The Visual Computer
https://doi.org/10.1007/s00371-025-03811-1

RESEARCH

Exploring the effects of synthetic data generation: a case study on autonomous driving for semantic segmentation

Manuel Silva[1,2] · Antonio Seoane[2] · Omar A. Mures[1,2] · Antonio M. López[3,4] · Jose A. Iglesias-Guitian[1,2]

Accepted: 9 January 2025

Abstract
Rendering 3D virtual scenarios has become a popular alternative for generating per-pixel-labeled image datasets, especially in fields like autonomous driving. The approach is valuable for training neural perception models, such as semantic segmentation models, particularly when data might be scarce, expensive, or difficult to collect. However, fundamental questions persist within the research community regarding the generation and processing of these synthetic images, particularly a better understanding of the key factors influencing the performance of deep learning models trained with such synthetic images. In response, we conducted a series of experiments to elucidate the impact that common aspects involved in the generation of rendered synthetic images may have on the performance of neural semantic segmentation tasks. Our study used a recent autonomous driving synthetic dataset as our main testbed, allowing us to investigate the effect of different approaches when modeling their geometric, material, and lighting details. We also studied the impact of rendering noise, typically produced by path-tracing algorithms, as well as the impact of using different color transformations and tonemapping algorithms.

Keywords Computer graphics · Rendering · Autonomous driving · Semantic segmentation

1 Introduction

Autonomous driving has fostered research in multiple computer vision tasks, such as object detection or semantic segmentation, so as to solve the ultimate challenge of self-driving vehicles. Training perception models is a key fundamental component of many of those solutions. In order to train such models, a great demand for acquiring and labeling driving datasets aroused, including labeling image datasets. Acquiring and labeling datasets for computer vision tasks can be really expensive, taking several minutes to annotate a single image for an average human annotator. Thus, data labeling can easily become an important bottleneck for deep learning supervised learning approaches.

Currently, there are multiple co-existing works aiming to address the source of this problem. In the literature, one can find approaches that minimize human intervention by selecting the most informative samples for labeling, like active learning [18, 43], or approaches that use less precise labels to reduce the annotation effort, as proposed in weakly supervised learning [38]. It is also common to use model predictions with the highest confidence levels to automatically label previously unlabeled samples and then train and improve the previous model's performance. This approach is known as semi-supervised learning [25]. Another paradigm is to design pretext tasks to create supervisory signals; those approaches are typically known as self-supervised learning [13].

In parallel, a promising trend has emerged, focusing on enhancing supervised learning through the generation of synthetic datasets leveraging virtual simulations [11]. This approach offers a compelling solution, as it enables automatic labeling of the generated image data during the synthesis process, eliminating the need for real image acquisition or human annotators. By leveraging virtual environments, it is possible to create diverse and annotated datasets tailored to specific tasks in autonomous driving. Moreover, in scenarios where real-world data are challenging, expensive, or impossible to

✉ Manuel Silva
m.silva1@udc.es

1 CITIC - Centre for ICT Research, A Coruña, Spain

2 Universidade da Coruña, A Coruña, Spain

3 Computer Vision Center & Computer Science Dpt., Universitat Autònoma de Barcelona, Bellaterra, Spain

4 Computer Science Department, Universitat Autònoma de Barcelona, Escola d'Enginyeria, Edifci Q, 08193 Bellaterra, Spain

Published online: 07 February 2025

obtain, as in rare, dangerous, or unavailable environments, virtually simulated synthetic datasets become the primary, and sometimes the only resource, for developing new models.

The resulting images of a virtual simulation can be affected by many aspects, such as the configuration of the modeled sensors, their noise, the simulated lighting and/or weathering conditions, and ultimately the parameters affecting image rendering algorithms (which may include, i.e., the rendering samples per pixel, number of simulated indirect ray bounces, etc.). Studying the latter would require full access to assets and rendering pipelines to regenerate equivalent image datasets with new parameters and re-evaluate the perception models accordingly. This is a tedious and costly task; consequently, very few papers exist conducting such ablative analyses.

Moreover, similar to real datasets that belong to different domains, synthetic datasets can also suffer from a domain shift problem, especially when compared to real datasets, typically referred to as the synth-to-real domain gap. Reducing such domain gap is an open research problem, especially in the area of domain adaptation, where several methods have been proposed to diminish the domain gap [32]. In general, it is widely accepted that the level of realism can help to mitigate the synth-to-real domain gap [47]. However, very few ablative studies exist that systematically analyze other relevant aspects involved in the generation and post-processing of synthetic image datasets. In this paper, we decided to investigate the influence of a subset of typical choices researchers have to take when generating synthetic image datasets to train perception models, focusing on their impact on semantic segmentation as a representative computer vision task.

In our study, we focus on exploring several aspects involved in 3D digital modeling that might influence the training of semantic segmentation perception models, starting with the amount of 3D geometric detail or the realism in material modeling, followed by the impact of generating images under various lighting conditions, i.e., using just direct lighting versus incorporating more complex indirect lighting. We address detailed aspects such as how relevant it might be to calculate more or less bounces for reflections and indirect shadows or the impact of Monte Carlo rendering noise typically produced by path-tracing rendering algorithms, the industry standard nowadays for photorealistic rendering. To satisfy the requirements of the proposed study, we obtained privileged access to the recent autonomous driving dataset UrbanSyn [10] and their digital assets. This last step was critical in order to regenerate the required image data needed to address the aforementioned research questions, as needed for such an extensive ablative study. Finally, we also explored the influence of alternative image post-processing techniques typically applied to rendered images. Since most computer vision models are prepared to digest low dynamic range (LDR) images, we study the influence that common tonemapping operators converting high dynamic range (HDR) into LDR images might have when training segmentation models. Moreover, we consider color space transformations typically applied to those images before training.

All our experiments are validated using three famous real-world datasets often used by the research community in autonomous driving: Cityscapes [6], Mapillary Vistas [21], and BDD100K [45]. Differently from many previous works that specialized the hyperparameters for each validation target, we opted to use the same model trained with common hyperparameters to validate on the three validation sets whenever that was feasible for each specific experiment. In summary, the main contributions of this work are:

- We conducted an extensive ablative study using the UrbanSyn dataset to evaluate the performance of semantic segmentation models. Our analysis focuses on the influence of several key factors in the image synthesis pipeline, including 3D geometric modeling, material realism, the comparative effects of simple versus complex lighting, the impact of Monte Carlo (MC) rendering noise produced by path tracers, and the impact of using different tonemapping operators and/or color space transformations.
- From our experiments, we distilled valuable conclusions across most of the studied categories, leading to the development of clear guidelines for improving the creation of future synthetic datasets. Additionally, boosted by these findings, we developed new models in the autonomous driving domain that trained on synthetic data outperform our synthetic baselines for semantic segmentation.

2 Related work

Studying and understanding the influence that image datasets have on the performance of computer vision models are a research question that motivated different investigation efforts to cover datasets consisting of either real or synthetic images. For the first group, image quality has already been one of the topics covered, for instance, studying how image classification is affected when images in a training set suffer different quality distortions, such as blur, noise, contrast, or even image compression artifacts [7]. In addition to the early discoveries about the advantages of generating labeled synthetic data [23], there has been an increasing demand to gain deeper insights around the image generation processes involved. Next, we revisit recent advancements in these directions, focusing on their exploratory nature.

2.1 Synthetic datasets and good practices

There are not many studies regarding how synthetic datasets should be generated to achieve better performance of machine vision systems. Paulin and Ivasic-Kos [22] presented a recent overview of existing synthetic datasets in the computer vision domain. They analyzed the methods and techniques mostly used for synthetic dataset generation, as well as describing various techniques aimed at improving image realism (by adding global noise) or techniques for solving domain and distribution gaps. Tsirikoglou et al. [36] presented a survey that reviews existing image synthesis methods for visual machine learning from a computer graphics perspective. The methods were categorized based on modeling and rendering aspects and discussed with respect to their target computer vision applications.

Already focusing on a specific domain, Zhang et al. [47] introduced a large-scale synthetic dataset with 500K physically based rendered images from 45K realistic 3D indoor scenes. The authors studied the effects of rendering methods and scene lighting on training for three computer vision tasks: surface normal prediction, semantic segmentation, and object boundary detection. This study provided tangible insights about good practices for training with synthetic data, such as the fact that more realistic rendering should be worth it, and pre-training a neural network backbone with synthetic datasets could improve over the previous state of the art on those three tasks.

In recent years, numerous synthetic datasets have emerged. Particularly in the field of autonomous driving, Ros et al. [29] and Ritcher et al. [28] were among the first ones to achieve remarkable research milestones using synthetic datasets. Some years later, Synscapes [41] also demonstrated the improvements achieved by using a more realistic rendering. More recently, the UrbanSyn dataset [10] was released, allowing to improve state-of-the-art synthetic baselines for semantic segmentation.

Complementary, other works focus on investigating semi-automatic creation pipelines for synthetic datasets addressing different computer vision tasks, e.g., object viewpoint estimation from rendered car images [19]. Their authors also explored the idea of combining synthetic images with a small amount of real data to fine-tune models and improve their estimation accuracy. Generally, these approaches rely on state-of-the-art rendering software to generate such large labeled datasets and model specific domains [37].

2.2 Studies on the image generation process

Within the domain of autonomous driving, specific studies have been released about the impact of camera parameters on neural network generalization capabilities [16]. Their findings suggest that the performance obtained from training on physically based simulations of camera images is nearly as effective in generalizing to real camera images as the performance achieved when generalizing between different camera image datasets.

There were previous studies about the impact of randomizing rendering parameters, such as scene lighting, by using techniques like domain randomization [35]. Lighting realism was also studied by Zhang et al. [47] as part of their studies, suggesting that realistic rendering typically helped.

Domain adaptation addresses how to reduce the domain gap among different dataset domains. In a sense, color transfer between source and target domains could be viewed as an initial endeavor to bridge this gap. Therefore, understanding color transfer is of significant interest, as it aims to reproduce an ideal color scheme on a source image that can be learned from a target reference one. Recent surveys, such as the one conducted by Lv et al. [17], have thoroughly reviewed mainstream methods for color transfer, along with related theories and frameworks.

The closest work in spirit to ours is Schlachter et al. [31] which discussed how different rendering techniques can affect the performance of final models. Their study incrementally enhanced the sophistication of rendering by incorporating increasing levels of complexity in shading, shadowing, and global illumination. Additionally, their authors proposed an ensemble of Generative Adversarial Network (GAN) models to approximate global illumination, particularly within the domain of 3D toy models. In their domain, GAN ensemble models demonstrated significant domain adaptation performance. Another interesting finding was that using learning-based denoisers could reduce the classification accuracy, suggesting that denoisers may modify image statistics relevant for their classification task. This finding is aligned with the latest research in post-denoising [9] evidencing that learning-based denoisers often introduce a certain bias in their denoised images.

In this study, we focus our experiments toward gaining new knowledge about how synthetic datasets in the autonomous driving field should be generated, using UrbanSyn as our testbed. We believe our study brings new insights to the research community. While previous ablative studies focused their analysis on CNN-based architectures [31] and general image classification tasks, our study covers systematic experiments for semantic segmentation using two common architectures (CNN and transformer based). We discarded diffusion models, which are recently also being applied to semantic segmentation problems [4], because of their non-negligible needs in terms of GPU resources. Our ablative study involves geometric detail, relevance of realistic material modeling, importance of complementary lighting, and the influence of Monte Carlo path-tracing noise. We also study the influence of HDR image tonemapping and color

correction on the performance of the segmentation models under study.

The design of all these experiments is detailed in Sect. 3, and the obtained results are reported in Sect. 4. Finally, we extracted some guidelines for crafting synthetic datasets in Sect. 5.

3 Design of experiments

In this section, we describe our experimental design, starting in Sect. 3.1 with a brief description of the synthetic and real-world datasets used in this work. Next, in Sect. 3.2 we describe the rendering framework utilized that allowed us to replicate various versions of the chosen synthetic dataset. Then, we describe the semantic segmentation models chosen to run our benchmarks in Sect. 3.3. After that, we describe details of each experiment dedicated to gain insights about: the importance of fine geometric details (see Sect. 3.4); the importance of material modeling realism (see Sect. 3.5); and the influence of simulated lighting (see Sect. 3.6). Next, we propose to gain some insights about the influence of Monte Carlo path-tracing rendering noise (see Sect. 3.7); the influence of alternative tonemapping operators (see Sect. 3.8); and the influence of color transformation algorithms (see Sect. 3.9).

3.1 Synthetic testbed and validation datasets

UrbanSyn. To assess the impact of different rendering parameters in synthetic data, we chose UrbanSyn [10] as our testbed synthetic dataset that we can use as our baseline to determine the effects of different settings of the generating process. The original UrbanSyn dataset consists of 7539 images with a resolution of 2048 × 1024 pixels, paired with semantic segmentation labels that use the same 19 classes proposed in Cityscapes. Furthermore, they are not divided into splits because all of them are used for training. These images belong to four different urban scenarios and were captured using the same sensor width and focal length as in Cityscapes. In addition, the camera is always located outside the vehicle, as it is the most common location in driving datasets.

Synthetic testbed: UrbanSyn.* An important challenge that may pass unnoticed is that our experiments require regenerating, from scratch, the same set of images contained in UrbanSyn. This is required to simply alter the original geometries, materials, or to render the original scenarios again but with different parameters. After that regeneration process, we noticed very minor differences with respect to the original; thus, we decided to refer to this new rendered version as UrbanSyn*. Figure 1 shows the differences between the new UrbanSyn* and the original version. To precisely replicate UrbanSyn* we need to store rendering node seeds and make our internal procedures more deterministic. After this, we can adopt this new version of the dataset, as it ensures the reproducibility of our experiments and maintains consistent conditions for our comparisons.

Fig. 1 We show minor differences between the original UrbanSyn (left) and our generated UrbanSyn* (right) used in our new ablation studies. The scene distribution and appearance are consistent, aside from minor car color variations due to the difficulty in replicating identical random choices

Real-world validation datasets. To evaluate the results of our study, we use three real-world datasets: Cityscapes [6], Mapillary Vistas [21], and BDD100K [45]. These datasets do not share a uniform class convention for semantic segmentation, so we standardized our evaluation using the 19 classes proposed by Cityscapes, the de facto standard for autonomous driving research. Typically, these datasets are split into training, validation, and testing sets. However, the test sets do not have publicly available labels, and accessing them requires requesting the dataset owners to run all our experiments. Given the extensive number of experiments, this approach was not feasible. Therefore, our results are based solely on the validation sets.

Cityscapes is a real-world dataset that contains 5000 finely annotated images with a resolution of 2048 × 1024 pixels that were split into: 2975 images for the training set, 500 images for the validation set, and 1525 images for the test set. The images were captured in different German cities with the same sensor configuration, which results in a very homogeneous look. It is a standard in the computer vision field, and it is commonly used as a benchmark in semantic segmentation.

Mapillary Vistas is another real-world dataset. This dataset comprises 25,000 images, split into 18,000 images for the training set, 2000 images for the validation set, and 5000 images for the testing set. Unlike Cityscapes and BDD100K, these images were captured with different sensors, different resolutions, and all over the world. The class convention is also different; they used two conventions with 66 and 124 different classes. To utilize this dataset in our work, it has been necessary to change the Mapillary classes convention to the Cityscape's one. Our criteria were to convert those classes that represent the same object and to label as void the other classes that would not fit into any other category precisely, as it is standard.

Fig. 2 Example images of the real-world validation datasets used in our experiments

Unlike the original UrbanSyn benchmark, where the authors prefiltered the Mapillary Vistas images to ensure a consistent aspect ratio, we utilized the full collection of images in our experiments. This decision aligns with our philosophy of prioritizing generalization. By retaining all images, we aim to train a single robust model rather than optimizing for the best possible results with multiple specialized models.

BBD100K is the last real-world dataset used in this work. This dataset consists of 100,000 videos of driving scenes. However, it only has 10,000 images for the semantic segmentation task, divided into 7000 for the training set, 2000 for the validation set, and 1000 for the test set. The images were captured in Berkeley, San Francisco, New York, and the Bay Area with a resolution of 1280 $\times$ 720 pixels, and they used the same class convention as Cityscapes. The data were captured with the same camera model but with different configurations regarding the location of the camera and the recording vehicle, which results in images with a different image composition compared to Cityscapes. Figure 2 shows a sample of each real-world dataset.

3.2 Rendering framework

We obtained privileged access to the digital assets of the recently published autonomous driving dataset UrbanSyn. This allowed us to regenerate the whole image dataset needed to address the planned research questions by creating different versions of the synthetic source dataset and conducting the corresponding ablation studies.

As described in UrbanSyn, we utilize a path-tracing rendering engine to recreate different synthetic image datasets and conduct the ablation studies. The rendering engine consists of a Monte Carlo path tracer that uses an adaptive sampling algorithm, configured with a minimum of 128 and a maximum of 256 samples per pixel depending on the estimated error noise, and the corresponding threshold, to determine the exact number of samples assigned for each pixel. Furthermore, they used a specular depth of 4, a diffuse depth of 4, and a scatter depth of 1 or 64, depending on whether the image has a participating medium or not. Regarding the lighting parameters, UrbanSyn uses several HDRs with different sun and cloud dispositions. Besides, in order to increase the simulation's realism and provide more diversity, a participating medium was set up in two of the four scenarios to mimic scattering atmospheric effects.

We use this rendering framework to create variations on lighting, rendering noise, and color transformations to study the influence that these changes have on the neural semantic segmentation model's performance. The new sets of data that we created were not exactly the same as the original data due to some features that were not possible to reproduce. Specifically, we identify the following changes: the color of the vehicles, the pattern of the vehicle's lights, and some minor differences in certain HDRs. The most noticeable distinction is the car's color, which is the only modification that appears in every frame. We can observe that alteration in Fig. 1.

3.3 Experimental framework

We use two different neural networks for semantic segmentation to perform our experiments: DeepLabv3+ [5] and SegFormer [44]. We decided to use both because they represent two different approaches to the task: DeepLabv3+ uses an encoder-decoder architecture typically with a convolutional neural network (CNN) backbone. SegFormer instead uses a hierarchical Transformer encoder and a simple multilayer perceptron decoder. Using two different architectures, we can study their differences with respect to our ablative study.

However, we reckon that recent alternatives based on Denoising Diffusion Probabilistic Models (DDPMs) for semantic segmentation [4] are rapidly gaining traction and would also be an interesting addition; however, the extensive training hours and GPU resources required to retrain DDPMs present a well-known challenge [40]. Adopting DDPMs would be prohibitive given our plans for extensive experimentation, which already included nearly one hundred experiments.

DeepLabv3+ model and hyper-parameters. We use the same DeepLabv3+ model utilized in the original UrbanSyn benchmarks based on Detectron2 [42]. The authors trained separate models using specific hyper-parameters and color transformations tailored for each validation set (Cityscapes, Mapillary Vistas, and BDD100K). In this research, we prioritized the generalization of the extracted conclusions over achieving the best possible performance on each individual dataset. Interestingly, as we will show, some of our newly trained models can directly compare with those optimized using more specialized hyper-parameters. To prevent biases toward one of the three validation datasets, we selected hyper-parameters that could perform acceptably across all three. In Table 1, we show the differences between the hyper-parameters used in the original UrbanSyn's benchmark for DeepLabv3+ and those used in this ablation study.

SegFormer model and hyper-parameters. We use the same SegFormer implementation available in MMSegmentation.

We adopt the same hyper-parameters used in the original UrbanSyn benchmarks, with the main difference that we adopted the B3 model with just 90k iterations instead of the B5 model with 180k iterations, since it was judged enough to extract general conclusions without wasting the available time and resources.

Fine-tuning. One of the main uses for synthetic data is to pre-train real data models to improve their performance and surpass the original baselines; this approach was proved successfully across different domains [41, 47]. For this reason, we decided to perform fine-tuning experiments using our best single-source models and fine-tune them with our three real datasets to analyze their ability to outperform the original models. For these experiments, we use only SegFormer because it produces the best results; furthermore, we use the same hyper-parameters as the single-source experiments, including the learning rate.

Domain shift adaptation. The original UrbanSyn uses a simple domain adaptation technique based on Reinhard's color transfer [26]. This technique consists of transferring the color from a target image to a source image using the CIELAB color space. However, in UrbanSyn, source and target images were randomly selected, thus producing non-optimal results. As we used three datasets instead of one to validate the model's performance, we have to get as a color transformation target any image belonging to the three datasets with an equal distribution to avoid an overrepresentation of the datasets with more images.

Experiment Deviations. Neural networks inherently contain non-deterministic elements that can lead to different results across multiple training runs, even with identical data, hyper-parameters, and seeds. To account for this variability, we performed three runs of the baselines for each network, measuring standard deviations to assess statistical significance, as shown in Table 2.

Fig. 3 We illustrate the impact of polygon count reduction on specific classes and how a reduced polygon count may affect visual quality (left). We also show wireframe visualizations (on the right) showing the changes in the overall geometric shape due to polygon reduction. Note that the displacement used is a per-pixel process that tesselates the geometry on demand during the rendering process, to create a more complex geometry

3.4 Effects of geometric modeling

The goal of this set of experiments was to find out the effect that fine geometric details may have on the performance of a semantic segmentation model. We simplified the geometric complexity only for objects belonging to a subset of classes. In our study, we focus our analysis on pedestrians, riders, vehicles, traffic signs, traffic lights, poles, and sidewalks.

To achieve this, we reduced the polygon count of the 3D meshes by using Blender's implementation of the Quadric Error Metrics (QEM) algorithm, which reduces geometry while preserving objects' shapes. We choose the percentage of reduction depending on the class, trying to remove the maximum quantity of polygons possible while maintaining their recognizability. We retained between 3% and 5% of the polygons in the vehicles, pedestrians, and riders classes. We preserve 10% for the poles, 15% for the traffic lights, and 20% for the traffic signs due to the few polygons these classes already have. In addition, we perform another kind of geometry simplification for the sidewalk class. This class stored most of the geometric details in the displacement maps inside its materials, using displacements to create the added geometry while rendering; for that reason, we removed that feature.

Finally, we produced a new rendered version of UrbanSyn, using the simplified geometry only for those specific classes under analysis. Last, we run the segmentation models to get the results, which are reported in Sect. 4.1. In Fig. 3, we illustrate examples of the main differences expected among original and reduced versions for assets belonging to some of the classes included in this study.

Table 1 Hyperparameter's comparison between the original UrbanSyn's configuration and our settings for DeepLabv3+

	Framework	Target	Learning rate	Batch size	Crop size
DLV3+	UrbanSyn	Cityscapes	0.002	8	1024 × 512
		Mapillary	0.002	16	816 × 608
		BDD	0.0001	4	1280 × 720
	Ours	All	0.000125	8	1024 × 512

Table 2 Baselines for UrbanSyn and UrbanSyn* using three runs and calculating the standard deviation for each experiment

Target	Metric	DeepLabv3+		SegFormer	
		UrbanSyn	UrbanSyn*	UrbanSyn	UrbanSyn*
Cityscapes	mIoU	51.92	52.26	59.53	60.65
	Std. Dev	±0.89	±1.7	±0.45	±0.58
Mapillary	mIoU	48.97	48.99	55.55	56.26
	Std. Dev	±0.55	±0.62	±0.82	±0.19
BDD	mIoU	33.08	33.87	40.79	40.26
	Std. Dev	±0.68	±0.72	±1.65	±1.14
	Mean	44.66	45.04	51.96	52.39
	Std. Dev	±0.7	±1.01	±0.97	±0.64

Fig. 4 UrbanSyn dataset rendered with three material modeling approaches tested in our experiments: (left) using a procedurally generated random noise to replace the albedo; (center) using random solid colors facilitating object parts differentiation; and (right) UrbanSyn's original approach with realistic BRDF and textures

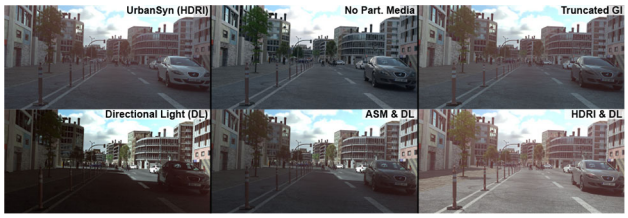

Fig. 5 We authored different variations in UrbanSyn to study the influence of lighting, ranging from the original UrbanSyn lit with an HDRI and simulating participating media, to using an HDRI but no participating media at all, using a GI approximation with a reduced number of ray bounces (Truncated GI), lighting using a single directional light (DL), or using a DL as part of an analytical sky model (ASM & DL), or combined with an HDRI (HDRI & DL)

3.5 Effects of material appearance

Crafting realistic material models for 3D synthetic datasets might be expensive and time-consuming. Domain randomization [34] introduced the idea that randomizing material parameters, among others, could force neural networks to learn the essential features of the objects of interest. Tremblay et al. [35] later demonstrated its usefulness for object detection tasks in the autonomous driving field. Following this hypothesis, we evaluate a similar approach in the context of semantic segmentation.

We created two alternative versions of the dataset concerning their material modeling. In the first version, we use materials with noisy grayscale texture patterns replacing the original albedo texture. Other textures that did not control the albedo, such as displacement or normal maps, remained unchanged. In the second version, we replace the original albedo texture with a random solid color, allowing us to differentiate objects and parts of objects, as opposed to the homogeneous grayscale noise pattern, which aims to force segmentation models to fix their attention on different features other than color. Please check Fig. 4 to see an example of the UrbanSyn dataset rendered using the three different material versions. The results of this experiment are detailed in Sect. 4.2.

3.6 Lighting alternatives

Photorealistic synthetic datasets for automated driving, i.e., Synscapes or UrbanSyn, often feature low-frequency lighting by leveraging path-tracing rendering and using a sky environment, either represented by a captured HDR panoramic image (HDRI) or by using an analytical sky model (ASM). However, generating those datasets using other reasonable

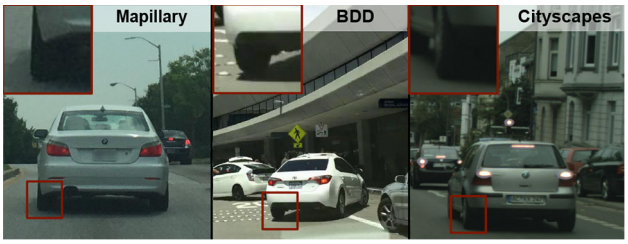

Fig. 6 Examples of noise present in real-world datasets. Cameras often introduce noise due to factors like lighting conditions or sensor quality. It is manifested as graininess, color distortion, or random variations in pixel values, which can affect the performance of neural networks trained on (clean) synthetic data

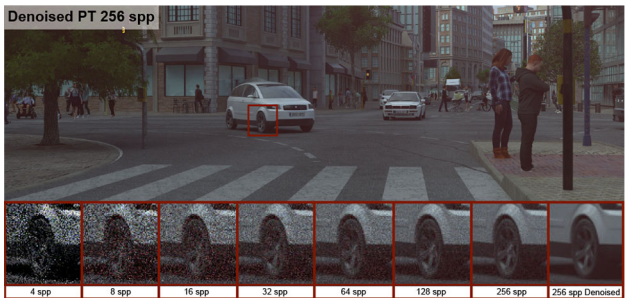

Fig. 7 Different sampling budgets tested in UrbanSyn*. MC rendering noise has a different nature than real cameras sensor noise. However, from a network's perspective, a moderate amount of noise in the training data could indirectly help the network become more robust when exposed to real camera noise

lighting alternatives has not yet been fully explored, despite the potential benefits of including enough lighting variations already described in the literature [19].

In this study, we propose different alternatives for rendering UrbanSyn and compare various lighting versions. We authored six different lighting variations, as shown in Fig. 5: the original UrbanSyn, lit with an HDRI and simulating participating media; an HDRI but no participating media at all; an ambient occlusion GI approximation (Truncated GI) with one or two ray bounces per light path and a non-unbiased kernel; a single directional light (DL); a DL as part of an analytical sky model (ASM & DL); and a DL combined with an HDRI (HDRI & DL).

This approach allowed us to cover a wide range of effects: hard shadows projected by a key directional light simulating the sun, soft shadows projected by a low-frequency GI, having an approximated GI with ambient occlusion and less photorealism, or having a low-frequency GI coupled with some key directional light, which combines the effects of both hard and soft shadows at the same time. Similarly, we can weigh the influence and trade-off of simulating participating media. Furthermore, we combined some of these different UrbanSyn versions to check the impact that using different lighting techniques together has on the results. All our lighting experiments are shown in Sect. 4.3.

3.7 Stochastic rendering noise

A key observation when generating synthetic datasets is the possibility of modeling camera sensor noise typically present in real data [16]. See Fig. 6 for examples of common sensor noise present in real-world datasets. Real noise could not only be a consequence of the camera sensors employed but also a side effect of the compression algorithms used to store images. All of them can affect a neural perception model's performance. Particularly when generating synthetic datasets, we can also find another source of noise: Monte Carlo (MC) rendering noise, a typical side effect of using MC path-tracing rendering. Such noise could also impact the results of a semantic segmentation neural network. Thus, we decided to study this potential influence by rendering UrbanSyn using different sampling budgets (see Fig. 7) and also combining noisy with denoised renderings in the same dataset. The results of these experiments are all reported in Sect. 4.4.

3.8 Tonemapping

While HDR images may contain richer visual information, their inclusion in training datasets introduces challenges such as increased storage and computational demands. Additionally, their usage may lead to potential inefficiencies during model training that are still being addressed by the community [20]. For these reasons, in neural semantic segmentation tasks, low dynamic range (LDR) images are commonly used for training neural networks.

Tonemapping operators (TMOs) are designed to convert high dynamic range (HDR) content into a lower dynamic range (LDR) format, a process typically used to map HDR images on standard screens with limited dynamic range. A specific TMO f_L typically operates only on the luminance of an image, and it can be formally defined as a mathematical function converting $L_d(x) = f_L(L_w(x))$ where x represents the image coordinates.

We conducted a series of experiments to assess the best strategy to follow when tonemapping synthetic images for training neural networks. We used an HDR toolbox [3] to test various approaches, particularly analytic methods versus perceptually based TMOs.

In our case, L_w would be an image resulting from the HDR rendering of UrbanSyn*. The first analytic and straightforward approach is to use linear exposure:

$$L_d(x) = e \cdot L_w(x),$$

where e is the exposure factor, and typically

$$e = \frac{1}{L_{w,max}},$$

where $L_{w,max}$ represents the maximum value in L_w. Using that particular value for e normalizes the values to the range 0.0 to 1.0. However, many rendering software, including ours, do not apply such normalization factor, thus $e = 1$. Typically, after the application of the TMO f_L, gamma correction is applied, and each color channel is clamped to the range [0, 255] for images using 8 bits per channel. In consequence, the generated LDR images often lose information, resulting in images having overexposed areas, as shown in Fig. 8.

Automatic exposure (or best exposure) is another analytic alternative where e, in this case, is defined by the number of well-exposed pixels within a certain range of the histogram, so the luminance range chosen allows a minimum loss of information, preserving the contrast ratio of all the correct pixels present in the image.

Another option is to use a logarithmic mapping:

$$L_d(x) = \frac{log_{10}(1 + q \cdot L_w(x))}{log_{10}(1 + k \cdot L_{w,max})},$$

where the user defines q and k to tune the desired final appearance.

As perceptually based alternatives, we selected Reinhard [27], KimKautz [12], and Drago [8] as representative global TMOs, and Lischinski [15] as local (segmentation) TMO. Reinhard [27] can be run either as a global or a local TMO, but in our study we run it as a global operator, performing per-pixel scaling of the dynamic range and using the default parameters in the HDR toolbox. KimKautz [12] is based on the assumption that human visual sensitivity is adapted to the average log-luminance of the scene and that it follows a Gaussian distribution. Drago [8] uses logarithmic functions with changing bases depending on the content of the scene to exploit adaptive human perception. As representative local TMO, we run Lischinski [15], using their automatic segmentation mode inspired by the Zone System [1]. Please check Sect. 4.5 to see the results of all the TMOs proposed here.

3.9 Color transformations

Regarding color transformations, we focus on studying two main post-processing approaches avoiding the need to repeat new renders: (i) color correction without a reference image, e.g., injecting noise to promote network consistency or using white balance techniques; and (ii) color transfer between real-world references (target) and synthetic images (source).

Fig. 8 In the absence of a proper tonemapping algorithm, higher luminance values are typically compressed, exhibiting overexposed image regions (left). Contrary, using better tonemapping algorithms may unveil new details in the image (right)

Fig. 9 (Top) Color transfer using a non-optimal target, while the lower images correspond to a selected image target using the LPIPS method

To implement our color correction approach injecting noise, we operate on the L*a*b* (CIELAB) [33] color space. CIELAB is defined by three channels: one represents perceptual lightness L*, and the other two (a*, b*) encode the color. This color space allows to perform color correction without altering the luminance channel. In our study, once the image is converted to L*a*b* space, we inject some noise only in the channels a* and b* without varying the luminance channel. We empirically found that random values $\in [-5, 5]$ could be perceptually acceptable. As a second alternative for color correction without a reference, we used DeepWB [2], a neural network able to correct images wrongly white balanced in origin.

After that, we study color transfer approaches using real-world (target) reference images. UrbanSyn used an approach based on Reinhard's color transfer coupled with a random selection of the target image. We observed that that approach could easily affect the colors of the image, for example, by producing overexposed regions, as shown in Fig. 9. Thus, we identified two different aspects in which the ablation should be performed: the method to select the reference image (target) for a given rendered image (source) and the color transfer algorithm. Next, we will detail our choices for each of these aspects.

Image selection methods. To study target reference image selection approaches, we tested different perceptual metrics: the structural similarity index measure (SSIM) [39], the deep features' cosine similarity using the Contrastive Language Image Pre-Training (CLIP) [24], and the Learned Perceptual

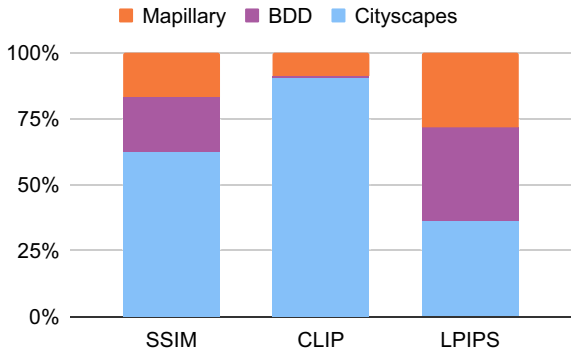

Fig. 10 Percentages of each dataset's representation utilizing the image selection methods. Random selection is not shown but imposes one-third of each dataset

Image Patch Similarity (LPIPS) [46]. Finally, we reported the results using random sampling as a baseline.

Moreover, we used the training sets of our three real-world datasets as target reference images. Due to the large processing times of calculating all the proposed metrics for a given source for all possible target images in the training sets, we decided to use a pool of just n randomly selected images ($n = 99$ in our experiments). We used one-third to represent each dataset and repeat the process for every different source image. Figure 10 shows the selected percentage from each dataset obtained by each metric. We can observe that LPIPS, a deep perceptual metric, performs a balanced selection of targets, while the other two metrics show a bias toward Cityscapes. Interestingly, CLIP almost ignores the BDD dataset, so we would not recommend it as a good choice in our settings.

Color transfer algorithms. We selected two methods capable of performing a color transfer between two images despite the similarity or not of their content, i.e., having different objects or being completely different scenes. Among those, we first chose Reinhard's color transfer [26] for being a classic approach for color transfer and widely used for domain adaptation, not just in autonomous driving but also in other fields like medical imaging [30]. As a second alternative, we chose a more recent deep color transfer using histogram analogy by Lee et al. [14]. This approach uses a neural network that transfers color between two images, establishing an analogy between their histograms. The results of all these experiments are found in Sect. 4.6.

4 Results

In this section, we will report and discuss the results of all the previously described experiments for the six studied categories. We have performed single-source experiments using DeepLabv3+ and SegFormer for each category and validated them in each of the three datasets. When promising models were found, we selected the best model in that category and fine-tuned it to test its ability to surpass the baseline results obtained only with real data. Furthermore, we performed two additional experiments, one for SegFormer and one for DeepLabv3+, joining all the improvements acquired in each category to test overall gains. Our experiments took approximately 1470 h on several high-end GPUs (mostly Nvidia A6000 with 48 GB and RTX 3090 with 24 GB).

4.1 Altering mesh geometric details

The geometry experiment findings for DeepLabv3+ are displayed in Table 3, allowing us to compare the differences between all the classes that experience a geometry reduction. We highlighted the best model using bold; from now on, we will use this convention to highlight the best model for each comparison. We can see that overall, the original version is better than the reduced one in Cityscapes and Mapillary, but not in BDD. Surprisingly, a version of UrbanSyn with reduced geometric details that is less photorealistic manages to overcome the original results when validated in BDD. We cannot suggest that reducing geometry is a good practice, even for BDD, because it is the hardest validation set of the three and produces very low results when trained either with highly detailed or low detailed geometry. So having a marginal improvement in this validation dataset is not considered relevant, especially when not accompanied by a similar trend with the other validation sets.

Moreover, analyzing the results for each class, we can observe that in Cityscapes and Mapillary, there are certain classes that also obtain some benefit from the geometry reduction. Specifically, the classes Pole and Traffic Light consistently improve their results for the three validation sets.

On the other hand, Table 4 shows the results for the same experiments but using the SegFormer segmentation model. We can observe that the tendency is consistent between the two architectures, while in Cityscapes and Mapillary, the original version is better; in BDD, the reduced version surpasses the original. We believe that this unexpected behavior could be enhanced by the fact that simplifying the shape of an object also simplifies its mask, making it more similar to human-like annotation masks that are typically used in the real-world validation sets.

We conclude that the more robustly a class is learned, the more capable the model is of overcoming a reduction in the quality of the geometric representations used in a training set. Contrarily, the classes that obtained lower IoUs require dedicating more effort to accurately represent their geometry, so the model can learn them better. Besides, there are not huge differences between the original and the reduced versions in Cityscapes and Mapillary, leading us to conclude that the

Table 3 Results for the geometry experiments using DeepLabv3+

Val.	Src	Affected Classes												
		Sidewalk	Pole	T. Light	T. Sign	Person	Rider	Car	Truck	Bus	Train	Moto	Bike	**mIoU**
City.	Red	40.17	**48.21**	**50.58**	55.58	65.90	**40.92**	74.76	10.40	**39.72**	19.92	45.74	**65.99**	50.76
	Urb.*	**42.98**	47.59	49.19	**59.40**	**67.15**	38.29	73.85	**25.92**	36.37	**23.23**	**48.41**	64.57	**52.26**
Mapi.	Red	27.55	**41.62**	**46.81**	58.28	58.87	40.68	60.33	**33.32**	20.90	6.15	51.46	48.20	46.69
	Urb.*	**30.76**	41.35	45.33	**60.24**	**63.68**	**42.18**	**68.85**	32.73	**25.38**	**12.38**	**53.28**	**51.28**	**48.99**
BDD	Red	24.46	**34.52**	**32.99**	33.54	**39.77**	**24.20**	**73.92**	**18.99**	**21.95**	**0.09**	**41.58**	**43.29**	**35.69**
	Urb.*	**27.34**	33.46	32.53	**34.34**	35.44	18.33	58.43	13.55	18.04	0.06	38.96	37.69	33.87

The results reported are from the affected classes by the geometry reduction. The table shows the comparison between the version and the original reduced for each validation set

Table 4 Results for the geometry experiments using SegFormer

Val.	Src	Affected classes												
		Sidewalk	Pole	T. Light	T. Sign	Person	Rider	Car	Truck	Bus	Train	Moto	Bike	**mIoU**
City.	Red	**52.63**	49.16	**61.10**	64.64	74.77	47.04	91.96	34.98	**67.46**	**29.52**	40.16	**70.81**	59.52
	Urb.*	45.66	**51.29**	59.85	**68.68**	**75.22**	**49.44**	**92.01**	**61.97**	58.79	29.00	**42.76**	70.17	**60.65**
Mapi.	Red	**34.13**	**47.54**	**60.20**	60.48	**72.66**	49.06	86.99	31.41	44.77	9.37	53.26	53.93	54.47
	Urb.*	32.44	46.10	58.17	**66.16**	70.53	**53.69**	**87.18**	**46.02**	**44.91**	**16.78**	**57.9**	**55.99**	**56.26**
BDD	Red	**29.70**	**37.43**	39.50	**37.66**	**51.34**	**24.34**	74.12	14.53	**45.59**	0.00	**33.06**	**32.89**	**40.50**
	Urb.*	25.94	36.32	**39.57**	37.63	47.61	23.92	**78.87**	**23.24**	44.88	0.00	32.21	31.89	40.26

The results reported are from the affected classes by the geometry reduction. The table shows the comparison between the version and the original reduced for each validation set

detail stored in the geometry is not the most important feature in a synthetic dataset.

4.2 Altering materials appearance

We can find the material's results for DeepLabv3+ in Table 5. In this experiment, the quality decrease is greater than in the geometry one, and the overall difference is heavily noticeable. That is why the results are far from the original baselines. Curiously, as in previous experiments with DeepLabv3+, there are classes that are able to surpass the original results while using this new, less realistic simplified shading. However, Car is the only class able to improve in two validation sets for both noise and random color datasets. In the results, we can observe that there are some classes more invariant to changes, such as Car, Vegetation or Road, that only suffer minor losses. Classes such as Sidewalk, Person or Traffic Sign experience instead heavy decreases due to these less realistic materials.

Table 6 shows us the material's results for the SegFormer model. The results are more consistent than those with DeepLabv3+. No single class surpassed the original results. Besides that, the robust classes that stood out were the same for both models. While Road, Vegetation, and Car resist the decrease in realism of the material appearance, the others suffer a great IoU loss.

These results show us an interesting finding: unlike domain randomization, using non-realistic renders with random colors or noise is not helping in any class. Furthermore, it is also important to observe that classes behave differently about these quality reductions. This different behavior could be explained by the information that the network learns from these classes. The robust classes, such as Road, Car, or Vegetation, are the ones that do not depend on color; for example, the road is always at the bottom of the image, and the vegetation has a very distinct shape that makes it very recognizable. These findings suggest that there are classes that depend more on material quality because they don't have other attributes from which the network could learn.

4.3 Influence of different lighting

We show in Table 7 the results comparing different lighting alternatives with DeepLabv3+. We can observe that overall, The best-performing model is the one trained with the combination of UrbanSyn* and DL. However, training with Truncated GI, surprisingly, produced the best result for BDD.

Table 8 shows the same lighting comparisons for SegFormer. As for DeepLabv3+, the best version, in this case,

Table 5 Results for the materials experiments

Val.	Source	Classes						
		Road	Sidewalk	T. Sign	Vegetation	Person	Car	**mIoU**
City.	Noise	78.49	25.65	8.85	80.71	42.44	72.19	40.54
	Color	71.83	29.99	21.23	83.95	45.14	51.70	43.86
	Urb.*	**82.35**	**42.98**	**59.40**	**87.07**	**67.15**	**73.85**	**52.26**
Mapi.	Noise	75.38	16.62	5.04	66.56	23.46	72.82	34.19
	Color	77.38	15.37	20.09	76.11	45.62	**73.85**	42.31
	Urb.*	**80.66**	**30.76**	**60.24**	**79.40**	**63.68**	68.85	**48.99**
BDD	Noise	**78.58**	15.98	7.34	64.14	21.89	**69.95**	31.16
	Color	71.62	17.67	10.41	69.02	29.47	64.79	31.49
	Urb.*	69.97	**27.34**	**34.34**	**73.40**	**35.44**	58.43	**33.87**

We report the most interesting classes as well the mean intersection over union for the UrbanSyn and the two materials versions for DeepLabv3+

Table 6 Results for the materials experiments

Val.	Source	Classes						
		Road	Sidewalk	Traffic sign	Vegetation	Person	Car	**mIoU**
City.	Noise	86.42	31.49	18.90	83.10	41.66	83.26	46.16
	Color	82.93	30.56	22.61	83.68	51.37	87.08	49.66
	Urb.*	**88.51**	**45.66**	**68.68**	**86.70**	**75.22**	**92.01**	**60.65**
Mapi.	Noise	82.83	20.75	24.51	77.06	30.58	80.36	41.12
	Color	83.78	19.26	28.70	78.64	51.86	82.62	48.26
	Urb.*	**87.07**	**32.44**	**66.16**	**80.26**	**70.53**	**87.18**	**56.26**
BDD	Noise	78.59	20.17	11.75	68.29	17.92	66.53	31.32
	Color	76.24	23.28	12.48	68.01	26.26	61.61	34.22
	Urb.*	**84.84**	**25.94**	**37.63**	**67.90**	**45.61**	**78.87**	**40.26**

We report the most interesting classes as well as the mean intersection over union for UrbanSyn and for the two materials versions for SegFormer

Table 7 Comparison of lighting experiments with DeepLabv3+

Val	Source Datasets								
	UrbanSyn*	No Part. Med.	DL	Trunc. GI	ASM & DL	HDRI & DL	Urb*. + DL	Urb*. + HDRI & DL	Urb.* + ASM & DL
City.	52.26	46.98	52.68	50.95	51.11	50.37	**53.87**	50.37	52.32
Mapi.	48.99	45.36	47.86	48.48	47.88	48.42	**49.48**	48.42	49.13
BDD	33.87	33.56	33.52	**35.18**	33.93	34.64	34.57	34.64	34.91
Mean	45.04	41.97	44.69	44.87	44.31	44.48	**45.97**	44.48	45.45

UrbanSyn* is a new rendered version as discussed in Sect. 3.1

Table 8 Comparison of the lighting experiments with SegFormer

Val	Source datasets								
	UrbanSyn*	No Part. Med.	DL	Trunc. GI	ASM & DL	HDRI & DL	Urb*. + DL	Urb*. + HDRI & DL	Urb.* + ASM & DL
City.	60.65	56.95	60.14	59.42	**61.47**	60.66	61.12	60.71	60.88
Mapi.	56.26	53.18	56.71	55.18	56.17	57.29	**57.51**	55.67	55.24
BDD	40.26	37.59	40.11	37.65	41.92	40.90	**42.21**	41.24	39.99
Mean	52.39	49.24	52.32	50.75	53.19	52.95	**53.61**	52.54	52.04

UrbanSyn* is a new rendered version as discussed in Sect. 3.1

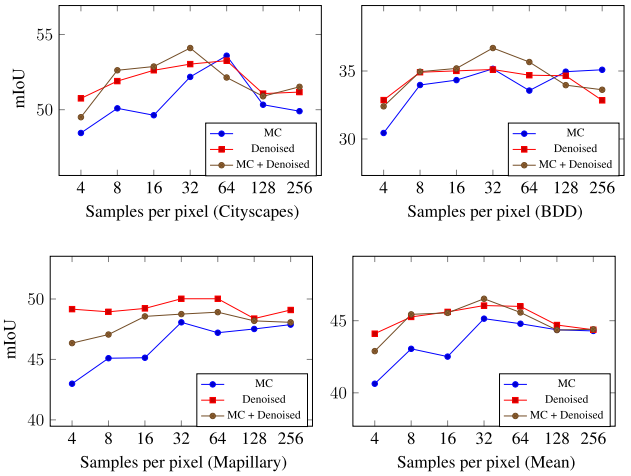

Fig. 11 DeepLabv3+ results for experiments using various sampling budgets and their denoised results. Each chart corresponds to a real-world validation set; the last one (bottom right) is their mean

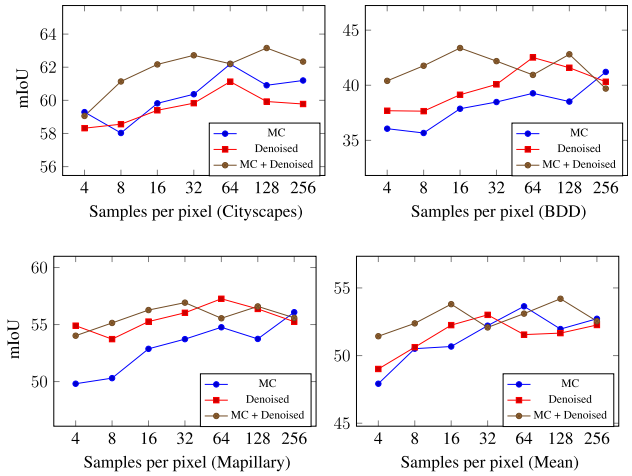

Fig. 12 SegFormer results for experiments using various sampling budgets and their denoised results. Each chart corresponds to a real-world validation set; the last one (bottom right) is their mean

it seems to be UrbanSyn* + DL. These findings suggest that the original lighting version producing a soft shadow look might not cover enough lighting diversity, and consequently, adding other types of complementary light sources might help. A general conclusion regarding participating media is that when it is visible, simulating it seems key to reducing the synth-to-real domain gap, as evidenced by performance decreases in both models when trained without simulating that effect.

After training the models, we selected the best one, which was the combination of UrbanSyn* and DL, and used it to fine-tune a model trained in real data for each of the three validation sets in SegFormer. Table 13 shows all the fine-tuning experiments performed compared to the UrbanSyn* dataset. We can observe that in the lighting category, the best model is able to surpass the UrbanSyn* fine-tuning in each validation tested. This leads us to conclude that adding a directional light set to UrbanSyn* helped improve the performance.

4.4 Effects of rendering noise

In Fig. 11, we show our first set of experiments with the DeepLabv3+ model. The figure consists of three plots: one training with noisy Monte Carlo estimations (MC), one for the MC denoised ones, and the last one presenting the results by mixing noisy and denoised images. These experiments tested different sampling budgets, as well as the combination of noisy and denoised rendered images. At first, we noticed a curious trend for DeepLabv3+, as using higher sampling budgets did not necessarily translate into better performance, obtaining pretty good results for this model already using just 32 samples per pixel. All this sparked our curiosity to test the combination of noisy and non-noisy images, which in fact ended up obtaining some of the best results overall for various validation sets.

When moving to the SegFormer model, see Fig. 12, we observed a similar trend where combinations of noisy and non-noisy seem beneficial; also, using the highest sampling budgets is not always the best option. However, in this case, 128 samples per pixel instead of 32 obtained the best results overall, including those of DeepLabv3+.

Moreover, we also choose the best model in this category for a fine-tuning. In this case, Table 13 shows us that mixing noisy and denoised renders does not particularly stand out when compared to other fine-tuned models.

4.5 Effect of tone mapping operators

Table 9 displays the results for the tone mapping operators (TMOs) experiments for DeepLabv3+. We can see that no other alternative to UrbanSyn* is able to improve its results in neither Mapillary nor BDD. However, for Cityscapes, there are some methods that manage to overcome the baseline, which are Logarithmic, Best Exposure, and Reinhard. Interestingly, these methods improve the baseline by a considerable margin of approx. +3 mIoU points but are not able to perform equally in the other two datasets. Specifically, these techniques have a significant detrimental impact on BDD.

We can refer to Table 10 to see the tone mapping operator results for SegFormer. We can observe that, contrarily to DeepLabv3+, three tonemapper operators are able to overcome the baseline in SegFormer, being Drago the best approach overall. In this case, there are no such great gains as the ones obtained in Cityscapes using DeepLabv3+, but there are neither such losses in the other validations, producing interesting results, especially in BDD and Mapillary.

We can conclude that the transformation given by a tonemapper is highly dependent on the architecture used. The linear exposure with posterior clamping used in the baseline is the best option for DeepLabv3+, while the Drago tonemapper is the best for SegFormer. We find these results especially interesting because the validation images for all three datasets are LDR obtained from real cameras that do have the same problems as the clamping could provoke: overexposed and underexposed areas. However, using a tonemapper that prevents images from having these effects is clearly superior in SegFormer.

Performing the fine-tuning, we found that the Drago version is the best for Cityscapes and the second best for Mapillary, only surpassed by the final UrbanSyn* version with all improvements applied.

4.6 Effect of color transfer approaches

Table 11 shows the experiments for the color comparison performed in DeepLabv3+. We can observe that using the dataset without any color correction produces the worst result by a great difference in all the validation sets, resulting in a more dramatic fall in Cityscapes and BDD than in Mapillary. Note that the baseline used in the UrbanSyn* paper already has one color transformation based on Reinhard using a random target selection. Adding a random color deviation significantly increases baseline performance across all datasets, which is an interesting finding. We can also observe that the color transfer methods are better than the approaches without a target in almost every validation. Specifically, There is no selection method that consistently outperforms the others in both Reinhard and Lee. Besides, random selection is the best for Lee that questions the usefulness of the other methods that are incapable of surpassing a random algorithm. Furthermore, Lee with random selection produces the best results of the comparison overall

Table 12 shows the color transformation results for SegFormer. The main difference that we observe with DeepLabv3+ is that UrbanSyn without any color transformation performs acceptably well, and it is even better than some methods in this category. Surprisingly, the color transformation methods that do not depend on target images, especially DeepWB, also perform above average. The great performance of the Random Color transformation in Mapillary method could be explained by the different sensor origins of Mapillary Vista's images. Unlike Cityscapes, each image was captured with a different camera under different lighting conditions and white balance, which results in different colors dominating the image. Using a random color deviation as in this method could help reduce the synth-to-real domain gap. The best model using SegFormer is the one trained on Reinhard's color transfer, but in this case, the best selection method is SSIM. While color transfer techniques seem to have a positive effect overall, we cannot choose a unique preferred approach in this case, as it highly depends on the validation target.

Table 14 shows the fine-tuning experiments, performed with Reinhard's SegFormer model using SSIM for image selection. We compared it with a model trained on UrbanSyn instead of UrbanSyn* since we can directly use the publicly available images. This model is only able to surpass the baseline for the BDD validation.

4.7 Optimized UrbanSyn*

We decided to combine the most promising parameters found in each category to create a version called Optimized UrbanSyn* to prove the improvement in these changes applied simultaneously. We created two versions, one for SegFormer and one for DeepLabv3+.

The DeepLabv3+ final version consists of renders with 32 samples per pixel using both the denoised version and the Monte Carlo original path tracer version. Regarding the lighting, it uses a mixture of a directional light and an HDR with a directional light. It did not use a tonemapper but a clamped range transformation as the baseline, and for the color transfer process, it leverages Lee, selecting the images randomly

The SegFormer final version consists of data rendered with 128 samples per pixel using both the denoised and the MC, and it uses the same lighting as the DeepLabv3+ version. To tonemap the images, it uses Drago, and for the color transfer process, it leverages Reinhard, selecting the images with SSIM.

Table 15 shows the comparison between these new versions and the baselines. We can observe that the new versions are always better except for the case DeepLabv3+ in Mapillary. These results prove the usefulness of the new optimized version and validate our guidelines to generate synthetic data.

5 Guidelines for synthetic image generation

Very few works have analyzed the impact of choices made during the image synthesis pipeline when using synthetic data for a specific vision task. Hopefully, this work helps alleviate this gap in the literature. This study aims to gain new knowledge about generating synthetic image datasets for training neural networks. We believe our work can bring interesting results and conclusions that could serve as guidelines for the synthetic data community, helping toward more effective and efficient synthetic datasets to train vision tasks. In the particular case of autonomous driving and semantic segmentation, we can share the following guidelines.

Rendering quality, samples, and noise. While intuitively one may think that the higher the quality of the rendered images,

Table 9 Results from the tonemapping experiments for DeepLabv3+

Datasets	Source datasets						
	UrbanSyn*	Log.	Best Exp.	Reinhard	Drago	KimKautz	Lischinski
City.	52.26	55.60	**55.67**	54.34	50.28	49.89	50.55
Mapi.	**48.99**	46.20	46.20	43.96	48.14	47.78	47.39
BDD	33.87	25.91	24.41	23.29	**34.37**	33.09	34.43
Mean	**45.04**	42.57	42.09	40.53	44.26	43.59	44.12

Notice UrbanSyn* is the new rendered version as discussed in Sect. 3.1

Table 10 Results from the tonemapping comparison experiment for SegFormer

Datasets	Source Datasets						
	UrbanSyn*	Log.	Best Exp.	Reinhard	Drago	KimKautz	Lischinski
City.	60.65	61.11	61.81	57.95	**61.94**	58.53	58.94
Mapi.	56.26	56.29	56.93	53.09	**57.22**	55.03	55.24
BDD	40.26	**42.25**	41.19	36.16	41.86	38.25	40.25
Mean	52.39	53.22	53.31	49.07	**53.67**	50.60	51.48

Notice UrbanSyn* is the new rendered version as discussed in Sect. 3.1

Table 11 Results of the color transfer approaches with the DeepLabv3+ model

	Color Transfer Img. Sel.	Source datasets										
		Or. Color	DeepWB	Rand. Color	Reinhard				Lee			
		None	None	None	Random	SSIM	CLIP	LPIPS	Random	SSIM	CLIP	LPIPS
Val. Datasets	Cityscapes	47.44	49.35	51.18	51.92	53.42	53.03	**53.37**	52.73	52.09	52.80	52.72
	Mapillary	46.83	47.74	47.52	48.97	**49.41**	47.22	48.97	48.99	48.04	46.64	47.06
	BDD	29.56	32.82	30.81	33.08	31.95	33.17	33.75	**35.19**	33.03	32.27	33.31
	Mean	41.28	43.55	43.17	44.66	44.93	44.47	45.36	**45.63**	44.39	43.90	44.36

Val. Datasets validation Datasets, *Img. sel.* Image Selection, *Or*. Color Original Color.

Table 12 Results of the color transfer approaches performed with the SegFormer model

	Color Transfer Img. Sel.	Source datasets										
		Or. Color	DeepWB	Rand. Color	Reinhard				Lee			
		None	None	None	Random	SSIM	CLIP	LPIPS	Random	SSIM	CLIP	LPIPS
Val. Datasets	Cityscapes	59.73	61.25	61.03	61.12	**63.31**	62.93	60.73	61.27	61.01	61.61	61.60
	Mapillary	56.47	57.11	57.26	56.88	56.94	56.38	55.97	56.27	56.71	57.29	**57.35**
	BDD	42.60	41.13	39.29	42.55	**42.86**	41.04	41.05	42.21	41.78	41.48	42.78
	Mean	52.93	53.16	52.03	53.52	**54.37**	53.45	52.58	53.25	53.17	53.46	53.91

Val. Datasets validation Datasets, *Img. sel.* Image Selection, *Or*. Color Original Color.

Table 13 Results from fine-tuning the best synthetic model into real data in SegFormer

	Type	Source Datasets					
		Baseline	Baseline	Lighting	Samples	TMO	All
	Pre-train	None	UrbS.*	UrbS.* + DL	128 spp	Drago	Opt. UrbS.*
	Fine-tune	Real	Real	Real	Real	Real	Real
Validation Datasets	Citys	78.47	81.37	81.43	80.96	**81.58**	81.23
	Mapi	76.08	76.41	76.65	76.26	76.79	**77.25**
	BDD	62.51	63.82	63.96	64.21	63.66	**64.29**

UrbanSyn* is the new rendered version with slight differences discussed in section 3.1. UrbS. is an acronym for UrbanSyn. and Opt. UrbS.* is the version in which we applied the most promising parameters found in each category

Table 14 Results from fine-tuning the best synthetic model into real data in SegFormer

	Type	Source Datasets		
		Baseline	Baseline	Color Transformation
	Pre-train	None	UrbanSyn	Reinhard SSIM
	Fine-tune	Real	Real	Real
Validation Datasets	Cityscapes	78.47	**81.78**	80.82
	Mapillary	76.08	**77.21**	76.31
	BDD	62.51	63.57	**63.78**

The original UrbanSyn is used here instead of UrbanSyn*, as in Table 13

Table 15 Comparison between the baselines and the final version of UrbanSyn

	Target	DeepLabv3+		SegFormer	
		UrbS.* (stdev)	Opt. UrbS*	UrbS.* (stdev)	Opt. UrbS*
Validation Datasets	Citys.	52.26 (±1.70)	56.62	60.65 (±0.58)	**64.0**
	Mapi.	48.99 (±0.62)	47.52	56.26 (±0.19)	**56.4**
	BDD	33.87 (±0.72)	34.17	40.26 (±1.14)	**42.74**
	Mean	45.04 (±1.01)	46.10	52.39 (±0.64)	**54.25**

The final version was created using the best parameters for each category, differentiating SegFormer and DeepLabv3+. UrbS. is an acronym for UrbanSyn. and Opt. UrbS.* is the version in which we applied the most promising parameters found in each category

the better the results. That is partially true, since we experimentally demonstrated that better results can be obtained with lower sampling budgets and by mixing noisy and non-noisy (denoised) images. There is a point where upgrading the sampling budget is wasteful, and perhaps because of the robustness gained by the networks when exposed to noisy data, it is not essential to generate perfect, noise-free images. While the optimal number of samples to be used may depend on the model architecture, it has been observed an overall benefit in mixing noisy and denoised images rendered with a moderate number of samples per pixel.

Participating media. A very interesting observation was that simulating the participating media could be one of the key factors to obtaining better semantic segmentation performance with synthetic datasets. For example, incorporating this effect increased the result by around +3 mIoU points consistently for both model architectures. In summary, if your domain has participating media, we highly recommend simulating it in your images to reduce the domain gap. While it might be seen as a subtle effect in some scenes, it will greatly help to reduce the domain gap.

Influence of lighting. We empirically demonstrated that better results could be obtained by simulating complementary light sources projecting both hard and soft shadows. In consequence, talking about daytime autonomous driving scenarios, lighting using a combination of HDRI environment maps and key directional lights seems recommended to lit virtual worlds.

Tone mapping operators (TMOs) and color transfer. Straightforward tonemappers and Drago, which is based on logarithmic functions and therefore is similar to those, appear to be improving the original version in some cases. It seems that tonemapping the image, avoiding overexposed areas, could be a beneficial post-process. Color transfer methods that use a target seem to be consistently superior to other color transformation methods and have proven to be far above the baseline. However, the results did not clarify the best strategy for the target selection, as no metric used for selection has proven to be better than the others.

Geometric modeling and material fidelity. Regarding geometry modeling, the higher the detail, typically the better, especially for modern segmentation architectures, such as SegFormer. However, a certain relaxation of the modeling accuracy seems to be better tolerated by certain classes, those that usually produce a good segmentation performance and occupy similar positions in the 3D world in every resulting image. For the most challenging classes, we highly recommend having your most detailed models; otherwise, those classes will suffer more from the lack of detail. Regarding material appearance, we did not observe a real benefit in training with other than realistic material shading; thus, we find agreement with previous work in recommending using realistic materials as much as possible.

6 Conclusions

This work presents an ablative study analyzing the impact that various factors involved in the image synthesis pipeline might have in the resulting generated images and consequently on the neural networks trained with them to solve downstream computer vision tasks, such as seman-

tic segmentation. The aspects under study include geometry modeling, material appearance, realistic lighting, rendering noise, tone mapping operators, and color transfer techniques. Our study is focused on analyzing the influence of all these aspects using the specific case study of UrbanSyn, a synthetic image dataset generated using computer graphics belonging to the autonomous driving domain that has been extensively tested for semantic segmentation. In our experiments, each proposed model was trained on the corresponding synthetic data and later validated in three real-world autonomous driving datasets: Cityscapes, Mapillary Vistas, and BDD100K.

In this work, we proposed guidelines according to the impact observed on the performance of neural network models trained with the synthetic image data generated. Our ablative study required nearly a hundred experiments and took approximately 1470 hours on several high-end GPUs. As an important limitation of this study, we should mention the difficulty of generalizing conclusions among different hyper-parameter configurations. In future work, it would be interesting to expand the ablation study to other computer vision tasks to verify if the conclusions achieved for semantic segmentation may also translate to other tasks.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00371-025-03811-1.

Acknowledgements We would like to thank the UrbanSyn team for providing us access to their original material and for their efforts in creating their dataset. The authors thank J.L. Gomez for valuable help while setting up our semantic segmentation experiments. Funding for open access charge: Universidade da Coruña/CISUG.

Author Contributions Conceptualization: M.S and O.A and J.I ; Investigation: M.S ; Project administration: J.I ; Software: M.S and A.S ; Resources: J.I and A.L ; Supervision: J.I and A.L ; Writing - original draft: M.S ; Writing - review & editing: J.I.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This work has been supported by Spanish grants Ref. PID2020-115734RB-C22 and PID2020-115734RB-C21, both funded by MCIN/AEI/10.13039/501100011033.
Manuel Silva acknowledges Xunta de Galicia (Consellería de Cultura, Educacion e Universidade) for his predoctoral grant (ED481A-2023-191). Antonio M. López acknowledges the financial support to his general research activities given by ICREA under the ICREA Academia Program and the support of the Generalitat de Catalunya CERCA Program and its ACCIO agency to CVC's general activities. J.A. Iglesias-Guitian also acknowledges the UDC-Inditex InTalent program, the Ministry of Science and Innovation (AEI/ RYC2018-025385-I) and Xunta de Galicia (ED431F 2021/11).

Data Availability We generated datasets and analyzed them in this work. Trained models are publicly shared in the supplemental material.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Adams, A.: The Print: The Ansel Adams Photography Series 3. Little, Brown (1981)
2. Afifi, M., Brown, M.S.: Deep white-balance editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
3. Banterle, F., Artusi, A., Debattista, K., Chalmers, A.: Advanced High Dynamic Range Imaging (2nd Edition). AK Peters (CRC Press), Natick (2017)
4. Baranchuk, D., Rubachev, I., Voynov, A., Khrulkov, V., Babenko, A.: Label-efficient semantic segmentation with diffusion models. arXiv preprint arXiv:2112.03126 (2021)
5. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 801–818 (2018)
6. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
7. Dodge, S., Karam, L.: Understanding how image quality affects deep neural networks. In: 2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX), pp. 1–6. IEEE (2016)
8. Drago, F., Myszkowski, K., Annen, T., Chiba, N.: Adaptive Logarithmic Mapping For Displaying High Contrast Scenes. Computer Graphics Forum (2003)
9. Gu, J., Iglesias-Guitian, J.A., Moon, B.: Neural james-stein combiner for unbiased and biased renderings. ACM Trans. Graphics (TOG) **41**(6), 1–14 (2022)
10. Gómez, J.L., Silva, M., Seoane, A., Borrás, A., Noriega, M., Ros, G., Iglesias-Guitian, J.A., López, A.M.: All for one, and one for all: Urbansyn dataset, the third musketeer of synthetic driving scenes. arXiv preprint arXiv:2312.12176 (2023)
11. Iglesias-Guitian, J., Ros, G., Kokkevis, V., Alvarez, J., Lee, Y., Slusallek, P.: Computer graphics for autonomous vehicles. In: ACM SIGGRAPH Frontiers Workshop. ACM SIGGRAPH (2019)
12. Kim, M.H., Weyrich, T., Kautz, J.: Modeling human color perception under extended luminance levels. ACM Trans. Graph. **28**(3) (2009)
13. Kumar, V.R., Klingner, M., Yogamani, S., Milz, S., Fingscheidt, T., Mader, P.: Syndistnet: Self-supervised monocular fisheye camera distance estimation synergized with semantic segmentation for autonomous driving. In: 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 61–71 (2021)
14. Lee, J., Son, H., Lee, G., Lee, J., Cho, S., Lee, S.: Deep color transfer using histogram analogy. Vis. Comput. **36**(10–12), 2129–2143 (2020)

15. Lischinski, D., Farbman, Z., Uyttendaele, M., Szeliski, R.: Interactive local adjustment of tonal values. ACM Trans. Graph. **25**(3), 646–653 (2006)
16. Liu, Z., Lian, T., Farrell, J., Wandell, B.A.: Neural network generalization: The impact of camera parameters. IEEE Access **8**, 10443–10454 (2020)
17. Lv, C., Zhang, D., Geng, S., Wu, Z., Huang, H.: Color transfer for images: a survey. ACM Trans. Multimedia Comput. Commun. Appl. (2023)
18. Mittal, S., Niemeijer, J., Schäfer, J.P., Brox, T.: Best practices in active learning for semantic segmentation. In: DAGM German Conference on Pattern Recognition, pp. 427–442. Springer (2023)
19. Movshovitz-Attias, Y., Kanade, T., Sheikh, Y.: How useful is photo-realistic rendering for visual learning? In: Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14, pp. 202–217. Springer (2016)
20. Mukherjee, R., Bessa, M., Melo-Pinto, P., Chalmers, A.: Object detection under challenging lighting conditions using high dynamic range imagery. IEEE Access **9**, 77771–77783 (2021)
21. Neuhold, G., Ollmann, T., Bulò, S.R., Kontschieder, P.: The mapillary vistas dataset for semantic understanding of street scenes. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 5000–5009 (2017)
22. Paulin, G., Ivasic-Kos, M.: Review and analysis of synthetic dataset generation methods and techniques for application in computer vision. Artificial Intelligence Review pp. 1–45 (2023)
23. Pepik, B., Benenson, R., Ritschel, T., Schiele, B.: What is holding back convnets for detection? In: Pattern Recognition: 37th German Conference, GCPR 2015, Aachen, Germany, October 7-10, 2015, Proceedings 37, pp. 517–528. Springer (2015)
24. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021)
25. Rangnekar, A., Kanan, C., Hoffman, M.: Semantic segmentation with active semi-supervised learning. In: 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 5966–5977 (2023)
26. Reinhard, E., Adhikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Comput. Graphics Appl. **21**(5), 34–41 (2001)
27. Reinhard, E., Stark, M., Shirley, P., Ferwerda, J.: Photographic tone reproduction for digital images. ACM Trans. Graph. **21**(3), 267–276 (2002)
28. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games. In: B. Leibe, J. Matas, N. Sebe, M. Welling (eds.) European Conference on Computer Vision (ECCV), *LNCS*, vol. 9906, pp. 102–118. Springer (2016)
29. Ros, G., Sellart, L., Materzyska, J., Vázquez, D., López, A.: The SYNTHIA dataset: a large collection of synthetic images for semantic segmentation of urban scenes (2016)
30. Roy, S., Kumar Jain, A., Lal, S., Kini, J.: A study about color normalization methods for histopathology images. Micron **114**, 42–61 (2018)
31. Schlachter, K., DeFanti, C., Herscher, S., Perlin, K., Tompson, J.: Beyond photo realism for domain adaptation from synthetic data. arXiv preprint arXiv:1909.01960 (2019)
32. Schwonberg, M., Niemeijer, J., Termöhlen, J.A., Schmidt, N.M., Gottschalk, H., Fingscheidt, T., et al.: Survey on unsupervised domain adaptation for semantic segmentation for visual perception in automated driving. IEEE Access **11**, 54296–54336 (2023)
33. Standard, C., et al.: Colorimetry-part 4: Cie 1976 l* a* b* colour space. International Standard pp. 2019–06 (2007)
34. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 23–30 (2017)
35. Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., Birchfield, S.: Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 969–977 (2018)
36. Tsirikoglou, A., Eilertsen, G., Unger, J.: A survey of image synthesis methods for visual machine learning. In: Computer Graphics Forum, vol. 39, pp. 426–451. Wiley Online Library (2020)
37. Tsirikoglou, A., Kronander, J., Wrenninge, M., Unger, J.: Procedural modeling and physically based rendering for synthetic data generation in automotive applications. arXiv preprint arXiv:1710.06270 (2017)
38. Wang, X., Ma, H., You, S.: Deep clustering for weakly-supervised semantic segmentation in autonomous driving scenes. Neurocomputing **381**, 20–28 (2020)
39. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
40. Wang, Z., Jiang, Y., Zheng, H., Wang, P., He, P., Wang, Z., Chen, W., Zhou, M., et al.: Patch diffusion: Faster and more data-efficient training of diffusion models. Adv. Neural Inf. Process. Syst. **36** (2024)
41. Wrenninge, M., Unger, J.: Synscapes: A photorealistic synthetic dataset for street scene parsing. arXiv:1810.08705 (2018)
42. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2 (2019)
43. Xie, B., Yuan, L., Li, S., Liu, C.H., Cheng, X.: Towards fewer annotations: Active learning via region impurity and prediction uncertainty for domain adaptive semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8068–8078 (2022)
44. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: simple and efficient design for semantic segmentation with transformers. In: Proceedings of the 35th International Conference on Neural Information Processing Systems, NIPS '21. Curran Associates Inc., Red Hook, NY, USA (2024)
45. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
46. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
47. Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J., Jin, H., Funkhouser, T.: Physically-based rendering for indoor scene understanding using convolutional neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

Manuel Silva is a Ph.D. student at the Universidade da Coruña (UDC). He earned a bachelor's degree in audiovisual communication with a keen interest in 3D computer graphics. Before starting his Ph.D. journey at UDC, Manuel interned as a 3D technician with the Computer Vision Center (CVC) in Barcelona and worked as a technical research assistant for an autonomous driving project at UDC. His research is currently focused on leveraging and understanding how synthetic data can enhance the performance of neural networks.

Antonio Seoane is an educator and researcher specializing in computer graphics, video game design, and animation at the Universidade da Coruña (UDC). His research focuses on real-time computer graphics and immersive virtual environments, with a particular emphasis on developing real-time algorithms for the interactive rendering of large 3D terrain models. As a teacher, Antonio guides his students through the foundations of video game design, 3D animation, and VFX technology.

Omar A. Mures M.Sc. in high-performance computing, is a researcher and instructor at the University of A Coruña (UDC), Spain. As a Computer Architecture Group (GAC) member, he specializes in applying high-performance computing to various domains such as artificial intelligence (AI), computer vision (CV), and computer graphics (CG). His research interests mainly focus on cutting-edge topics such as synthetic data generation, segmentation, detection, data analysis, and real-time rendering. He has also worked with several industry partners like Cinfo, S.L.

Antonio M. López leads the autonomous driving laboratory at the Computer Vision Center (CVC) in the Universitat Autònoma de Barcelona (UAB). His research focuses on the intersection of computer vision, simulation, machine learning, and autonomous driving. He is one of the creators of the SYNTHIA dataset and the CARLA simulator. Antonio collaborates with industry partners to bring cutting-edge techniques to the field of autonomous driving. He has been awarded the Catalan ICREA Academia program for his research contributions.

Jose A. Iglesias-Guitian is a Ramón y Cajal researcher at CITIC, Universidade da Coruña (UDC), co-financed by the InTalent program of UDC and Inditex. His research field spans visual computing and computer graphics, focusing on rendering, virtual reality, medical visualization, and synthetic data generation for AI. José was also an associate researcher at Disney Research in the UK and a Marie Curie fellow at CVC and UAB. He received his Ph.D. degree in electronics and computer engineering while working at the CRS4 research center in Italy.