1109/TITS.2024.3412432, provided by the authors.

Digital Object Identifier 10.1109/TITS.2024.3412432

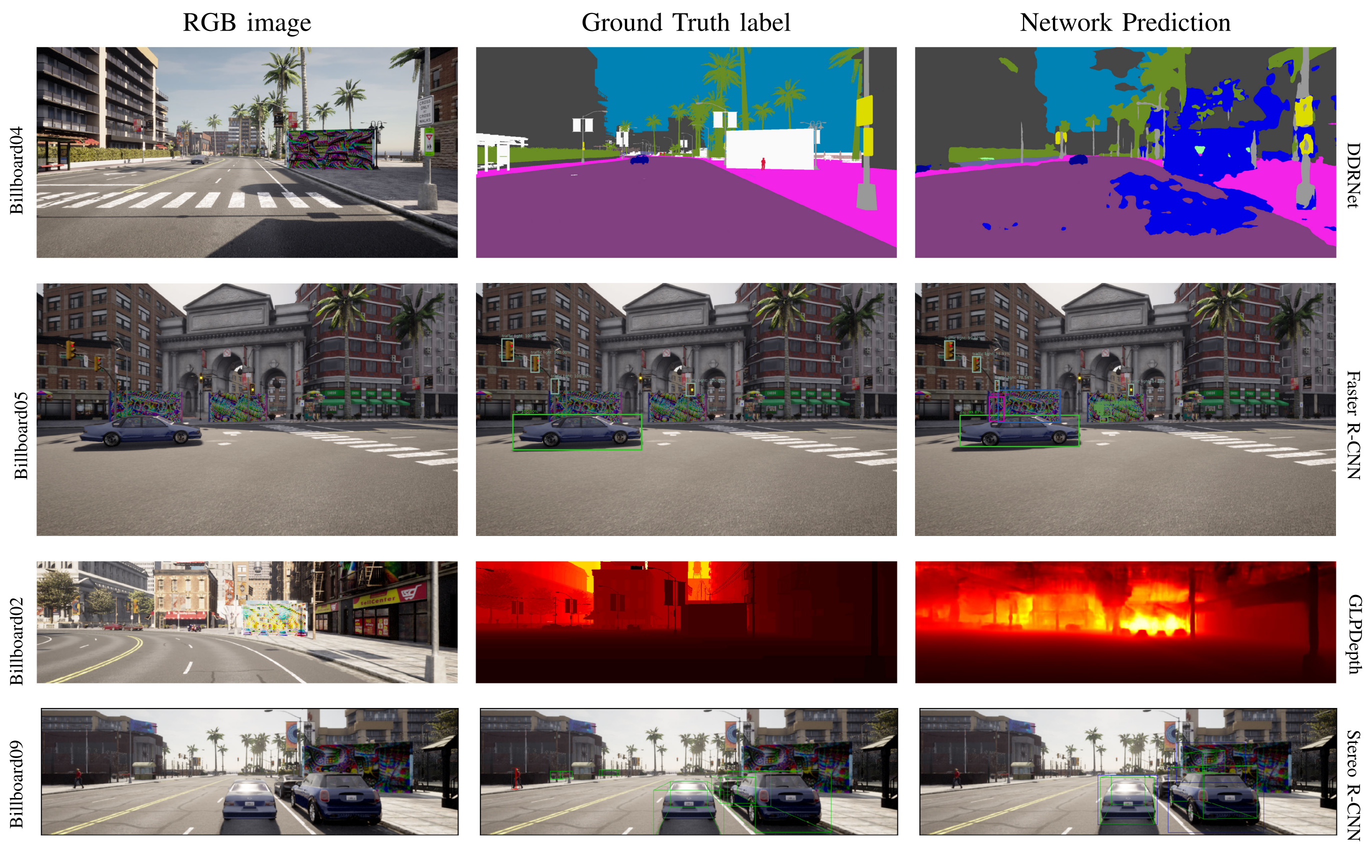


Fig. 1. Examples of different attack scenarios and tasks. Best viewed in digital, zooming in. DDRNet for semantic segmentation on the billboard04 scenario, Faster R-CNN for 2d object detection on billboard05, GLPDepth for monocular depth estimation on billboard02, and Stereo R-CNN for stereo 3d object detection on billboard09. First column shows the RGB image, second column shows the ground truth generated by CARLA, and the third column shows the prediction of the corresponding CNN.

Figure 1 illustrates some representative driving scenarios considered in this work. Given a urban scenario where an adversarial patch might be placed (e.g., on a billboard or a truck) and, optionally, an adversarial patch, CARLA-GEAR attaches the patch on the selected surface and iteratively places the vehicle and the attached camera around the scene, collecting high-definition RGB images, the ground-truth labels, and additional information on the camera intrinsic and extrinsic matrices, as well as the position of the billboard in the scene.

If the patch is not available, it can be generated by first collecting a dataset around a certain billboard with no patch attached, and then running the optimization algorithm proposed in [6]. The resulting patched dataset can be used to evaluate the performance of different defense mechanisms, or to evaluate the adversarial robustness of a target CNN. Several datasets, including different attack scenarios, can be collected and used for a more systematic evaluation. To this end, together with the generation tool, this paper presents an extensive comparison of a selection of adversarial defense and detection methods on several datasets produced with CARLA-GEAR. The experimental results indicate that defense mechanisms can produce different outcomes when tested in simulated driving scenarios, implying potential variability in complex real-world environments. This observation underscores a critical aspect of evaluating defense techniques, emphasizing the need for their thorough assessment in scenarios that closely resemble the intended target environments.

CARLA-GEAR is licensed under MIT License and is hence free and open to use. Ethically, this work raises security and safety concerns about the deployment of safety-critical systems based on vision. We believe that the extensive customization options and photo-realism offered by CARLA-GEAR not only provide a systematic tool for evaluation but also uncover essential insights regarding the correctness of testing the robustness of neural networks and defense strategies against physical adversarial attacks in driving scenarios. In summary, the contributions of this work are:

- It presents CARLA-GEAR, the first tool for the automatic generation of datasets for (i) evaluating the adversarial robustness of CNNs in the physical world and (ii) comparing the performance of different adversarial defense/detection methods in the context of autonomous driving for four different computer vision tasks: semantic segmentation, monocular depth estimation, 2d object detection, stereo 3d object detection. The tool is available at https://github.com/retis-ai/CARLA-GeAR.
- It presents an extensive comparison of a selection of adversarial defense and detection methods on a set of datasets created with CARLA-GEAR, suggesting that such a tool can be used to generate benchmarks for a standardized comparison of defense methods. The datasets

and corresponding patch generation and evaluation code are available at http://carlagear.retis.santannapisa.it.

The remainder of this paper is organized as follows: Section II reviews the related literature, Section III presents CARLA-GEAR and the generation of adversarial patches, Section IV reports the experimental results, and Section V discusses the limitations of the approach, future work, and states the conclusions.

II. Related Work

Since the discovery of the phenomenon of adversarial examples [7] and [8], a huge amount of work has been produced around them.[1] Adversarial attacks can be coarsely divided in two categories: digital attacks, for which the attacker has complete control of the digital representation of the image, and physical-attacks [9], mainly realized by patches [10] that can be printed and placed in the real world.

Digital attacks are typically composed of image-specific, norm-limited, full-image perturbations and received most of the attention by researchers because of their effectiveness. However, the scope of these attacks is limited to those systems that allow the upload of a user-defined image. On the other hand, physical adversarial attacks require more complex optimization to make the perturbation robust to viewpoint and illumination changes [11]. Furthermore, they are image-agnostic perturbations [12] that can be printed and placed in the physical world to fool CNNs operating in the wild. This fact might be a serious concern for safety-critical systems relying on CNN-based perception such as autonomous driving. Therefore, a crucial step for a safe and secure deployment of such systems in the real world is an extensive evaluation of their robustness against this kind of attacks.

A. Physical Attacks and Defenses

Inspired by the Expectation Over Transformation (EOT) algorithm [11], in the last five years several works have spread the success of physical attacks by crafting different forms of real-world adversarial objects: patches [10], T-Shirts [13], 3D objects [14], etc. Among them, adversarial patches have shown to be quite effective in real-world environments (e.g., autonomous driving scenarios) in multiple deep learning tasks, such as image classification [10], [15], [16], object detection [17], [18], [19], [20], optical flow [21], LiDAR object detection [22], depth estimation [23], and semantic segmentation [24].

In parallel with the development of attack strategies, several defense mechanisms have been proposed to detect or mitigate the adversarial effects of physically-realizable objects. Some works [25], [26], [27] leverage adversarial training or gradient-masking strategies to directly enhance the physical adversarial robustness of the original model. However, such approaches are computationally expensive, especially in a real-world context. This problem stimulated the investigation of alternative defense algorithms, often less training-expensive and less model-invasive. Defense algorithms against real-world objects [24], [28], [29], [30], [31], [32], [33], [34], [35] aim at detecting those images corrupted with adversarial patches or masking the corresponding adversarial areas to mitigate the adversarial effect in the model outcome.

B. Datasets for the Real-World Robustness

Despite the large success of adversarial attacks, only a few works proposed datasets and benchmarks for a systematic evaluation of the robustness of models. Most of them are focused on digital adversarial examples [36], [37], [38], [39] and none of them considers the autonomous driving context. In [35], [40], the authors systematically evaluate the effectiveness of real-world patches. Other works explore other kinds of attacks against assisted [41] or autonomous driving systems [42]. Another work close to CARLA-GEAR is DeepBillboard [43], which generates adversarial patches to be attached on billboards. However, DeepBillboard attacks end-to-end autonomous driving models, whereas CARLA-GEAR considers several perception tasks. Furthermore, being built on a simulator, CARLA-GEAR allows for fully-controllable scenarios, which is a crucial feature for evaluating of the robustness of a system.

To the best of our records, APRICOT [44] is the only publicly available dataset that includes physical-world adversarial patches. However, it can only be used to test 2D object detection models. Furthermore, the dataset does not include non-adversarial images and the patches are not always effective due to patch bending or extreme view angles. Conversely, CARLA-GEAR is available for four different vision tasks for autonomous driving and, being built on the CARLA Python API, it enables a full control of the tested scenarios.

The adversarial patches included in the static dataset provided by CARLA-GEAR were generated using the adversarial pipeline proposed in [6] to perform untargeted white-box attacks. Among all the adversarial defenses and detection methods, we restricted the analysis to those methods designed to detect and mask adversarial patches that are easily extendable to the physical world case and to different tasks. In particular, we chose Z-Mask [24], FPDA [35], LGS [34], and HyperNeuron [33]. Additional details are discussed in Section IV and better addressed in the supplementary material.

III. Proposed Dataset Generation

The proposed pipeline is designed to generate datasets for four different computer vision tasks involved in autonomous driving perception. Each dataset is intended to evaluate the adversarial robustness of a CNN and/or the performance of a defense method in a situation in which a physical adversarial attack might be present (chosen from a library of attack situations). For each situation, the ego vehicle with its cameras is iteratively spawned at different distances from the attackable surface, together with the random non-playing characters (NPCs, i.e., vehicles and pedestrians). The dataset is composed of the so-collected RGB images and the corresponding ground-truth annotations, which are different for each task. The supplementary material includes in-depth explanations of the generation and collection process.

[1] https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html

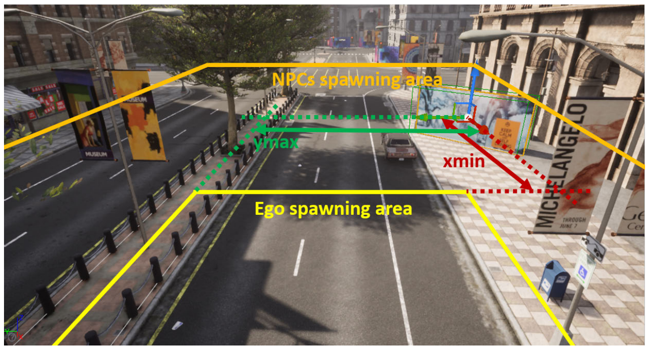


Fig. 2. An example of spawn area for ego vehicle and NPCs. All the distances are relative to the billboard.

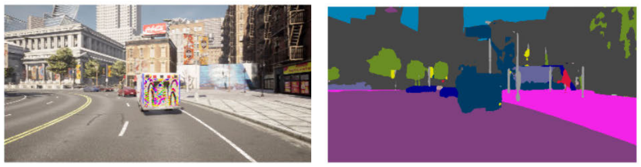


Fig. 3. Illustration of an adversarial patch placed on the back of a truck. The output prediction is obtained with the DDRNet [45] model.

A. Attack Situations

This work considers real-world adversarial attacks based on patches. The tool considers two different types of situations: (i) a patch (or two in the case of multi-patch attacks) on a billboard on the side of the road, and (ii) a patch on the back of a truck placed in front of the camera. Each billboard situation is specified in a `yml` file as the fixed position of the billboard in the map. The same file specifies the spawn positioning limits of the ego vehicle and the NPCs relative to the billboard. The spawning is then randomized within these limits, as illustrated in Figure 2. In this way, it is possible to generate different views of the same potentially dangerous scene.

While the billboards have fixed positions, the truck is randomly spawned in the map, and the ego vehicle is spawned behind it at different randomized distances (see Figure 3).

B. Configuration Files

Three configuration files are required: (i) `collection_config.yml`, which is the main collection configuration file, (ii) the `billboard_config.yml` file, which includes information about the billboard positioning and the limits for the spawning area of the ego and NPC vehicles, and (iii) the `simulation_config.py` file, which sets the seed and other simulation parameters (number of NPCs in a scene and other Unreal utilities). The role of these configuration files is explained in detail in the supplementary material.

The pipeline described in this section generates single dataset splits (train, validation, test, and so on) starting from the main configuration file `collection_config.yml`, that specifies:

- The target task (one between semantic segmentation, 2D object detection, stereo 3D object detection, monocular depth estimation). This directly specifies the folder structure and the ground truth annotation types required;
- The CARLA town. Any CARLA town would work, but the one used throughout this paper is Town10HD, since it is the town with the most photo-realistic meshes in CARLA;
- The dataset root folder and split;
- The desired scene. As explained in Section III-A, this configures the adversarial surface positioning, the ego vehicle, and NPCs spawning.
- The desired patch to be uploaded and additional info on it. The patch path, if specified, is used to render the patch on the surface selected. If it is not specified, no patch is rendered. This possibility is detailed in Section III-E.

The aforementioned `billboard_config.yml` and `simulation_config.py` configuration files are then read to define the attack situation and simulation settings.

C. Data Generation Algorithm

Figure 4 shows the data generation and collection pipeline, which is discussed next.

Read config files is a set of parsers that extract information from the configuration files presented in Section III-B.

Setup simulation and data collection sets up the client-server communication with CARLA through its Python API and writes a few necessary settings. It then loads the specified town and spawns the selected billboards. It also sets up the camera types required for the specific task, the seed for repeatability, and generates the folder tree to properly store images and annotations.

Data Generation and Collection is the core of the pipeline illustrated in Figure 4. The dataset is constructed by iterating three steps: (i) cleanup of any additional vehicle/pedestrian, (ii) spawn of the ego vehicle, its sensors, and randomized NPCs following the spawning limits of the specific situation, (iii) save the RGB image, the ground truth (this task-specific operation is detailed in Section III-D), the billboard, and the camera poses. Additional details are provided in the supplementary material.

D. Dataset Structure and Annotations

RGB images are always saved as uint8 with different resolutions, while each task has different ground-truth annotations and folder structure: semantic segmentation datasets follow the CityScapes [46] format, 2D object detection uses the COCO format [47], stereo 3D object detection the Kitti Stereo Object Detection format [48], and monocular depth estimation the Kitti Depth format. This choice ensures that the same original metrics, evaluation procedures and deep learning framework data loaders can be used to evaluate these custom datasets.

CARLA has special semantic segmentation and depth camera sensors that allow immediate ground-truth computation. It can also compute all the 3D bounding boxes in the simulated world. However, the 2D and 3D bounding boxes that are visible in each scene must be computed heuristically (details are reported in the supplementary material).

Additionally, CARLA-GEAR collects all the billboard positions together with the camera extrinsic and intrisic matrices. This allows knowing the precise pose of the attackable

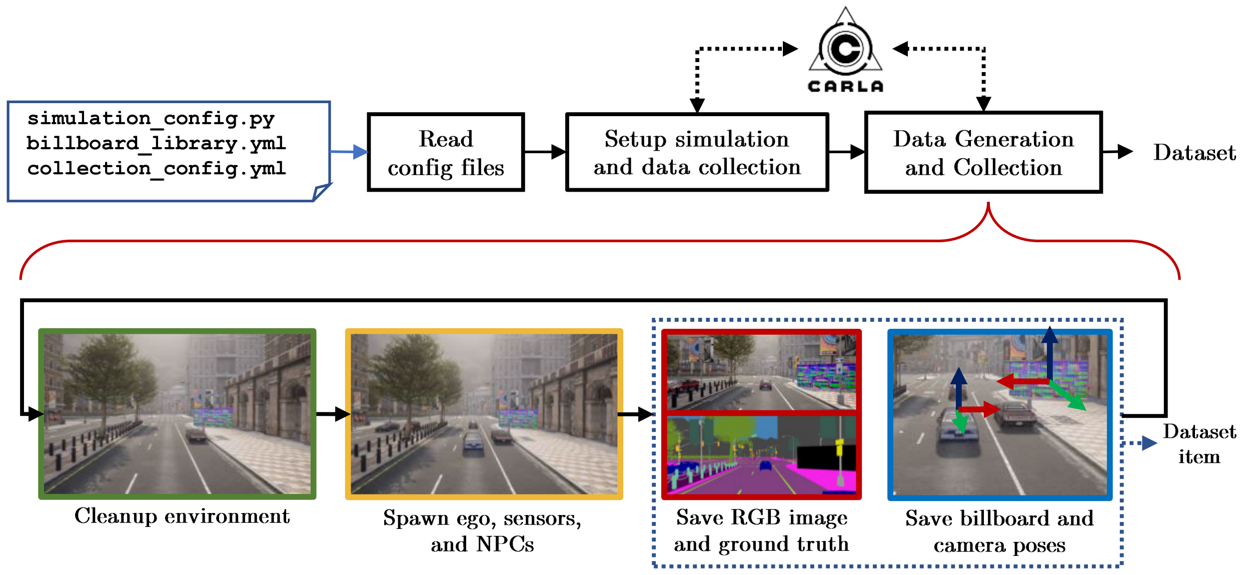

simulation_config.py
billboard_library.yml
collection_config.yml
Read config files
Setup simulation and data collection
CARLA
Data Generation and Collection
Dataset
Cleanup environment
Spawn ego, sensors, and NPCs
Save RGB image and ground truth
Save billboard and camera poses
Dataset item

Fig. 4. Complete generation flow (top) and detail of the data generation and collection phase.

surface in the scene, which is crucial to make an accurate digital patch placement (see details in Section III-E) and obtain the ground truth for evaluating the accuracy of masking defense methods against adversarial patches.

E. Patch Generation

As described in Section III-A, the `collection_config.yml` file specifies the path of the patch that must be uploaded and rendered on the selected billboard. If the file path is not defined, the tool does not render anything on the billboard. This is useful to have an additional test split to check the performance of a CNN or a defense method in a non-adversarial case. Furthermore, these non-adversarial dataset splits can be used to train adversarial patches for that specific situation. We follow the *scene-specific* attack method proposed in [6], which requires camera and billboard poses to accurately reproject the digital patch to maintain differentiability and the possibility to perform white-box attacks. Implementation details are reported in Section IV-A.

IV. Testing the Tool

This section presents an instance of how the CARLA-GEAR toolbox might be used to benchmark the robustness of CNNs and the performance of a set of defense and detection algorithms. In Section IV-B, preliminary tests were performed to set the simulation configuration used for the generation of datasets. Furthermore, in Section IV-C the datasets generated are used to compare a selection of state-of-the-art defenses and detection methods for each task. The experimental setup is briefly listed in Section IV-A and better detailed in the supplementary material, together with additional illustrations.

A. Experimental Setup

The CARLA simulator version 0.9.13 was used (Unreal-Engine 4.26 on Ubuntu 18.04) on a machine equipped with an Intel i7-4790K CPU @ 4.00GHz $\times$ 8 and an NVidia GTX 1080Ti GPU. The adversarial patch optimizations were performed on an NVidia Tesla A100 GPU using PyTorch.

The CNNs used in this paper were DDRNet23Slim [45] and BiSeNetXception39 [49] for semantic segmentation, Faster R-CNN [50] and RetinaNet [51] for 2D object detection, GLPDepth [52] and AdaBins [53] for monocular depth estimation and Stereo R-CNN [54] for stereo 3D object detection. Please note that we focused the analysis on CNNs as they were most prominently addressed by known real-world attacks [13], [35] and defenses. However, other architectures, such as visual transformers [55], can also be tested with CARLA-GEAR using the proposed framework and the presented pipeline.

The defenses used for comparison are FPDA [35], Z-mask [24] and HyperNeuron [33] for adversarial detection, and Z-mask [24] and LGS [34] for adversarial defense.

Local Gradient Smoothing (LGS) aims at filtering out high-frequency areas in the image, based on the fact that digital adversarial patches mainly rely on high-frequency information. The other approaches (Z-Mask, FPDA, and HyperNeuron) are based on the assumption that adversarial patches induce anomalous over-activations in specific network layers. To this end, they identify internal over-activated areas through statistical analysis and leverage them to detect and/or mask the patches. Additional details about their learning procedure and selected layers are available in the supplementary materials.

The comparison metrics used in this paper are (i) mIoU for semantic segmentation, (ii) the COCO mAP for 2D object detection, (iii) root mean square error (RMSE) in meters for monocular depth estimation, and (iv) kitti AP for "moderate" label difficulty for stereo 3D object detection. Please note that additional metrics are reported in the supplementary materials, but, nonetheless, these metrics do not entirely reflect the effectiveness of a real-world attack or defense: the labels produced by CARLA are often too fine-grained and occlusions might occur. Hence, to properly reflect the performance of

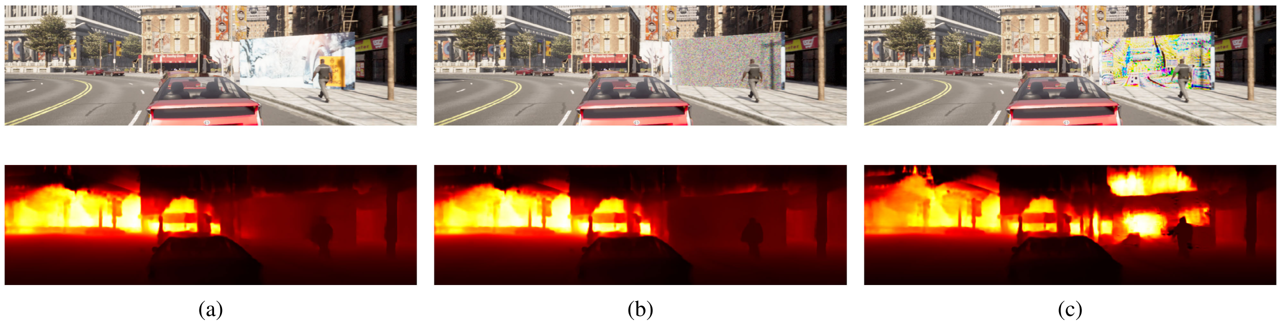


Fig. 5. Comparison of RGB images (top) and network prediction (AdaBins, bottom) for a sample of the dataset billboard02: (a) test_nopatch, (b) test_random, (c) test_adabins.

the analyzed scenarios there is a strong need for specialized metrics designed for this purpose. This is left as a future work in perspective of the construction of a benchmark— a discussion of this issue is presented in Section V. The performance of adversarial detection algorithms is evaluated with the area under the receiver operating characteristic (AUROC). Each ROC is computed on two "twin" datasets, one including the adversarial patches and the other containing the same images, but without any patch.

1) Life-Cycle of a Patch: To evaluate the effects of real-world adversarial patch attacks in CARLA it is necessary to follow a few steps. Given a certain task and an attack scenario, it is possible to collect a training dataset split to digitally optimize the patch to attack a specific CNN using a *scene-specific* optimization algorithm described in [6]. The same training set can be used to craft patches for different networks but for the same task. Once the patch is optimized, it can be imported in CARLA and applied to the billboard to generate the test set. In this way, it is rendered realistically as the other objects in the scene and it constitutes a "virtual" real-world adversarial object. Different patches (for different networks) are used to generate different test sets. However, each test set shares the same seed. This allows generating datasets with exactly the same images, except for the selected patch, hence providing a more fair comparison. Furthermore, each adversarial patch is tested against a control test set that includes a random patch or no patch.

The datasets used in Section IV-C include 10 attack situations, namely 9 billboard-based attacks (billboard01 to billboard09, of which billboard05 to billboard07 are double billboard for double patch attacks) and 1 truck-based attack.

Each of these dataset includes images collected around the attack surface, as explained in Section III, arranged in the following splits:

- A train split (100 images) with no patch attached. This can be used to optimize patches for the specific situation.
- A val split (50 images) with no patch attached. It can be used as a validation split during the patch optimization.
- A test_net split (50 images), which includes a patch crafted specifically for the model indicated by "net" (e.g., ddrnet, bisenet, adabins, and so on).
- A test_random split (50 images), which includes a random patch.
- A test_nopatch split (50 images), which includes no patch on the attack surface.

Please note that each test split has the same seed. Therefore, the only difference between the three test split images described above is the patch applied on top of the attack surface, as showed in Figure 5. This is useful for a fully controllable and fair comparison of the adversarial effect of the patch, as well as the evaluation of the adversarial robustness of different defense methods and models.

Furthermore, for the conducted tests we provide a `no_billboard` dataset, which includes only a train split (100 images) with no billboard spawned. The ego vehicle and its camera are moved randomly around the city. This dataset can be used as a reference for the distribution of the scenes produced by CARLA. In fact, it was used to update the batch normalization parameters of DDRNet, BiseNet and AdaBins to improve the performance and handle an unavoidable domain shift between CARLA scenes and realistic images from Cityscapes. It was also used to train the defense methods that required a clean sample distribution.

2) On the Flexibility of CARLA-GEAR*:* The proposed framework is aimed at systematically evaluating the adversarial robustness in several attack situations. However, for computational and time limits, the experimental results presented in this section are just a subset of those that can be obtained with the proposed tool. More specifically, we tested untargeted attacks that produce highly effective patches based on robust and flashy adversarial input patterns. However, different threat models might be considered during the patch generation phase to obtain more inconspicuous patterns (such as the one in [42]) or different effects. These include (i) defense-aware attacks, which are omitted since the paper focuses on the dataset generation toolbox rather than the evaluation itself, (ii) patches with different shapes (that can be easily changed, as illustrated in the supplementary material), and also (iii) targeted real-world adversarial attacks, which are left to be investigated in future work.

Even though these aspects are not considered in this paper for computational limits, CARLA-GEARis flexible enough to seamlessly cope with them to perform a thorough assessment of the robustness of a CNN or a defense method.

B. Ablation Studies

This section reports the results of ablation studies performed in the early stages of the experimentation to set the parameters used for the data generation. The objective was to find a good trade-off between the CNN performance and the attack

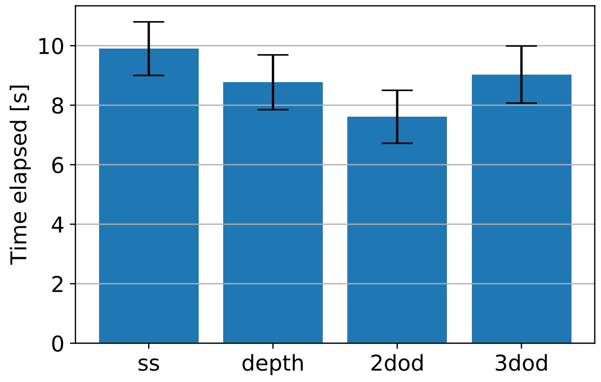


Fig. 6. Time required for the generation of a dataset item for each task. The error bar represents the standard deviation.

effectiveness while maintaining reasonable computation times. Inspired by a previous work [35], we decided to use the same billboard configuration that allows the application of a 3.7m×7.4m patch with 150 × 300 pixels. The CNNs used for these studies are Faster R-CNN for 2D object detection, GLPDepth for depth estimation, and DDRNet for semantic segmentation.

1) Computation Time: The creation of such datasets is a time-consuming task. In this section we show the average time required to generate and save an RGB image and ground-truth annotation for each task. From the results reported in Figure 6, it is clear that the most time-consuming task is semantic segmentation. This is attributed to the fact that segmented images (and ground-truth labels) have the highest resolution, following the Cityscapes format. Then, stereo 3D object detection and depth estimation follow, since the first saves pairs of RGB images and the latter saves an RGB and a depth image. 2D object detection is the least expensive task for generation, since it has to save the RGB image only, while the annotation is a json file. The timing data have been collected during the generation of the same 50-sample test set, changing the seed ten times. Hence, the generation of a 50-sample dataset takes roughly eight minutes on average.

2) Number of Samples for the Optimization: An important parameter that must be set is the number of samples required to run the optimizations properly. Too few samples might not be enough to obtain effective attacks, while too many samples slow the generation and optimization processes down. Figure 7a shows the attack performance with respect to the baseline CNN (obtained with a random patch) as a function of the number of samples for each task. The results were averaged over ten different optimizations, changing the seed each time. Since optimizing a patch with 100 samples leads to better results overall, we set the training set dimension to 100 images in the generation stage.

3) Weather Conditions: The CARLA simulator allows full control of the weather parameters, ranging from the time of day (i.e., the elevation angle of the sun) to the amount of rain and wetness of the road, from the fog density to the Rayleigh scattering. Given the large number of different parameters to evaluate, which would lead to a combinatorial explosion in the number of settings, we decided to limit the study to the weather presets available in CARLA, ignoring those that lie outside the distribution of the datasets used to train the CNNs under test (i.e., no heavy rain, puddles, night, or fog). The tested weather presets were `CloudyNoon`, `ClearNoon`, `CloudySunset`, `ClearSunset`, `SoftRainNoon`. Nevertheless, note that in CARLA-GEAR the weather parameters are fully configurable.

We decided to restrict the search to three attack situations with different billboard orientations (and, hence, camera orientation) with respect to the sunlight position: billboard02 has the sun behind (shining directly on the billboard), billboard04 has the sun on the side, and billboard09 has the sun in front (behind the billboard - see Figure 1). The results in Figure 7b show that, for each task tested, the baseline performance (i.e., applying a random real-world patch) when using presets `CloudySunset` and `ClearSunset` is almost always slightly worse. The attack performance resulted not to be particularly influenced by the weather conditions. Hence, we used the `ClearNoon` weather for the following experiments, since it performs slightly better overall. We reported additional illustrations of the adversarial effect of patches under different weather conditions in the supplementary material.

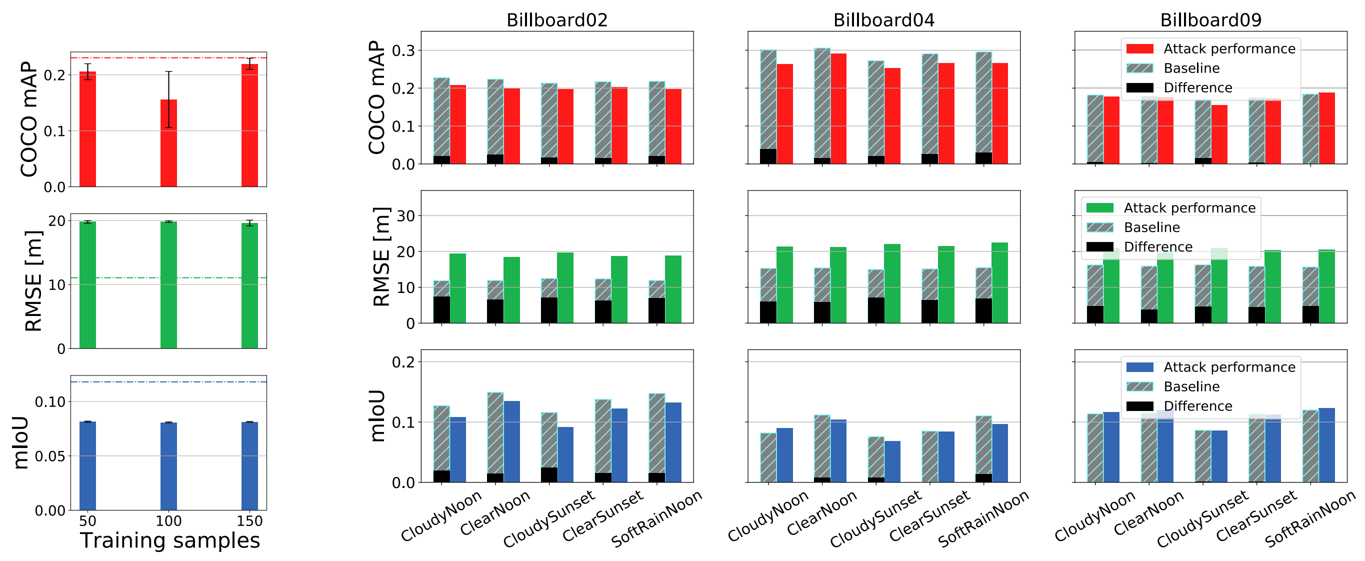
C. Defense Comparison

This section presents an extensive comparison of different defense methods when applied to the dataset generated by CARLA-GEAR for each of the four computer vision tasks considered in this work.

1) Adversarial Detection: Table I compares the performance of Z-Mask, FPDA, and HyperNeuron in terms of AUROC. All the methods present good performance for most of the networks and scenarios, with Z-Mask being the most consistent detection method overall. Please note that, in some scenarios (*bilboard08, bilboard09 and truck*), all the tested algorithms surprisingly get to low detection performance. Such values clearly differ from the detection performance expected by the tested algorithms, showing that such scenarios depict critical situations that question the reliability of the addressed defenses.

2) Adversarial Masking: Table II reports the performance of each defense method considered for each task. Notably, there are a few attack situations that do not induce a large adversarial effect with respect to the addressed metrics (e.g., Billboard01, 02, 03, 05, 09, and Truck on DDRNet), where the defense is actually harming the performance of the network. Nevertheless, it is worth noting how the mAP metric for 2D object detection indicates a consistent low adversarial effect and bad defense performance, while in reality the effect of the adversarial patch is always present (see Figure 1 for an example) and the defense is masking the patch with fair accuracy. As it is also remarked in the final section, this observation highlights the need for a more accurate metric to evaluate the effect of these patches (and related defenses) and highlights one of the limitations to the realization of a proper real-world robustness benchmark. Please consult the supplementary material for the results with additional metrics, such as the IoU of the patch mask, and further illustrations.

Regarding the achieved results, also these experiments report different defense performance among all the addressed



(a) (b)

Fig. 7. (a) Performance of the attack for each task as a function of the number of samples used for the optimization of the image-agnostic patch. The experiment was performed on billboard02 ten times, changing the seed for each experiment. The dash-dotted line represents the baseline performance (random patch) whereas the error bar represents the standard deviation. (b) Effect of the different weather presets of CARLA on the baseline performance (random patch) against the attack effectiveness for each task. The experiments were performed for three different attack situations with different camera-sunlight relative orientations.

TABLE I

Comparison of Detection Performance (Measured in AUROC) Across Various Defense Strategies (Z-Mask (ZM), FPDA, and HyperNeuron (HN)), Computed for Each Attack Situation, Model, and Task. The Evaluated Tasks Include Semantic Segmentation (SS), 2D and 3D Object Detection (2D/3D OD), and Depth Estimation

		SS		2DOD		Depth		3DOD
	Def	DDRNet	BiseNet	FRCNN	RetinaNet	AdaBins	GLPDepth	Stereo RCNN
Billboard01	ZM	0.99	0.95	0.50	0.49	0.27	0.00	0.82
	FPDA	1.00	0.97	0.53	0.49	0.20	0.49	0.50
	HN	1.00	1.00	0.52	0.53	0.09	0.12	0.89
Billboard02	ZM	1.00	1.00	1.00	1.00	0.89	0.26	0.92
	FPDA	1.00	1.00	0.97	0.66	0.90	0.00	0.50
	HN	1.00	1.00	0.97	0.78	0.98	0.51	0.96
Billboard03	ZM	0.98	0.90	0.99	0.99	0.99	0.33	0.74
	FPDA	0.86	0.92	0.64	0.85	0.89	0.43	0.50
	HN	0.95	0.87	0.88	0.82	0.97	0.23	0.75
Billboard04	ZM	1.00	1.00	0.95	0.96	0.97	0.39	0.97
	FPDA	0.99	1.00	0.58	0.56	0.92	0.49	0.50
	HN	1.00	1.00	0.70	0.66	0.97	0.40	0.95
Billboard05	ZM	1.00	1.00	0.86	0.90	0.31	0.05	0.92
	FPDA	0.99	1.00	0.62	0.64	0.30	0.50	0.78
	HN	0.99	1.00	0.74	0.72	0.27	0.39	0.95
Billboard06	ZM	1.00	1.00	1.00	1.00	0.96	0.59	0.98
	FPDA	1.00	1.00	0.87	0.61	0.86	0.07	0.50
	HN	1.00	1.00	0.85	0.85	0.88	0.94	0.99
Billboard07	ZM	1.00	0.60	1.00	1.00	1.00	0.61	0.93
	FPDA	1.00	1.00	0.89	0.60	0.97	0.49	0.50
	HN	1.00	1.00	0.96	0.73	0.98	0.27	0.92
Billboard08	ZM	1.00	0.97	0.66	0.67	0.29	0.00	0.98
	FPDA	0.87	1.00	0.58	0.55	0.23	0.50	0.50
	HN	0.93	1.00	0.59	0.67	0.21	0.10	0.99
Billboard09	ZM	0.68	0.65	0.84	0.86	0.91	0.48	0.59
	FPDA	0.62	1.00	0.58	0.46	0.90	0.51	0.50
	HN	0.61	1.00	0.76	0.51	0.99	0.51	0.65
Truck	ZM	0.87	0.68	0.58	0.56	0.60	0.42	0.63
	FPDA	0.87	0.72	0.52	0.54	0.52	0.46	0.50
	HN	0.86	0.77	0.53	0.52	0.58	0.43	0.62

scenarios, thus helping understand potential critical situations uncovered during the design of the defense. Figure 8 provides some illustrations of the masking effect of Z-Mask among generated images for semantic segmentation and object detection.

To conclude the analysis, despite the issues and open challenges highlighted above and better discussed in Section V, the obtained experimental results provide relevant quantitative information about the potential power of CARLA-GEAR as a generator of adversarial situations for evaluating the behavior of neural models and/or the defense under test. In fact, leveraging simulated adversarial environments helps understand critical scenarios, which are likely uncovered during the design and testing of many defense algorithms, as well as better expose robustness situations that can undermine the nominal DNNs performance.

V. Concluding Remarks

A. Threats to Validity and Future Challenges

At the current stage, CARLA-GEAR presents a few limitations. The following is a list of open problems linked to the evolution of CARLA-GEAR:

1) Domain shift on CARLA: Although the meshes used in Town10HD of CARLA are photo-realistic, there is a clear domain shift between the CARLA-generated images and common real-world datasets used for scene-understanding (e.g., COCO, CityScapes, Kitti). This raises concerns about the generalizability of the evaluation conducted with our tool to real-world scenarios. Nevertheless, real-world evaluations are challenging and expensive to achieve, as pointed out by related studies (e.g., [35] and [42]), thus remarking the necessity of using simulators. The need for clarification on the proper understanding of result generalization opens further avenues for investigation. To address this, future efforts might focus on two areas: firstly, understanding the level of transferability in light of recent advancements in photorealism within driving simulators; secondly, developing new optimization attack

TABLE II

PERFORMANCE OF Z-MASK AND LGS FOR EACH ATTACK SITUATION, FOR EACH TASK. WHITE COLUMNS SHOWS THE RESULTS IN THE CASE WITHOUT PATCH, WHEREAS GREYED COLUMNS SHOWS THE PERFORMANCE IN THE ADVERSARIAL CASE. ALL THE CELLS REPORT THE RELATIVE PERFORMANCE WITH RESPECT TO THE UNDEFENDED, NON-ADVERSARIAL CASE (IN BOLD). THE VALUE IS COLORED IN RED [GREEN] IF THE DEFENSE IS WORSENING [IMPROVING] THE PERFORMANCE OF THE MODEL, OR BLUE IF THERE IS NO EFFECT

		SS (mIOU)				2DOD (mAP)				Depth (RMSE)				3DOD (mAP)	
	Def	DDRNet		BiseNet		FRCNN		RetinaNet		AdaBins		GLPDepth		Stereo RCNN	
Billboard01	-	**33.04**	0.65	**28.35**	-10.20	**19.89**	-0.59	**20.12**	-2.66	**15.51**	6.00	**13.52**	3.22	**51.43**	-2.84
	ZM	0.00	-2.31	-0.15	-5.32	0.02	-0.49	-0.25	-2.95	-0.36	1.72	-0.10	3.22	-0.45	-0.86
	LGS	-0.07	0.45	0.11	-8.63	0.02	-1.38	-0.25	-4.41	-0.36	5.94	-0.10	3.12	-0.45	-3.09
Billboard02	-	**34.11**	-1.74	**29.66**	-4.32	**19.66**	-0.93	**19.65**	-0.45	**16.04**	4.93	**12.33**	5.61	**38.29**	3.70
	ZM	-0.04	-1.76	-0.21	-2.35	0.06	-1.32	-0.17	-1.12	-0.02	0.07	-0.03	3.04	-0.41	0.85
	LGS	-0.20	-3.51	0.07	-5.09	0.06	-2.90	-0.17	-2.63	-0.02	3.54	-0.03	4.51	-0.41	6.23
Billboard03	-	**34.39**	-0.60	**28.88**	-2.81	**19.30**	-0.89	**18.92**	-0.69	**14.23**	3.22	**12.05**	3.33	**77.54**	-11.56
	ZM	0.00	-0.60	0.00	-2.81	0.00	-0.77	0.00	-0.70	-0.00	1.03	0.00	3.33	-8.86	-16.59
	LGS	1.60	1.01	1.92	-4.09	0.00	-2.27	0.00	-1.60	-0.00	4.24	0.00	3.11	-8.86	-27.77
Billboard04	-	**34.22**	-6.53	**24.71**	-6.40	**25.01**	1.80	**27.83**	0.99	**18.62**	6.11	**15.73**	5.32	**47.75**	0.86
	ZM	0.00	-5.20	0.00	-3.01	0.00	1.79	0.00	0.92	-0.01	0.71	0.00	5.32	-7.97	0.03
	LGS	-0.14	-8.30	-0.71	-6.43	0.00	0.17	0.00	-2.32	-0.01	5.80	0.00	5.23	-7.97	1.11
Billboard05	-	**31.01**	-1.01	**25.10**	-5.06	**22.45**	-0.21	**22.17**	-0.53	**11.75**	10.41	**12.54**	7.01	**46.14**	-4.10
	ZM	0.00	-1.56	0.00	-1.70	0.00	-0.21	0.00	-0.53	0.36	4.67	0.03	7.01	-8.30	-9.69
	LGS	-0.20	-0.87	-0.24	-4.04	0.00	-1.01	0.00	-1.55	0.36	9.74	0.03	6.39	-8.30	-2.17
Billboard06	-	**33.26**	-2.31	**26.51**	-9.50	**24.04**	-0.62	**22.86**	-0.23	**11.47**	9.71	**8.11**	8.08	**64.95**	-27.01
	ZM	0.00	-4.12	0.00	-5.87	0.01	-1.22	-0.09	-0.11	0.00	4.27	0.04	5.40	-0.08	-11.60
	LGS	-2.07	-4.06	-0.40	-6.65	0.01	-1.98	-0.09	-2.03	0.00	10.28	0.04	7.73	-0.08	-25.48
Billboard07	-	**30.73**	-12.69	**20.56**	-7.00	**12.76**	-1.01	**9.43**	-0.15	**13.35**	13.65	**13.35**	8.84	**45.41**	-33.17
	ZM	0.00	-12.69	-0.91	-4.84	0.00	-0.31	0.00	-0.17	0.00	6.12	-0.08	8.68	-0.33	-30.72
	LGS	0.87	-11.97	-0.43	-6.01	0.00	-0.95	0.00	-0.51	0.00	12.73	-0.08	8.54	-0.33	-32.83
Billboard08	-	**31.86**	-3.03	**24.46**	-2.93	**37.18**	0.78	**38.59**	-0.55	**12.14**	8.48	**12.31**	6.19	**43.15**	-8.01
	ZM	-0.11	-2.38	-1.02	-3.00	-0.02	0.31	-0.20	-0.29	0.55	1.30	0.29	6.19	-8.49	-8.93
	LGS	-3.28	-1.22	-0.00	-1.60	-0.02	-0.26	-0.20	-0.74	0.55	9.21	0.29	6.04	-8.49	-8.58
Billboard09	-	**37.41**	-1.11	**23.68**	-6.75	**14.29**	-0.32	**14.53**	-3.39	**19.81**	1.74	**15.84**	2.61	**61.73**	-2.55
	ZM	0.00	-1.11	-4.76	-5.89	0.00	-0.22	-0.00	-1.97	-0.03	0.47	-0.40	2.15	-0.94	-1.50
	LGS	-0.18	-1.25	0.19	-5.83	0.00	-0.83	-0.00	-3.72	-0.03	2.17	-0.40	2.34	-0.94	-4.17
Truck	-	**27.89**	0.09	**22.75**	-4.51	**10.22**	0.29	**12.68**	0.49	**14.08**	1.21	**11.97**	0.15	**42.88**	-0.11
	ZM	-0.35	-0.20	0.46	-0.22	-0.08	-0.22	-0.00	0.47	0.01	0.42	0.08	0.11	-0.40	-8.23
	LGS	-0.44	-0.39	-0.56	-4.62	-0.08	0.05	-0.00	-0.84	0.01	0.77	0.08	-0.18	-0.40	-9.12

methods that adapt adversarial features to transfer also within real-world scenarios.

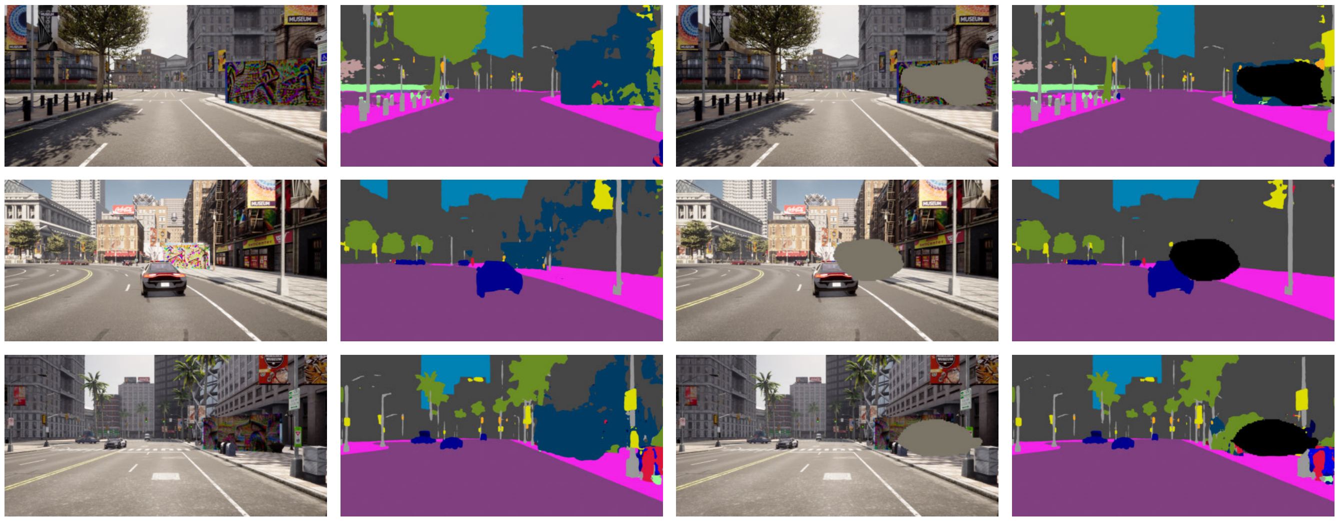
2) Testing Scenarios: Another issue with the simulated images is that urban scenarios in CARLA towns are too limited for a comprehensive evaluation of driving scenarios. Hence, the scenarios addressed in this paper do not cover all the possible patch attacks in the wild. The experiments tried to bridge the gap between the complexity of real-world scenarios and simulated environments by randomizing non-playable characters (NPCs), such as pedestrians and cars, across all images within a given scenario. Additionally, we proposed two types of attack settings: static attacks using adversarial billboards and dynamic attacks with adversarial trucks. These settings are designed to effectively generalize CARLA-GEAR across various attack scenarios. For instance, we can simulate adversarial road signs or walls using billboards. Despite these efforts, we acknowledge that the intrinsic limitations of CARLA environments might not provide the same diversity of scenarios as the real world. This issue will be addressed in a future work.

3) Computational and Timing Costs: The proposed dataset generation pipeline is computationally intensive and requires a considerable amount of time, even with a powerful machine. Such costs are mainly due to the optimization algorithms involved in crafting proper adversarial attacks. Future work will need to refine these optimizations, thus reducing the processing time and making the pipeline affordable even for more resource-constrained architectures. Nevertheless, at the current stage, we aim at alleviating this problem by (i) providing a set of datasets with pre-computed patches for a set of CNNs spanning four different tasks, and (ii) providing support for custom patch and dataset generation upon request to the authors.

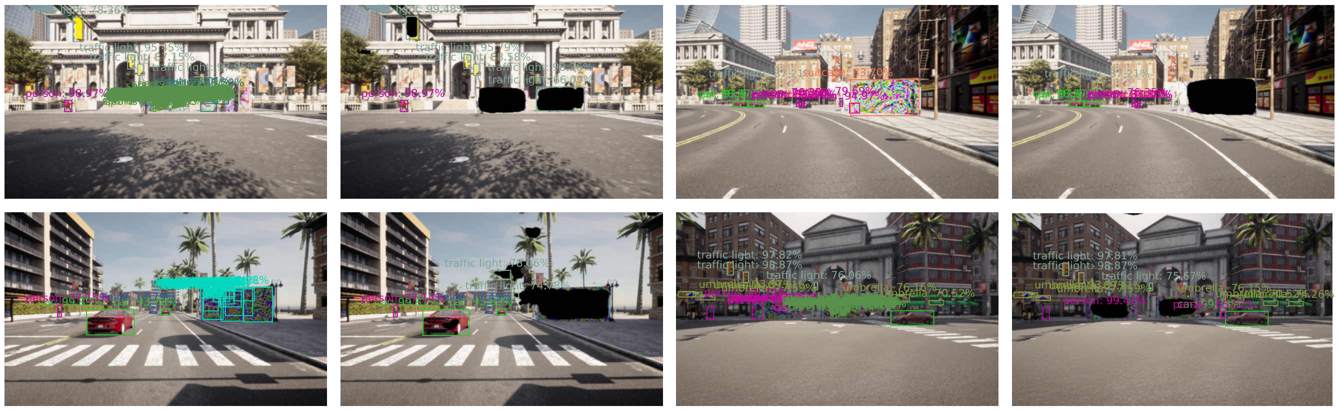
4) Adversarial Objects: Urban scenarios could include many types of adversarial objects, from car camouflage [56] to adversarial T-shirts [13], and other 3D objects [57]. Nevertheless, this work focuses on planar patches only, as they represent a flexible and generic attack scenario for vision models. Being CARLA flexible enough to include any texture that Unreal Engine 4 can manage, future work will assess the possibility of creating adversarial cities filled with different types of adversarial objects.

5) Attacks Strategies: By changing the loss function used during the patch optimization process it is possible to modify the attack objective, and, hence, its effect. This paper only focuses on white-box untargeted attacks, which represent a well-known approach for directly assessing worst-case adversarial scenarios. However, the tool could be quickly extended to study any kind of attack, e.g., targeted or black-box attacks.

6) Ad-hoc Robustness Metrics: To comprehensively evaluate the real-world robustness of CNNs, designers should face with the definition of proper ad-hoc, model-specfic metrics



(a)



(b)

Fig. 8. Illustration of the effectiveness of adversarial billboards (column one and two) and of Z-Mask [24] (column three and four) for Semantic Segmentation (a) and Object Detection (b). Images are extracted from the generated datasets.

that can better express the model robustness. In this work, we considered common task-based metrics (such as mAP for object detection, MioU for semantic segmentation, and RMSE for depth estimation), which are typically involved in evaluating the model performance on not attacked scenarios. The use of these metrics is a common practice for the majority of related work. Future work should instead be devoted to the derivation of more expressive metrics, which better depict and measure the concrete adversarial robustness of a given model.

B. Conclusion

This paper presented CARLA-GEAR(http://carlagear.retis.santannapisa.it), a tool for the automatic generation of photo-realistic synthetic datasets that can be used to systematically evaluate the adversarial robustness of CNNs against physically-realizable attacks and several learning tasks, such as object detection, semantic segmentation and depth estimation. The tool is based on the CARLA autonomous driving simulator, which overcomes the typical difficulties that arise while testing autonomous driving algorithms in various scenarios. The discussed procedural pipeline provides an automatic strategies for optimizing proper adversarial patches, placing them into simulated urban scenario, and exporting the photo-realistic synthetic dataset with the required labelled data.

Experimental tests were conducted to validate the potentiality of the tool, by evaluating the effect of multiple adversarial scenarios on both original models and defense mechanisms. Results have remarked the benefits of CARLA-GEAR, which, in a nutshell, enhances the testing coverage of the addressed models and defenses to clarify and question their applicability on autonomous computer vision scenarios.

As a final remark, it is worth noting that the automatic generation of datasets is only a first step towards the development of real-world benchmarks for the evaluation of adversarial robustness and adversarial defense methods. As a matter of fact, future work will address solutions for the open challenges discussed above, as well as the deployment of a standardized evaluation platform, which will eventually lead to a solid benchmark and so to a valuable leaderboard in adversarial detection, defense, and robustness.

REFERENCES

[1] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," 2017, *arXiv:1711.03938*.

[2] J. Zhang and C. Li, "Adversarial examples: Opportunities and challenges," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2578–2593, Jul. 2020.

[3] M. R. Alam and C. M. Ward, "Adversarial examples in self-driving: A review of available datasets and attacks," in *Proc. IEEE Appl. Imag. Pattern Recognit. Workshop (AIPR)*, Oct. 2022, pp. 1–6.

[4] J. I. Choi and Q. Tian, "Adversarial attack and defense of YOLO detectors in autonomous driving scenarios," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Aachen, Germany, Jun. 2022, pp. 1011–1017.

[5] J. Zhang, Y. Lou, J. Wang, K. Wu, K. Lu, and X. Jia, "Evaluating adversarial attacks on driving safety in vision-based autonomous vehicles," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3443–3456, Mar. 2022.

[6] F. Nesti, G. Rossolini, S. Nair, A. Biondi, and G. Buttazzo, "Evaluating the robustness of semantic segmentation for autonomous driving against real-world adversarial patch attacks," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2022, pp. 2826–2835.

[7] B. Biggio et al., "Evasion attacks against machine learning at test time," in *Proc. Eur. Conf. ECML PKDD*. Prague, Czech Republic: Springer, Sep. 2013, pp. 387–402.

[8] C. Szegedy et al., "Intriguing properties of neural networks," 2013, *arXiv:1312.6199*.

[9] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial Intelligence Safety and Security*. Boca Raton, FL, USA: Chapman & Hall/CRC, 2018, pp. 99–112.

[10] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," 2017, *arXiv:1712.09665*.

[11] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 284–293.

[12] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1765–1773.

[13] K. Xu et al., "Adversarial t-shirt! evading person detectors in a physical world," in *Proc. 16th Eur. Conf. Vis. (ECCV)*, Glasgow, U.K. Cham, Switzerland: Springer, Aug. 2020, pp. 665–681.

[14] T. Stolik, I. Lang, and S. Avidan, "SAGA: Spectral adversarial geometric attack on 3D meshes," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 4284–4294.

[15] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1528–1540.

[16] K. Eykholt et al., "Robust physical-world attacks on deep learning visual classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1625–1634.

[17] Z. Wu, S. Lim, L. S. Davis, and T. Goldstein, "Making an invisibility cloak: Real world adversarial attacks on object detectors," in *Proc. 16th Eur. Conf. Comput. Vis. (ECCV)*, A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, Eds. Glasgow, U.K. Cham, Switzerland: Springer, 2020, pp. 1–17.

[18] T. Wu, X. Ning, W. Li, R. Huang, H. Yang, and Y. Wang, "Physical adversarial attack on vehicle detector in the Carla simulator," 2020, *arXiv:2007.16118*.

[19] M. Lee and Z. Kolter, "On physical adversarial patches for object detection," 2019, *arXiv:1906.11897*.

[20] Y. Zhang, H. Foroosh, P. David, and B. Gong, "CAMOU: Learning physical vehicle camouflages to adversarially attack detectors in the wild," in *Proc. Int. Conf. Learn. Represent.*, 2019. [Online]. Available: https://openreview.net/pdf?id=SJgEl3A5tm

[21] A. Ranjan, J. Janai, A. Geiger, and M. J. Black, "Attacking optical flow," 2019, *arXiv:1910.10053*.

[22] J. Tu et al., "Physically realizable adversarial examples for LiDAR object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 13713–13722.

[23] K. Yamanaka, R. Matsumoto, K. Takahashi, and T. Fujii, "Adversarial patch attacks on monocular depth estimation networks," 2020, *arXiv:2010.03072*.

[24] G. Rossolini, F. Nesti, F. Brau, A. Biondi, and G. Buttazzo, "Defending from physically-realizable adversarial attacks through internal over-activation analysis," 2022, *arXiv:2203.07341*.

[25] P.-H. Chiang, C.-S. Chan, and S.-H. Wu, "Adversarial pixel masking: A defense against physical attacks for pre-trained object detectors," in *Proc. 29th ACM Int. Conf. Multimedia*, Oct. 2021.

[26] J. H. Metzen, N. Finnie, and R. Hutmacher, "Meta adversarial training against universal patches," 2021, *arXiv:2101.11453*.

[27] A. Saha, A. Subramanya, K. Patil, and H. Pirsiavash, "Role of spatial context in adversarial robustness for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 3403–3412.

[28] C. Xiang, A. N. Bhagoji, V. Sehwag, and P. Mittal, "$PatchGuard$: A provably robust defense against adversarial patches via small receptive fields and masking," in *Proc. 30th USENIX Secur. Symp. (USENIX Secur.)*, 2021, pp. 2237–2254.

[29] C. Xiang, S. Mahloujifar, and P. Mittal, "$PatchCleanser$: Certifiably robust defense against adversarial patches for any image classifier," in *Proc. 31st USENIX Secur. Symp. (USENIX Secur.)*, 2022, pp. 2065–2082.

[30] E. Chou, F. Tramèr, and G. Pellegrino, "SentiNet: Detecting localized universal attacks against deep learning systems," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2020, pp. 48–54.

[31] Z. Xu, F. Yu, and X. Chen, "LanCe: A comprehensive and lightweight CNN defense methodology against physical adversarial attacks on embedded multimedia applications," in *Proc. 25th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2020, pp. 470–475.

[32] B. Li et al., "Detecting adversarial patch attacks through global-local consistency," in *Proc. 1st Int. Workshop Adversarial Learn. Multimedia*, 2021, pp. 35–41.

[33] K. T. Co, L. Muñoz-González, L. Kanthan, and E. C. Lupu, "Real-time detection of practical universal adversarial perturbations," 2021, *arXiv:2105.07334*.

[34] M. Naseer, S. Khan, and F. Porikli, "Local gradients smoothing: Defense against localized adversarial attacks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 1300–1307.

[35] G. Rossolini, F. Nesti, G. D'Amico, S. Nair, A. Biondi, and G. Buttazzo, "On the real-world adversarial robustness of real-time semantic segmentation models for autonomous driving," 2022, *arXiv:2201.01850*.

[36] Y. Dong et al., "Benchmarking adversarial robustness on image classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 321–331.

[37] L. Hsiung, Y.-Y. Tsai, P.-Y. Chen, and T.-Y. Ho, "CARBEN: Composite adversarial robustness benchmark," in *Proc. 35st Int. Joint Conf. Artif. Intell.*, Jul. 2022.

[38] N. Mu and J. Gilmer, "MNIST-C: A robustness benchmark for computer vision," 2019, *arXiv:1906.02337*.

[39] M. Pintor et al., "ImageNet-patch: A dataset for benchmarking machine learning robustness against adversarial patches," *Pattern Recognit.*, vol. 134, Feb. 2023, Art. no. 109064.

[40] B. Jefferson and C. O. Marrero, "Robust assessment of real-world adversarial examples," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 3442–3449.

[41] B. Nassi, Y. Mirsky, D. Nassi, R. Ben-Netanel, O. Drokin, and Y. Elovici, "Phantom of the ADAS: Securing advanced driver-assistance systems from split-second phantom attacks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 293–308.

[42] Z. Kong, J. Guo, A. Li, and C. Liu, "PhysGAN: Generating physical-world-resilient adversarial examples for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14242–14251.

[43] H. Zhou et al., "DeepBillboard: Systematic physical-world testing of autonomous driving systems," in *Proc. IEEE/ACM 42nd Int. Conf. Softw. Eng. (ICSE)*, Oct. 2020, pp. 347–358.

[44] A. Braunegg et al., "Apricot: A dataset of physical adversarial attacks on object detection," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2020, pp. 35–50.

[45] H. Pan, Y. Hong, W. Sun, and Y. Jia, "Deep dual-resolution networks for real-time and accurate semantic segmentation of traffic scenes," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3448–3460, Mar. 2023.

[46] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[47] T. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[48] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

[49] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiseNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 325–341.

[50] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[51] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[52] D. Kim, W. Ka, P. Ahn, D. Joo, S. Chun, and J. Kim, “Global-local path networks for monocular depth estimation with vertical CutDepth,” 2022, *arXiv:2201.07436*.

[53] S. F. Bhat, I. Alhashim, and P. Wonka, “AdaBins: Depth estimation using adaptive bins,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jan. 2021, pp. 4009–4018.

[54] P. Li, X. Chen, and S. Shen, “Stereo R-CNN based 3D object detection for autonomous driving,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7636–7644.

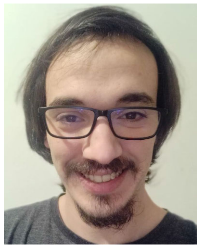
[55] A. Dosovitskiy et al., “An image is worth 16×16 words: Transformers for image recognition at scale,” 2020, *arXiv:2010.11929*.

[56] N. Suryanto et al., “DTA: Physical camouflage attacks using differentiable transformation network,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 15284–15293.

[57] C. Chen and T. Huang, “Camdar-adv: Generating adversarial patches on 3D object,” *Int. J. Intell. Syst.*, vol. 36, no. 3, pp. 1441–1453, Mar. 2021.

Federico Nesti received the degree (cum laude) in robotics and automation engineering from the University of Pisa. He is currently pursuing the Ph.D. degree with the Real-Time Systems (ReTiS) Laboratory, Scuola Superiore Sant’Anna, Pisa, where he investigated adversarial examples in the physical world and the trustworthiness of AI-based computer vision systems.

Giulio Rossolini (Member, IEEE) received the degree (cum laude) in embedded computing systems engineering and the master’s degree jointly offered by the Scuola Superiore Sant’Anna, Pisa, and the University of Pisa. He is currently pursuing the Ph.D. degree with the Real-Time Systems (ReTiS) Laboratory, Scuola Superiore Sant’Anna. His current research interests include the design and implementation of software tools to support and increase the trustworthiness of machine learning algorithms used in computer vision applications and safety-critical systems.



Gianluca D’Amico received the degree in computer science and networking and the master’s degree jointly offered by the Scuola Superiore Sant’Anna, Pisa, and the University of Pisa. He is currently pursuing the Ph.D. degree with the Real-Time Systems (ReTiS) Laboratory, Scuola Superiore Sant’Anna. His current research interests include computer graphics, especially the LiDAR technology, and the design and implementation of visual simulation tools to test algorithms and neural network approaches in the railway environment, and automotive.

Alessandro Biondi (Member, IEEE) received the degree (cum laude) in computer engineering from the University of Pisa, Italy, within the excellence program, and the Ph.D. degree in computer engineering from the Scuola Superiore Sant’Anna, under the supervision of Prof. Giorgio Buttazzo and Prof. Marco Di Natale. Since 2016, he has been a Visiting Scholar with the Max Planck Institute for Software Systems, Germany. He is currently an Associate Professor with the Real-Time Systems (ReTiS) Laboratory, Scuola Superiore Sant’Anna. His research interests include the design and implementation of real-time operating systems and hypervisors, schedulability analysis, cyber-physical systems, synchronization protocols, and safe and secure machine learning. He was a recipient of six best paper awards, one outstanding paper award, the ACM SIGBED Early Career Award 2019, the IEEE TCCPS Early Career Award 2023, and the EDAA Dissertation Award 2017.

Giorgio Buttazzo (Fellow, IEEE) received the degree in electronic engineering from the University of Pisa, the M.S. degree in computer science from the University of Pennsylvania, and the Ph.D. degree in computer engineering from the Scuola Superiore Sant’Anna, Pisa. He is currently a Full Professor of computer engineering with the Scuola Superiore Sant’Anna. He has authored seven books on real-time systems and more than 300 articles in the field of real-time systems, robotics, and neural networks. He received 13 best paper awards. He has been the Editor-in-Chief of *Real-Time Systems* and an Associate Editor of *ACM Transactions on Cyber-Physical Systems*.