





Article

Quantifying the Simulation–Reality Gap for Deep Learning-Based Drone Detection

Tamara Regina Dieter ^{1,2,*} , Andreas Weinmann ³ , Stefan Jäger ²  and Eva Brucherseifer ^{1,2} ¹ Department of Computer Science, University of Applied Sciences Darmstadt, 64295 Darmstadt, Germany² Institute for the Protection of Terrestrial Infrastructures, German Aerospace Center (DLR), 53757 Sankt Augustin, Germany³ Department of Mathematics, University of Applied Sciences Darmstadt, 64295 Darmstadt, Germany

* Correspondence: tamara.dieter@dlr.de

Abstract: The detection of drones or unmanned aerial vehicles is a crucial component in protecting safety-critical infrastructures and maintaining privacy for individuals and organizations. The widespread use of optical sensors for perimeter surveillance has made optical sensors a popular choice for data collection in the context of drone detection. However, efficiently processing the obtained sensor data poses a significant challenge. Even though deep learning-based object detection models have shown promising results, their effectiveness depends on large amounts of annotated training data, which is time consuming and resource intensive to acquire. Therefore, this work investigates the applicability of synthetically generated data obtained through physically realistic simulations based on three-dimensional environments for deep learning-based drone detection. Specifically, we introduce a novel three-dimensional simulation approach built on Unreal Engine and Microsoft AirSim for generating synthetic drone data. Furthermore, we quantify the respective simulation–reality gap and evaluate established techniques for mitigating this gap by systematically exploring different compositions of real and synthetic data. Additionally, we analyze the adaptation of the simulation setup as part of a feedback loop-based training strategy and highlight the benefits of a simulation-based training setup for image-based drone detection, compared to a training strategy relying exclusively on real-world data.

Keywords: drone detection; deep neural networks; synthetic data; simulation–reality gap



Citation: Dieter, T.R.; Weinmann, A.; Jäger, S.; Brucherseifer, E.

Quantifying the Simulation–Reality Gap for Deep Learning-Based Drone Detection. *Electronics* **2023**, *12*, 2197. <https://doi.org/10.3390/electronics12102197>

Academic Editors: Egils Ginters, Zhenhua Guo, Kawa Nazemi and Michael Bažant

Received: 25 February 2023

Revised: 1 May 2023

Accepted: 9 May 2023

Published: 11 May 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Protecting private or safety-critical infrastructures against attacks (e.g., espionage) by unmanned aerial vehicles (UAVs or drones) is an important security issue given the increasing prevalence of UAVs in commercial and private sectors (e.g., [1–3]). Therefore, there is also an increasing need for robust detection systems capable of reliably identifying and locating unauthorized UAVs.

Conventional drone detection systems typically rely on single sensing technologies, such as audio [4], optical [5], radio frequency [6] or radar sensors [7], or are built on compositions of these technologies [8] to mitigate sensor-specific limitations (e.g., the uncertainty of acoustic sensors in the case of high noise levels, or the ineffectiveness of RF sensors in the case of autonomous drones [9]). However, a sensor’s practical suitability is also determined by its price and availability. Optical sensor- or camera-based solutions are therefore of great interest, given their cost efficiency and broad availability. Along with the ease of sensor data readability and implementation, the presence of cameras in conventional security systems is yet another benefit in favor of camera-based approaches.

The advantages of optical sensors in combination with the rapid advances in computer vision (CV) have especially accelerated the research and development of image-based detection techniques built on deep learning (DL) algorithms [5,8,10–14]. Despite

the promising performance of modern object detectors (e.g., region-based convolutional neural networks (R-CNNs) [15–17], single-shot multibox detector (SSD) [18], or you only look once (YOLO) [19]), developing robust drone detection algorithms has proven to be a challenging task which necessitates large amounts of diversified and annotated data.

As data availability remains limited and the cost of acquiring real data remains high, using synthetic data to train DL models represents a popular approach in drone detection [13,20–25] and other application domains (e.g., autonomous driving [26–30]). In addition to their inexpensive generation, utilizing synthetic data offers the potential of overcoming real-world restrictions (e.g., imposed by no-fly zones), enhancing data accessibility, and achieving greater data diversification [31]. A wide variety of different generation techniques (e.g., domain randomization (DR) or general adversarial networks (GANs)) enable the rapid creation of domain-specific data. In particular, the use of game engine-based simulations (e.g., via Unreal [32] or Unity [33]) in three-dimensional environments is a promising approach as shown by various studies across different domains [23,24,27–29,34]. It offers the potential of representing complex illumination (e.g., inter-reflection between objects and scenes or between objects themselves [35]), occlusions, object dynamics (e.g., drone movements), and other environmental conditions in a physically precise manner, increasing realism and diversification. Furthermore, the precise control over environmental factors enables the accurate representation of real-world scenarios and facilitates the creation of automatically annotated data.

However, transferring systems (e.g., object detectors) trained exclusively on simulated data to real-world applications often results in performance degradation caused by the so-called simulation–reality gap (also known as simulation-to-reality gap [36], reality gap [37], domain gap [20], or Sim2Real problem [21,25]). The simulation–reality gap is highly data-dependent and thus directly affected by the quality of the underlying data (both synthetic and real), making its quantification and mitigation a controversial and application-dependent topic of ongoing research [20–23,26–28,30,38].

In the present work, we investigate the applicability of physically realistic (game engine-based) simulations to generate synthetic data for DL-based drone detection in a static camera setting (e.g., a typical surveillance scenario). We assess the effectiveness of these simulations for real-world applications, explore techniques for addressing the simulation–reality gap, and propose a feedback loop-based training strategy. Our findings indicate that integrating small shares of real-world data during the training process, in conjunction with the generated synthetic data, mitigates the impact of the simulation–reality gap, leading to enhanced performance compared to training exclusively with real-world data. A detailed overview of our contributions is given in Section 1.2.

1.1. Related Work

Synthetic Data Generation. Current approaches for generating synthetic data for CV tasks range from fairly basic techniques (e.g., two-dimensional copy-and-paste algorithms [22,39,40]) to more elaborate simulations in three-dimensional environments [28,29], prioritizing either realism or randomization.

Regardless of the application domain, the majority of approaches rely on data synthesis by means of domain randomization (DR) as shown in [13,20–25,27,41]. Based on the idea that non-realistic randomization enhances the learning of essential features, two- or three-dimensional objects of interest (e.g., drones or cars) are typically placed randomly on selected (two-dimensional) backgrounds (e.g., images [23], video sequences [22] or environmental maps [25]). To synthesize data for drone detection (in a surveillance setting), Chen et al. [22] propose a model-based augmentation technique by randomly positioning two-dimensional drone models on video sequences of real indoor and outdoor scenes. In the work of Akyon [20] and Rozantsev [41], on the other hand, three-dimensional drone models are employed (e.g., coarse computer aided design (CAD) models [41]). Similar techniques are presented by Symeonidis et al. [24] and Özfuttu [23]. Even though their generation pipelines are game engine-based, they do not employ three-dimensional environments.

Instead, they use two-dimensional backgrounds composed of real images belonging to different categories (e.g., sky, urban and nature) [23] or projections of real video data [24]. Symeonidis et al. additionally include three-dimensional models of other flying objects (e.g., birds) to be positioned randomly in the scene. However, a primary limitation of fusing three-dimensional drone models with real background images or video data is the inability to regulate environmental variables (e.g., lighting), which can lead to inconsistencies and unnatural representations (e.g., drone models are not properly aligned or lit to match the background). To include more realistic illumination and material properties, Peng et al. [13] and Barisic et al. [25] deploy environmental maps for background representation. While [13] aims at photo-realistic rendering of drone images, [25] focuses on improving the robustness of aerial object detectors (specifically for UAV-to-UAV detection) through shape-based representations induced by texture randomization. However, the application of two-dimensional environmental maps does not account for occlusions or shadow projections caused by background objects and their effects on a drone's appearance, unlike simulations in three-dimensional environments. Approaches based on three-dimensional environments instead of two-dimensional backgrounds as in the work of Marez et al. [21] (combining game engine-based simulations in three-dimensional environments with DR) are rare in the context of drone detection.

In other research areas (e.g., autonomous driving), game engine-based approaches using three-dimensional environments are far more established and provide promising results (cf. [27,30]). This also applies to video game-based generation techniques (e.g., Grand Theft Auto [26,28,29]), albeit lacking the ability to easily customize the simulation environment (which is essential for handling the complexity associated with real-world scenarios). Recent studies in the field of autonomous driving also show that context-aware data generation techniques can lead to more suitable training data (cf. [26,30,35]), which in turn improves the detection quality. This contradicts the principle idea of DR (most commonly used in drone detection as in the work of Marez et al. [21]) and advocates a stronger emphasis on realism. Ensuring the fidelity of synthetic data to the real world is crucial, as the use of data that do not accurately represent the physical world can significantly impact the performance of detection systems, leading to a discrepancy between simulation and reality.

Simulation–Reality Gap. The gap between simulation and reality is typically quantified by evaluating models trained on synthetic data against real-world data [21,25] (also known as zero-shot sim-to-real transfer learning capability [34,42]) and comparing them with models trained on real data only [13,20,26,28,30] (cf. e.g., Figure 1). Popular approaches for narrowing the gap, such as fine-tuning models on real data which are pre-trained exclusively on synthetic data [21,23,27] or following mixed-training strategies [13,20,22–24,27,28], are often considered as well. In the above references, high data dependencies and differences in the experimental design lead mostly to controversial results. Except for the work of Johnson-Roberson et al. [26], where models trained on synthetic data perform better than models trained on real data for a sufficiently large training dataset, the majority of research indicates that a combination of real and synthetic data (either through fine-tuning or mixed-data training) yields the best model performance in terms of mean average precision (mAP) or recall [13,20–23,27,28]. In the context of drone detection, Akyon et al. [20] and Peng et al. [13] show that augmenting real data with an optimal subset of synthetic data can improve the performance on real data as opposed to models trained exclusively on one or the other. Similar results are reported in [22–24,28,30]. Nowruz et al. [38], on the other hand, show that fine-tuning models pre-trained on synthetic data with small amounts of real data yields better detection results than mixed-data training in terms of mAP. This is also supported by the results of Marez et al. [21] and Özfuttu [23], who use synthetic data for pre-training drone detection models while fine tuning them based on diverse shares of real data.



Figure 1. Assessment of the detection accuracy. A comparison of the detection results on real data obtained with model M_S trained solely on synthetic data (**left**) and model M_R trained on real data only (**right**). Both models exhibit a comparable level of confidence in detecting the drone.

Apart from different experimental designs, there are also significant differences in the composition of synthetic and real data used to quantify and bridge the simulation–reality gap (both for fine-tuning and mixed-data training), with real-data shares ranging from 0.4% to 50% [20,23,24,28,38,39]. In addition, the comparability of research results is further complicated by the use of different evaluation metrics. Despite the predominant use of mAP, it is often determined for different intersection over union (IoU) thresholds (e.g., 0.5 [20,23–25,27,30], 0.7 [13,26] or 0.75 [21]) or as average over varying thresholds ranging from 0.5 to 0.95 in increments of 0.05 [21,23,24,38]. Less frequently used metrics are precision, recall [23,36], average recall (AR) [21], and mean average recall (mAR) [21,24]. The work of Reway et al. [36] even introduces a simulation–reality gap score comprising multiple performance measures.

Other (less frequently used) approaches to traverse the gap between simulation and reality are domain adaptation (e.g., via GANs [43]) and simulation parameter optimization [37].

1.2. Contribution and Outline

In this paper, we investigate the applicability of synthetic data obtained by physically realistic simulations based on three-dimensional environments in the context of DL-based drone detection and make the following contributions:

- We present a novel game engine-based simulation approach (built on Unreal 4.25 and Microsoft AirSim [44]) to generate synthetic data for image-based drone detection in a static camera setting that exploits the full potential of three-dimensional environments (as opposed to randomization techniques against two-dimensional backgrounds) and quantify the respective simulation–reality gap.
- We quantitatively and qualitatively evaluate the effectiveness of established techniques for bridging the gap between simulation and reality, systematically considering different compositions of real and synthetic data (inspired by recent research). In this context, we also consider a feedback loop-based training strategy focusing on adapting and extending the simulation setup to increase realism in synthetic data generation and thus narrow the gap.
- We highlight the benefits of a simulation-based training setup for image-based drone detection built on DL algorithms over training setups relying solely on real-world data. In particular, we evaluate the robustness of the resulting models against real-world challenges, including variations in weather and lighting conditions (a necessity in real-world applications).

The remainder of the paper is organized as follows: First, we describe the experimental setup and methodology behind our analyses in Section 2, including the underlying datasets, the employed object detector, and the considered evaluation metrics. In Section 3, we present the main results of our analyses and provide a brief discussion to contextualize the findings. Finally, we draw conclusions and provide an outlook for future work in Section 4.

2. Materials and Method

In this chapter, we present the comprehensive methodology utilized for analyzing the simulation–reality gap and assessing the efficacy of different training strategies to advance the development of more sophisticated and accurate simulation models. This includes the characterization of the datasets in Section 2.1, the specification of the models in Section 2.2, the definition of the evaluation metrics in Section 2.3, and the description of the experimental design in Section 2.4.

2.1. Data

In our analysis, we employ data from both real and virtual measurement campaigns. The real-world measurement campaigns are conducted in an urban environment characterized by medium-height buildings and medium to low vegetation density, using a Yuneec Mantis G drone (a cost-effective alternative with strong resemblance to commonly distributed DJI drone models [45]). The data are captured by a static ground-mounted Basler acA200-165c camera, equipped with two lenses of varying focal lengths (25 mm and 8 mm) from three distinct camera positions, including varying illuminations. The performed measurement campaigns lead to five real-world datasets (R1–R5) consisting of RGB images with a resolution of 2040×1086 pixels (cf. Table 1 and Figure 2), where R3 to R5 are for evaluation purposes only.

Table 1. Overview of real and synthetic training and validation datasets.

Dataset	Type	Image Count				Camera	
		Train	val	Test	Total	no. pos.	Focal Length
R1	real	7524	2508	2508	12,540	2	8 mm
R2	real	3834	1279	1278	6391	2	25 mm
R3	real	1784	595	595	2974	2	25 mm
R4	real	2099	700	700	3499	1	8 mm
R5	real	3246	1082	1082	5410	1	8 mm
S1	synth.	10,446	3483	3483	17,412	5	-
S2	synth.	423	143	143	709	2	-
S3	synth.	969	324	324	1617	1	-



Figure 2. Visualization of real and synthetic dataset samples. Illustration of data and camera perspectives for the datasets (S1–S2 and R1–R5) listed in Table 1.

The virtual measurement campaigns are performed using the game engine-based data generation pipeline presented in [31]. The pipeline comprises four principle components: (i) the Unreal Engine [32], which is accountable for providing object information, sensor data, and visualizing the simulation, (ii) the data generation module, enabling the parallel acquisition of automatically labeled sensor data, (iii) the drone control module, responsible for the navigation of drones, and (iv) Microsoft AirSim [44], connecting all aforementioned components (for more details, refer to [31]). Our synthetic data generation process aims to achieve high diversification of backgrounds with respect to object localization and type,

while also obtaining extensive variation in drone appearance through different poses and alignments. We cover a wide range of drone sizes and positions within the camera's field of view as well as various color schemes.

To approximate the real data as closely as possible, we employ virtual environments exhibiting similar characteristics to the environment underlying the generation of R1–R5 (particularly with respect to the selected camera positions and orientations depicted in Figure 2). Specifically, we use the commercially available environment Urban City [46] and the City Park Environment Collection [47], along with a variety of drone models (similar to the DJI Tello Ryze, the DJI Phantom, and the UDI Quadcopter). In the Urban City environment, data collection is performed using five distinct camera positions and orientations (cf. Figure 2, S1) to increase dataset diversity, while data collection in the City Park environment is only based on two (cf. Figure 2, S2). Given the collected data (i.e., RGB images sized 2040×1080 pixels), we create the synthetic datasets S1 (based on Urban City) and S2 (based on City Park), where S2 is used for evaluation purposes only (for details, refer to Table 1). For dataset S3, measurement campaigns are performed analogously to S1, albeit with an adapted simulation environment (for more details see Section 2.4).

Both real and synthetic datasets contain approximately 7–8% background images, representing scenarios where no drone is visible within the frame. Detailed information on the size and composition of each dataset (real and synthetic) is given by Table 1.

2.2. Deep Learning Model

In the domain of drone detection, single-stage detectors, such as YOLO, have become the predominant approach for real-time detection as demonstrated by several prior studies (cf. [48–52]). In this research, we use the high-performing state-of-the-art detector YOLOv5 [53] (similar to [49]) which was released shortly after YOLOv4 [54] and builds on the work of the original YOLO authors [19]. Its combination of high performance, efficiency, flexibility, and open-source nature makes YOLOv5 a promising choice for scientific studies on object detection tasks. Specifically, we use [53] YOLOv5l [53] which offers a good trade-off between model complexity, training time, and detection performance [31]. To train our models, we use COCO weights [55] for weight initialization, a default input size of 640×640 pixels, and a training time of 300 or 30 epochs depending on the training strategy (i.e., 30 epochs for fine-tuning, otherwise 300, cf. Section 2.4). All other hyper-parameters are set according to their specified default values (cf. [53]).

2.3. Evaluation Criteria

The performance of an object detector is typically measured by the mean average precision (mAP) for an IoU threshold of 0.5 or the average of multiple IoU thresholds ranging from 0.5 to 0.95 in increments of 0.05 (cf. [56] for more details). The mAP is a compromise performance measure including different aspects of object detection. However, ensuring the reliable protection of private or safety-critical areas against drones in particular requires the detection of all incoming drones at a high confidence level, as well as the minimization of false alarms. This can be expressed by the false negative rate (FNR) defined by

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}} \quad (1)$$

(i.e., the proportion of predicted negatives that are incorrectly inferred to be negatives), and the false discovery rate (FDR) represented by

$$\text{FDR} = \frac{\text{FP}}{\text{FP} + \text{TP}} \quad (2)$$

(i.e., the proportion of predicted positives that are incorrectly inferred to be positives). Here, the false negatives are denoted by the symbols FN, the false positives by FP, and the true positives by TP [57], respectively. It is important to note that in the case of FDR and FNR, low values are indicative of good performance, whereas for mAP, high values are desirable.

2.4. Experimental Design

To investigate the suitability of synthetic data generated by physically realistic simulations exploiting three-dimensional environments for training DL-based drone detectors, we conduct three main experiments using the aforementioned DL model, datasets, and quality measures (cf. Sections 2.1–2.3).

Experiment 1: Quantifying the Simulation–Reality Gap for Models Trained Exclusively on Synthetic Data. The first experiment focuses on exploring the potential use of models trained exclusively on synthetic data for drone detection in real-world applications. Following the state of the art, we train a YOLOv5l model (denoted by M_S , cf. Table 2) based on the synthetic dataset S1 (cf. Table 1) according to the specification in Section 2.2. We then evaluate the resulting model M_S on a variety of real-world datasets with visual similarity to the training data (to test its zero-shot sim-to-real transfer learning capability [34]), and on a synthetic dataset not derived from the training data distribution. More precisely, we consider the test and validation sets of R1–R5 and S2, respectively (cf. Table 1). Apart from quantifying the simulation–reality gap using mAP (cf. Section 2.3), we also perform a detailed analysis of false alarms and missed detections (using FNR and FDR) to identify potential background distractors.

Table 2. A comprehensive overview of models explored in Experiments 1–3.

Model	Experiment	Description
M_S	1	Model M_S is trained from scratch for 300 epochs using only the synthetic dataset S1, without the inclusion of real-data shares.
M_F	2	Model M_F is generated by fine-tuning the pre-trained model M_S for 30 epochs using a range of different real-data shares (0.001, 0.002, 0.003, 0.004, 0.0005, 0.01, 0.05, 0.1, 0.5, and 1) from datasets R1 and R2.
M_M	2	The model M_M is the result of using mixed-training strategies, where the model is trained from scratch on a combination of synthetic and real data for 300 epochs. Specifically, the synthetic dataset S1 and varying shares of the real datasets R1 and R2 are used (in analogy to M_F).
$M_{S'}$	2	Model $M_{S'}$ is obtained by fine-tuning the pre-trained model M_S on the synthetic dataset S3 for 30 epochs as part of the feedback loop-based training strategy.
$M_{F'}$	2	Model $M_{F'}$ is an intermediate fine-tuned version of the pre-trained model $M_{S'}$, which was initially trained on dataset S1 and fine-tuned on dataset S3 using a feedback loop-based training approach. Similar to the training process of M_F , fine-tuning for $M_{F'}$ is performed for 30 epochs and involves varying shares (i.e., 0.001, 0.002, 0.003, 0.004, 0.0005, 0.01, 0.05, 0.1, 0.5, and 1) of the real datasets R1 and R2.
M_R	3	Model M_R is trained from scratch for 300 epochs using only the real datasets R1 and R2, without the inclusion of synthetic-data shares.

Experiment 2: Evaluating the Effectiveness of Established Techniques in Bridging the Simulation–Reality Gap. The second experiment aims to evaluate the effectiveness of established techniques for bridging the gap between simulation and reality. This investigation entails a comprehensive evaluation (quantitatively and qualitatively) of the widely employed approaches of fine tuning and mixed-data training (cf. Section 1.1), and the analysis of a novel feedback loop-based training strategy. For fine tuning, we employ a model pre-trained on synthetic data (specifically M_S , cf. Table 2) as a baseline and progressively consider increasing real-data shares for supplementary training, starting from 0.001 and advancing to one (cf. model M_F , Table 2). For analyzing mixed-training strategies, we contemplate training datasets that comprise both real and synthetic data. Specifically, we explore the enrichment of dataset S1 with varying proportions of R1 and R2 (cf. model M_M , Table 2). In this context, M_F denotes the models obtained via fine tuning, whereas

M_M refers to the models obtained through mixed-data training. The datasets S1 and S2 as well as R1–R5 are used for evaluation purposes.

Drawing upon the findings of Experiment 1 (particularly in terms of R1), we further investigate the impact of a feedback loop-based training strategy. Here, the simulation environment Urban City is modified to incorporate objects introducing errors in the evaluation of M_S against real data. Specifically, the traffic sign, identified as the source of high false positive rates (cf. Section 3.1), is incorporated into the simulation environment as depicted in Figure 3. The modified simulation environment serves as a basis for generating additional synthetic data (namely dataset S3, cf. Table 1), which will then be employed to fine tune M_S (leading to $M_{S'}$, cf. Table 2). In this context, we also explore the benefits of incorporating real-data shares by fine tuning $M_{S'}$ with small shares of R1 and R2 (in analogy to conventional strategies for bridging the gap), leading to the refined model $M_{F'}$ (cf. Table 2). The final models $M_{S'}$ and $M_{F'}$ undergo further evaluation, particularly in terms of the R1 test and validation set.



Figure 3. Adaptation of the Urban City simulation environment. Visualization of the process of modifying the simulation environment used to generate S1 (cf. camera perspectives labeled S1) based on the false alarms identified in the application of model M_S to real-world data. The camera perspectives based on the modified environment are labeled S3.

Experiment 3: Comparing the Effectiveness of Simulation-Based Training and Conventional Real-Data Training Approaches. The main objective of our third experiment is to demonstrate the potential benefits of simulation-based training techniques for the development of drone detection systems applicable in real-world scenarios. Hence, we perform a comparative analysis of the training strategies employed in Experiments 1 and 2, which (mainly) utilize synthetic data generated by our proposed simulation methodology along with small real-data shares, and (conventional) training techniques that rely solely on manually annotated real-world data. To this end, we begin by conducting a detailed performance analysis of a YOLOv5l model (denoted by M_R , cf. Table 2) trained exclusively on real-world data according to the specifications outlined in Section 2.2. The model's performance is evaluated on the datasets listed in Table 1 using the metrics detailed in Section 2.3. Subsequently, we perform a comprehensive evaluation of M_R in comparison to other drone detection models developed using different training strategies that incorporate synthetic data to varying degrees. These strategies include training exclusively on synthetic data (i.e., M_S , cf. Table 2) as well as training techniques that combine synthetic and real data to different extents (i.e., M_F , M_M , and $M_{F'}$, cf. Table 2). Given the practical necessity that detection models exhibit a certain degree of robustness to variations in environmental conditions, we specifically evaluate the performance of the considered models under varying weather and light conditions as reflected in dataset R4 (i.e., representing a fair reference of comparison).

3. Results and Discussion

3.1. Experiment 1: Quantifying the Simulation–Reality Gap for Models Trained Exclusively on Synthetic Data

The model M_S (trained exclusively on synthetic data) shows high performance on data following a distribution congruent to its training data as evidenced by its mAP of 0.877 and 0.714, respectively (cf. dataset S1, Table 3). This level of performance also extends

to synthetic data with differing data distributions caused by variations in simulation parameters, e.g., triggered by the application of distinct three-dimensional environments (as demonstrated by dataset S2, Table 3). Despite maintaining a high mAP value for S2 and an average TP confidence level of 0.933, a notable increase in FDR can be observed. However, the FP exhibits a significantly lower confidence level compared to the TP, with an average of 0.591.

Table 3. Evaluation of model M_S across different datasets. The performance of model M_S , which was trained exclusively on the synthetic dataset S1, is analyzed on both real and synthetic data.

Dataset	mAP		FNR	FDR
	@ 0.5	avg. @ 0.5–0.95		
R1 (real)	0.551	0.230	0.463	0.500
R2 (real)	0.432	0.175	0.745	0.290
R3 (real)	0.478	0.113	0.550	0.002
R4 (real)	0.656	0.167	0.336	0.512
R5 (real)	0.136	0.050	0.787	0.832
S1 (synth.)	0.877	0.714	0.140	0.120
S2 (synth.)	0.822	0.771	0.167	0.432
S3 (synth.)	0.491	0.294	0.314	0.631

The evaluation of the model on real-world data (R1–R5) reveals substantial variations in performance. The results for datasets R1 to R4 as indicated by their respective mAP values ranging from 0.432 to 0.656 (cf. Table 3), show a significant decline compared to the results derived from S1 and S2. Particularly striking is the weak performance observed for dataset R5 (characterized by extremely low mAP values). Additionally, the proportion of missed detections (represented by the FNR) varies greatly across different datasets. While M_S shows an FNR of 0.337 for R1, the FNR for R5 is significantly higher, at 0.787 (see also Figure 4). Similar deviations can be observed for the FDR, which shows values between 0.002 (R3) and 0.832 (R5). Further examinations of the FPs for both synthetic and real-world data indicate that the FPs are confined to particular regions within a camera’s individual field of view. Therefore, the elevated FDRs can be attributed to a limited number of image areas, which are influenced by the presence of particular background objects. For instance, almost 99.95% of the false positive detections in datasets R1 and R5 can be traced to a single object (namely a traffic sign in the case of R1 and a car in the case of R5, cf. Figure 5). Despite the performance variations, a consistently high level of confidence can be observed for correctly identified objects of R1–R5, with an average value between 0.765 and 0.856.

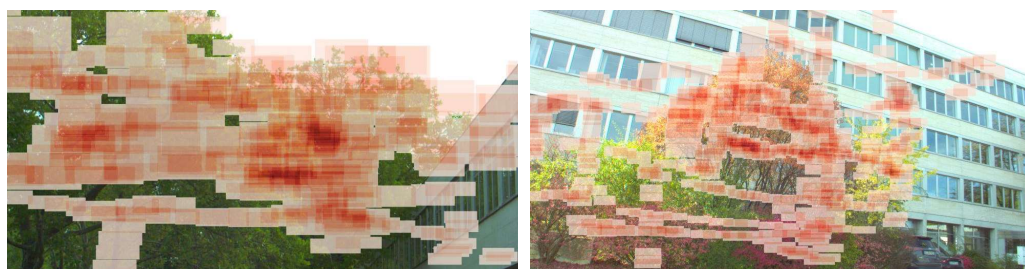


Figure 4. Drones missed by model M_S . A visualization of the regions where model M_S , trained on synthetic data, failed to detect the drones (exemplified by R2, left, and R5, right).

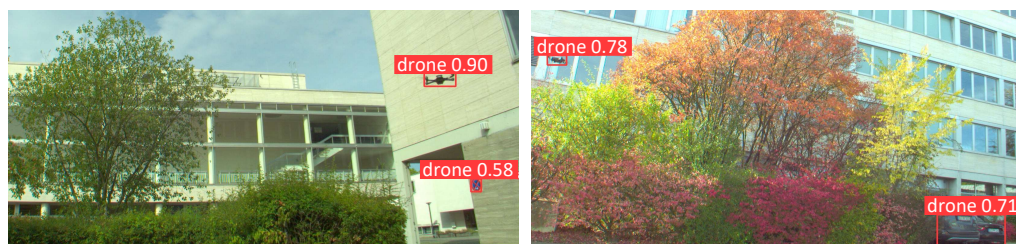


Figure 5. Model M_S on real data—Examples. The results of the detection performed by M_S on samples of R1 (left) and R5 (right) reveal the confinement of FP to specific background objects, such as traffic signs or cars.

Consequently, our findings indicate that (pure) synthetic data (in their present form) are not yet sufficient to guarantee the dependable application of a model trained exclusively on synthetic data in real-world settings without a noticeable degradation in performance. This is coherent with other studies in the field (cf. Section 1.1). In particular, the elevated FDRs and the allocation of FP to specific image areas seem to be a consequence of the non-inclusion of certain background objects (e.g., traffic signs) in the training data. Accordingly, this leads to the assumption that the simulation setup may benefit from the inclusion of elements commonly found in real-world scenarios. Furthermore, heavily textured backgrounds, such as trees (cf. Figure 4), and associated camouflage effects appear to be the primary cause of FN regardless of the underlying dataset.

3.2. Experiment 2: Evaluating the Effectiveness of Established Techniques in Bridging the Simulation–Reality Gap

Fine tuning for Bridging the Simulation–Reality Gap. Fine tuning model M_S (pre-trained on S1, cf. Experiment 1) with incremental shares of R1 and R2 reveals substantial performance improvements on real-world data (cf. Figure 6). Even small data shares ($\leq 5\%$) lead to a significant improvement in mAP with increases of 40 to 50 percentage points, especially for datasets R1–R3. Furthermore, the FNR and FDR of datasets R1, R2, and R3 are significantly reduced as Figure 6 (bottom left and right) illustrates. Integrating only 1% of real-world data into the fine-tuning process already leads to a substantial reduction in FP for datasets R1 and R2, with a decline of 77%. A similar trend in mAP improvement is also noticeable for R4 and R5 (cf. Figure 6, top left), albeit with inferior overall performance compared to R1–R3. However, a distinct difference between R4 and R5 is observed in the FDR values. As the share of real-world data integrated into the fine-tuning process increases, the FDR for R4 exhibits an initial increase as well, followed by a decrease for a real-data share above 10% (cf. Figure 6, bottom left). Conversely, the FDR for R5 initially decreases before exhibiting an upward trend. The FNR of R5 does not show substantial deviations with increasing real-data share, while the FNR of R4 experiences a continuous decline (cf. Figure 6, bottom right).

The results of the the evaluation on synthetic dataset S1 contrast the trend observed for real datasets R1–R3. In the case of synthetic data, fine tuning with even small amounts of real-world data leads to a major decrease in performance as evidenced by a significant decline in mAP (cf. Figure 6, top). This decline is further reflected in the increase in FDR and FNR, depicted in Figure 6 (bottom). Additionally, there is a significant reduction in the TP confidence level, up to 21 percentage points. In contrast, the performance on dataset S2 appears to exhibit limited variation with respect to the shares of real-world data utilized for fine tuning as indicated by the stability of mAP values (at an IoU threshold of 0.5) and the FNR (cf. Figure 6, top left and bottom right). Only the mAP values averaged over IoU thresholds ranging from 0.5 to 0.95 seem to deteriorate with the integration of increasing shares of real-world data (cf. Figure 6, top right).

Summing up, the utilization of even small shares of real data significantly improves the mAP values and reduces the FNRs with respect to datasets R1–R3 (cf. Figure 6). However, the FDRs are still high and exhibit a high level of variability, revealing no clear tendency

(cf. Figure 6, bottom left). Additionally, the findings depicted in Figure 6 (bottom left and top right) convey the impression that integrating large shares of real-world data into the fine-tuning process tends towards overfitting rather than providing a general improvement in performance. This is also supported by the substantial performance decline observed for dataset S1. Consequently, incorporating small shares of real data seems to be the right choice to obtain satisfactory results for real-world scenarios, in which only limited and less diversified data are available.

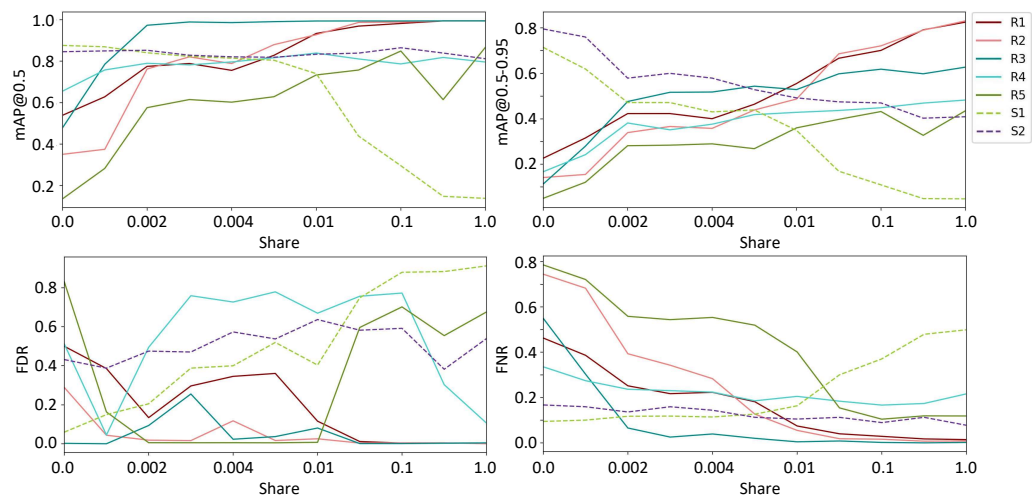


Figure 6. Fine-tuning M_5 for simulation–reality gap reduction. An evaluation of the model performance using mAP (**top left and right**), FDR (**bottom left**), and FNR (**bottom right**) as a function of the real-data share used in fine tuning. (The legend at the top right applies to all graphs.)

Mixed-Data Training. Analyzing the performance outcome obtained by mixed-training strategies utilizing varying shares of real-world data, as illustrated in Figure 7, reveals dataset-specific variations in performance levels, akin to those observed in fine tuning. However, these variations appear to be less pronounced and more consistent as in fine tuning. For instance, when examining the progression of mAP values for an IoU threshold of 0.5 (see Figure 7, top left), we see a slight increase for R1–R5 when using mixed-data training with small shares of real-world data ($\leq 10\%$), followed by stabilization at an almost constant level with further increases in real-data shares. The stabilization of the mAP value is contingent upon the respective dataset, exhibiting values between 0.5 and 0.99. The mAP values with respect to datasets S1 and S2 remain almost unchanged as the shares increase. A comparable behavior can be seen in the progression of FNRs, where even minor shares of real data lead to a rapid decline in FN for all datasets except R5 (cf. Figure 7). Dataset R5 exhibits the highest level of FDR and the lowest mAP values, while datasets R1, R2, and R3 record the best results in terms of FDRs and mAPs.

The evaluation of the FDR, on the other hand, reveals pronounced variations depending on the underlying dataset and the share of real-world data employed in the fine-tuning process. In particular, the FDR values for datasets R4 and R5 show significant fluctuations when fine tuning is performed with small real-data shares ($\leq 10\%$). Conversely, the FDRs of datasets R3 and S2 exhibit an initial increase, followed by a subsequent decline (cf. Figure 7, bottom left). While the FDR associated with dataset S1 displays slight variations in the range of 0 to 0.18, the FDRs of datasets R1 and R2 settle at a fairly consistent level for real-data shares above 0.5% and 0.3%, respectively.

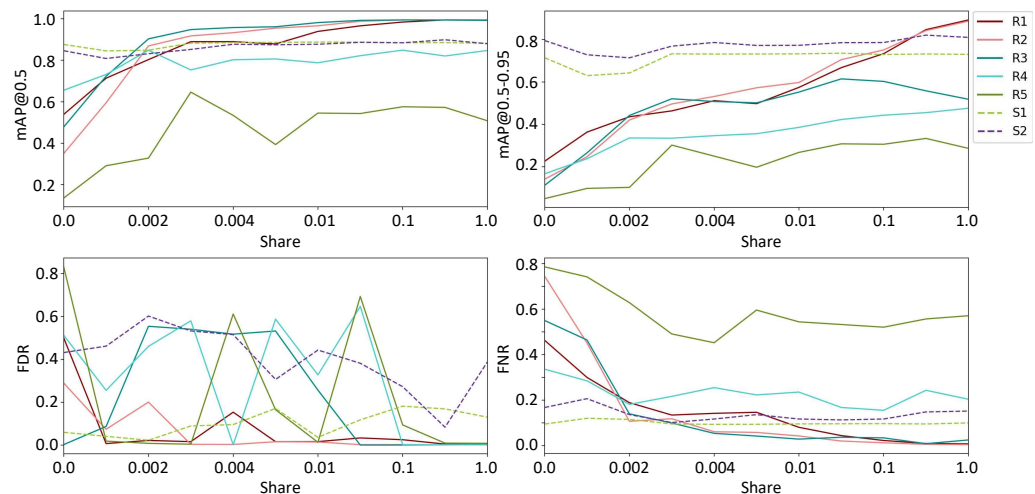


Figure 7. Mixed-data training for simulation–reality gap reduction. The effectiveness of combining real and synthetic data in reducing the simulation–reality gap is measured by the mAP (**top left and right**), FDR (**bottom left**), and FNR (**bottom right**). (The legend at the top right applies to all graphs.)

Summing up, incorporating small shares of real data in mixed-training strategies greatly improves the mAP values for datasets R1–R4 (cf. Figure 7, top left) and reduces the FNRs (cf. Figure 7, bottom right), while the FDRs are still high, strongly affected by noise, and thus exhibit no discernible trend (cf. Figure 7, bottom left). Furthermore, the trend of mAP values for varying IoU thresholds (see Figure 7, top right) indicates a tendency towards overfitting to R1 and R2 as the share of real data increases (cf. Figure 7, top right). Therefore, small amounts of real data seem to be sufficient to provide satisfactory results using mixed-training strategies, which is consistent with our previous findings on fine tuning for bridging the simulation–reality gap.

Feedback Loop-based Training Strategy. In response to the high FDRs of model M_S observed in Experiment 1 when exposed to individual background objects not yet represented in the training data (cf. Section 3.1), we implemented a feedback loop-based training strategy. This training strategy aims to improve the performance of M_S , specifically in terms of false alarms. Building upon the findings of dataset R1, where the high FDR of M_S is found to be attributed to the presence of a traffic sign (cf. Figure 5, left), this involved (i) incorporating the source of the high FDR (i.e., the traffic sign) into the simulation environment (cf. Figure 3), (ii) collecting new data in the modified environment to create a further dataset (S3, cf. Section 2.1), and (iii) fine-tuning M_S using the newly acquired dataset, leading to the refined model $M_{S'}$. Evaluating this iterative training approach reveals substantial improvements in FDR performance, especially when applied to R1 (cf. Tables 4). In particular, the analysis of the refined model $M_{S'}$ on R1 shows a 44 percentage point decrease in FDR, effectively eliminating the number of false alarms triggered by the presence of the traffic sign. However, we also observe a deterioration in other performance metrics, such as mAP and FNR, when compared to our original model M_S .

Drawing upon the findings of our prior investigations on fine tuning and mixed-training strategies, which emphasized the efficacy of integrating small shares of real-world data into the training process, we further explore the potential benefits of incorporating small real-data shares (specifically, R1 and R2) into the iterative training approach as a means of addressing the performance issues (in terms of mAP and FNR). As our results in Table 4 show, fine tuning $M_{S'}$ with small portions of real data (to generate $M_{F'}$) yields a significant improvement in model performance. By incorporating only 0.5% of real-world data, a twofold increase in mAP values is observed for both the IoU threshold of 0.5 and the average over various thresholds ranging from 0.5 to 0.95. Furthermore, a

substantial reduction in FNR by over 50 percentage points is observed, while maintaining a low FDR. Doubling the real-data share to 1% leads to further (albeit less pronounced) performance improvements.

Table 4. Evaluation of the feedback loop-based training strategy. Analyzing the model performance of $M_{S'}$ and $M_{F'}$ on dataset R1 and comparing them to models built on fine tuning (M_F) or mixed-training strategies (M_M).

Real-Data Share	Model	mAP		FNR	FDR
		@ 0.5	avg. @ 0.5–0.95		
0	M_S	0.551	0.230	0.463	0.500
	$M_{S'}$	0.397	0.185	0.667	0.060
0.005	$M_{F'}$	0.853	0.448	0.141	0.058
	M_F	0.829	0.464	0.182	0.361
	M_M	0.879	0.498	0.146	0.016
0.01	$M_{F'}$	0.912	0.514	0.096	0.007
	M_F	0.934	0.556	0.074	0.116
	M_M	0.940	0.575	0.081	0.016

Comparing the performance of model $M_{F'}$ to conventional fine tuning and mixed-training strategies (i.e., models M_F and M_M) for corresponding real-data shares reveals a slightly superior performance of $M_{F'}$ over M_F for small real-data shares ($\leq 0.5\%$). Specifically, there are significant differences in the FDR at a data share of 0.5%, with M_F achieving a score of 36.1%, while $M_{F'}$ achieves a much lower score of 0.7% as shown in Table 4. Conversely, model M_M exhibits a slightly better performance than $M_{F'}$ for a real-data share of 0.5%, although the differences in the evaluation metrics are relatively small (cf. Table 4). As the proportion of real data increases to 1% or higher, the performance values of models $M_{F'}$, M_F , and M_M all remain within a similar high range. While model M_M demonstrates superior performance in terms of mAP, the best performance in terms of FDR is achieved by $M_{F'}$ (cf. Table 4, real-data share of 0.01). Overall, these results highlight the benefits of the feedback loop-based training approach followed by fine tuning with real data (particularly for smaller data shares), leading to comparable performance in terms of mAP and a controlled reduction in FDR to a low level.

3.3. Experiment 3: Comparing the Effectiveness of Simulation-Based Training and Conventional Real-Data Training Approaches

The results of our analysis on model M_R (trained exclusively on real-world data, i.e., R1 and R2) confirm the widely held belief that training DL models on real data results in exceptionally high performance on datasets drawn from the same distribution as the training data. Here, this is indicated by high mAP values of 0.995 at an IoU threshold of 0.5 on both dataset R1 and R2 as well as consistently high mAP values (≥ 0.892) when averaged over a range of IoU thresholds from 0.5 to 0.95 (cf. Table 5). Moreover, the FNRs and FDRs are exceptionally low, both with values below 1% on the same datasets (exhibiting approximately the same distribution).

Table 5. Evaluation of model M_R across different datasets. The performance of model M_R , which was trained exclusively on the real-world datasets R1 and R2, is analyzed on both real and synthetic data.

Dataset	mAP		FNR	FDR
	@ 0.5	avg. @ 0.5–0.95		
R1 (real)	0.995	0.907	0.007	0.005
R2 (real)	0.995	0.892	0.005	0.006
R3 (real)	0.995	0.571	0.010	0.003
R4 (real)	0.827	0.468	0.196	0.005
R5 (real)	0.749	0.382	0.317	0.607
S1 (synth.)	0.099	0.029	0.663	0.943
S2 (synth.)	0.797	0.386	0.105	0.527
S3 (synth.)	0.022	0.110	0.573	0.952

Despite delivering excellent performance on R1 and R2, we observe that even slight alterations in illumination and weather conditions lead to a substantial performance decline in M_R . For instance, comparing the M_R performance on R3 and R2 shows a significant drop in the mAP values (averaged over various IoU thresholds) by 32.1 percentage points, even though the data of R3 were captured using the same camera position and drone model as R2 (cf. Figure 2). This effect can also be observed in a direct comparison between the results on R1 and R4. Here, the model's mAP values decline by a substantial margin of 16.8 and 43.9 percentage points, whereas the FNR increases by 18.9 percentage points (rising from below 1% to 19.6%). The performance decline is even more pronounced for slight alterations in camera position within the real environment as in dataset R5. In this case, the mAP values decrease significantly, while the FNR and FDR increase to levels of 31.7% and 60.7%, respectively. In contrast, the evaluation of M_R on synthetic data reveals a consistently poor performance, except for S2, where M_R performs similarly to R5 (cf. Table 5). On S1 and S3, M_R exhibits mAP values close to zero, FNRs of 57.3% and 66.3%, and FDRs exceeding 90%. Similar to the findings for model M_S (cf. Section 3.1 and Figure 5, right), the high FDRs observed for R5 and S1–S3 can be attributed to specific background objects within the images with dependence on the camera position. The TP confidence level remains stable for datasets R1–R4 with values exceeding 93%, whereas R5 and S1–S3 exhibit inferior confidence levels between 74.6% and 90%.

We then conduct a direct performance comparison between two models: M_R , trained exclusively on real-world data of R1 and R2, and M_S , trained on synthetic data only (cf. Experiment 1, Section 3.1). As previously noted, both models achieve high mAP values on data similar to their respective training data distributions. Specifically, M_S performs exceptionally well on the synthetic datasets S1 and S2, while M_R achieves higher mAP values on data sampled from R1 and R2 (cf. Tables 4 and 5). However, the nearly perfect mAP values attained by M_R suggest overfitting to R1 and R2, which is further supported by the observed drop in performance on fairly similar datasets (such as R5, where the model's mAP values decline by 24.6 and 52.6 percentage points, respectively, cf. Table 5) and the mAP values approaching zero on synthetic data (cf. dataset S1, Table 5). Even though M_S exhibits some dataset-dependent performance degradation as well, overfitting in the case of M_S seems to be less pronounced as indicated by lower variations in the corresponding quality measures across different datasets (e.g., the performance gap in terms of mAP between S1 and R1 is 0.326 for M_S and 0.896 for M_R , cf. Tables 3 and 5). Nevertheless, the performance of M_S on real-world data is rather poor, with particularly high FNRs (e.g., the FNR on R1 is as high as 46.3%), significantly compromising the model's utility in practical applications.

Addressing the shortcomings of a purely simulation-based training approach for DL models destined for real-world applications, we expand our comparative analysis by

incorporating models trained on a combination of synthetic data and small real-data shares. Specifically, we include the models M_F , M_M , and $M_{F'}$, which were studied in Experiment 2 and exhibited promising results (cf. Section 3.2 for detailed information). Evaluating these models on dataset R4, we find that even small shares of real-world data (as low as 0.2%), in combination with synthetic data, lead to model performance results that are comparable to or occasionally even slightly better than those of M_R . For instance, while M_R achieves an mAP value of 0.827 at an IoU threshold of 0.5 (cf. Table 6), the performance of M_F , M_M , and $M_{F'}$ in terms of mAP ranges between 0.788 (for M_M with a real-data share of 1%) and 0.863 (for $M_{F'}$ with a real-data share of 0.5%). A similar trend can be observed for mAP values averaged over various IoU thresholds (ranging from 0.5 to 0.95), with values between 0.427 and 0.29 for M_F , M_M , and $M_{F'}$, and a slightly superior performance of M_R with an mAP of 0.468. Regarding the FNR, we find that M_F , M_M , and $M_{F'}$ exhibit values between 16.9% and 23.7%, with model $M_{F'}$ achieving the lowest average FNR at a real-data share of 0.5%. However, we observe significant differences in FDR between the models M_F , M_M , and $M_{F'}$. While the FDR for M_F and M_M is generally rather high, reaching values between 32.8% and 77.9%, model $M_{F'}$ exhibits FDRs below 7.2%, which decrease further with increasing real-data shares (cf. Table 6). In fact, for a real-data share of 1%, the FDR of $M_{F'}$ even exceeds the performance of M_R .

Table 6. Evaluation of the effectiveness of simulation-based training strategies. The performance of various models, obtained by combining different compositions of real and synthetic training data and distinct training strategies, on dataset R4.

Real-Data Share	Model	mAP		FNR	FDR
		@ 0.5	avg. @ 0.5–0.95		
0	M_S	0.656	0.167	0.336	0.512
	$M_{S'}$	0.680	0.272	0.364	0.083
0.002	$M_{F'}$	0.793	0.401	0.212	0.072
	M_F	0.791	0.382	0.237	0.495
	M_M	0.849	0.337	0.181	0.461
0.005	$M_{F'}$	0.863	0.419	0.169	0.056
	M_F	0.817	0.419	0.186	0.779
	M_M	0.806	0.356	0.223	0.588
0.01	$M_{F'}$	0.808	0.427	0.198	0.031
	M_F	0.840	0.429	0.205	0.670
	M_M	0.788	0.384	0.235	0.328
1	M_R	0.827	0.468	0.196	0.005

In summary, our observations demonstrate that incorporating small amounts of real-world data alongside the synthetic training data generated by our proposed methodology yields outcomes commensurate with those attained by a model exclusively trained on real data (in terms of relevant quality measures).

4. Conclusions and Topics of Future Research

In this work, we investigated the suitability of physically realistic three-dimensional simulations to generate synthetic data for training DL-based drone detectors. We proposed a novel game engine-based data generation technique for drone detection relying on genuine three-dimensional simulations, rather than domain randomization techniques built on two-dimensional backgrounds (as in previous literature). We quantified the simulation–reality gap introduced by our methodology and observed that training exclusively on synthetic data yields satisfactory results on synthetic test data but limits its effectiveness in real-world scenarios. To address this issue, we systematically evaluated established techniques for bridging the gap between simulation and reality and found that incorporating small shares of real data (0.2–1%) in addition to synthetic data generated by our approach (either via

fine tuning or mixed-training strategies) leads to a significant increase in mAP values and a substantial decline in FNRs on real test data. Moreover, we successfully employed a feedback loop-based training strategy to reduce high FDRs. Finally, we highlighted the potential of a simulation-based training approach for developing drone detection models, in comparison to training strategies relying solely on real data. Our findings indicate that the inclusion of small amounts of real data (0.2–1%) into the training process leads to performance outcomes comparable to those obtained through exclusive real-data training (with respect to relevant quality measures), while significantly reducing the required resources. Furthermore, our experiments suggest that networks trained solely on real-world data are prone to overfitting, a phenomenon that is not observed in training configurations using synthetic data paired with smaller real-data shares.

In light of our findings, there are several promising avenues for future research. One topic of interest will be the detection of drones in front of highly textured backgrounds, such as trees, which pose a significant challenge for current detection systems (not fully resolved by our approach). Developing effective techniques for improving the discriminative power of drone detection models (e.g., by combining different feature extraction and image processing techniques) could considerably enhance the detection quality in such scenarios, as well as the overall robustness and applicability of image-based drone detectors. Another promising research direction is the incorporation of more diversified objects commonly encountered in real-world scenarios, both static (e.g., road signs) and dynamic (e.g., vehicles, pedestrians, or birds), into the simulation environment. Currently, our approach does not comprehensively represent such objects and their associated effects, such as motion blur. Incorporating such objects could facilitate the generation of more realistic synthetic data, which, in turn, could lead to better generalization and improved detection performance in real-world settings. A systematical expansion of the simulation setting could be realized based on the scenario space introduced in [3] (e.g., focusing on relevant threat scenarios).

Author Contributions: Conceptualization, T.R.D. and A.W.; methodology, T.R.D.; software, T.R.D. and S.J.; validation, T.R.D.; formal analysis, T.R.D.; investigation, T.R.D. and A.W.; resources, E.B.; data curation, T.R.D. and S.J.; writing—original draft preparation, T.R.D.; writing—review and editing, T.R.D. and A.W.; visualization, T.R.D.; supervision, A.W.; project administration, E.B.; funding acquisition, E.B. All authors have read and agreed to the published version of the manuscript.

Funding: We acknowledge support by the Deutsche Forschungsgemeinschaft (DFG German Research Foundation) and the Open Access Publishing Fund of Hochschule Darmstadt—University of Applied Sciences.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: The authors gratefully acknowledge support by the German Aerospace Center (DLR), Institute for the Protection of Terrestrial Infrastructures, for providing essential resources and technical support.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AR	average recall
CAD	computer aided design
CV	computer vision

DL	deep learning
DR	domain randomization
FDR	false discovery rate
FN	false negatives
FNR	false negative rate
FP	false positives
GAN	generative adversarial network
IoU	intersection over union
mAP	mean average precision
mAR	mean average recall
R-CNN	region-based convolutional neural network
SSD	single shot multibox detector
TP	true positives
UAV	unmanned aerial vehicle
YOLO	you only look once

References

- Zhang, X.; Kusriani, K. Autonomous Long-Range Drone Detection System for Critical Infrastructure Safety. *Multimedia Tools Appl.* **2021**, *80*, 23723–23743. [[CrossRef](#)]
- Zhang, X.; Izquierdo, E.; Chandramouli, K. Critical Infrastructure Security Using Computer Vision Technologies. In *Security Technologies and Social Implications*; John Wiley & Sons, Ltd.: Hoboken, NJ, USA, 2022; Chapter 6, pp. 149–180. [[CrossRef](#)]
- Schneider, M.; Lichte, D.; Witte, D.; Gimbel, S.; Brucherseifer, E. Scenario Analysis of Threats Posed to Critical Infrastructures by Civilian Drones. In Proceedings of the 31st European Safety and Reliability Conference (ESREL), Angers, France, 19–23 September 2021; pp. 520–527. [[CrossRef](#)]
- Bernardini, A.; Mangiatordi, F.; Pallotti, E.; Capodiferro, L. Drone detection by acoustic signature identification. *Electron. Imaging* **2017**, *2017*, 60–64. [[CrossRef](#)]
- Seidaliyeva, U.; Akhmetov, D.; Ilipbayeva, L.; Matson, E.T. Real-Time and Accurate Drone Detection in a Video with a Static Background. *Sensors* **2020**, *20*, 3856. [[CrossRef](#)] [[PubMed](#)]
- Nguyen, P.; Truong, H.; Ravindranathan, M.; Nguyen, A.; Han, R.O.; Vu, T.N. Cost-Effective and Passive RF-Based Drone Presence Detection and Characterization. *GetMobile Mob. Comput. Commun.* **2018**, *21*, 30–34. [[CrossRef](#)]
- Nuss, B.; Sit, L.; Fennel, M.; Mayer, J.; Mahler, T.; Zwick, T. MIMO OFDM radar system for drone detection. In Proceedings of the 18th International Radar Symposium (IRS), Prague, Czech Republic, 28–30 June 2017; pp. 1–9. [[CrossRef](#)]
- Svanström, F.; Englund, C.; Alonso-Fernandez, F. Real-Time Drone Detection and Tracking With Visible, Thermal and Acoustic Sensors. In Proceedings of the 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 7265–7272. [[CrossRef](#)]
- Chiper, F.L.; Martian, A.; Vladeanu, C.; Marghescu, I.; Craciunescu, R.; Fratu, O. Drone Detection and Defense Systems: Survey and a Software-Defined Radio-Based Solution. *Sensors* **2022**, *22*, 1453. [[CrossRef](#)] [[PubMed](#)]
- Kashiyama, T.; Sobue, H.; Sekimoto, Y. Sky Monitoring System for Flying Object Detection Using 4K Resolution Camera. *Sensors* **2020**, *20*, 71. [[CrossRef](#)] [[PubMed](#)]
- Srigarom, S.; Lee, S.M.; Lee, M.; Shaohui, F.; Ratsamee, P. An Integrated Vision-based Detection-tracking-estimation System for Dynamic Localization of Small Aerial Vehicles. In Proceedings of the 5th International Conference on Control and Robotics Engineering (ICCRE), Osaka, Japan, 24–26 April 2020; pp. 152–158. [[CrossRef](#)]
- Rozantsev, A.; Lepetit, V.; Fua, P. Detecting Flying Objects Using a Single Moving Camera. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 879–892. [[CrossRef](#)] [[PubMed](#)]
- Peng, J.; Zheng, C.; Lv, P.; Cui, T.; Cheng, Y.; Lingyu, S. Using images rendered by PBRT to train faster R-CNN for UAV detection. In Proceedings of the 26th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Pilsen/Prague, Czech Republic, 28 May–1 June 2018; pp. 13–18. [[CrossRef](#)]
- Grác, S.; Beño, P.; Duchoň, F.; Dekan, M.; Tölgyessy, M. Automated Detection of Multi-Rotor UAVs Using a Machine-Learning Approach. *Appl. Syst. Innov.* **2020**, *3*, 29. [[CrossRef](#)]
- Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587. [[CrossRef](#)]
- Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448. [[CrossRef](#)]
- Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]

18. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37. [[CrossRef](#)]
19. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788. [[CrossRef](#)]
20. Akyon, F.C.; Eryuksel, O.; Özfuttu, K.A.; Altinuc, S.O. Track Boosting and Synthetic Data Aided Drone Detection. In Proceedings of the 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Washington, DC, USA, 16–19 November 2021; pp. 1–5. [[CrossRef](#)]
21. Marez, D.; Borden, S.; Nans, L. UAV detection with a dataset augmented by domain randomization. In *Geospatial Informatics X*; SPIE: Bellingham, WA, USA, 2020; Volume 11398. [[CrossRef](#)]
22. Chen, Y.; Aggarwal, P.; Choi, J.; Kuo, C.C.J. A deep learning approach to drone monitoring. In Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Kuala Lumpur, Malaysia, 12–15 December 2017; pp. 686–691. [[CrossRef](#)]
23. Özfuttu, K.A. Generating Synthetic Data with Game Engines for Deep Learning Applications. Master’s Thesis, Hacettepe University, Ankara, Turkey, 2022.
24. Symeonidis, C.; Anastasiadis, C.; Nikolaidis, N. A UAV Video Data Generation Framework for Improved Robustness of UAV Detection Methods. In Proceedings of the IEEE 24th International Workshop on Multimedia Signal Processing (MMSp), Shanghai, China, 26–28 September 2022; pp. 1–5. [[CrossRef](#)]
25. Barisic, A.; Petric, F.; Bogdan, S. Sim2Air - Synthetic Aerial Dataset for UAV Monitoring. *IEEE Robot. Autom. Lett. (RA-L)* **2022**, *7*, 3757–3764. [[CrossRef](#)]
26. Johnson-Roberson, M.; Barto, C.; Mehta, R.; Sridhar, S.N.; Rosaen, K.; Vasudevan, R. Driving in the Matrix: Can Virtual Worlds Replace Human-Generated Annotations for Real World Tasks? In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 746–753. [[CrossRef](#)]
27. 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 (CVPRW), Salt Lake City, UT, USA, 18–22 June 2018.
28. Richter, S.R.; Vineet, V.; Roth, S.; Koltun, V. Playing for Data: Ground Truth from Computer Games. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9906, pp. 102–118.
29. Richter, S.R.; Hayder, Z.; Koltun, V. Playing for Benchmarks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2232–2241. [[CrossRef](#)]
30. Prakash, A.; Boochoon, S.; Brophy, M.; Acuna, D.; Cameracci, E.; State, G.; Shapira, O.; Birchfield, S. Structured Domain Randomization: Bridging the Reality Gap by Context-Aware Synthetic Data. In Proceedings of the International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 7249–7255. [[CrossRef](#)]
31. Dieter, T.; Weinmann, A.; Brucherseifer, E. Generating Synthetic Data for Deep Learning-Based Drone Detection. In Proceedings of the 48th International Conference of Applications of Mathematics in Engineering and Economics (AMEE), Sozopol, Bulgaria, 7–13 June 2022.
32. Epic Games. Unreal Engine. Available online: <https://www.unrealengine.com/en-US/> (accessed on 6 February 2023).
33. Unity Technologies. Unity. Available online: <https://unity.com/de> (accessed on 6 February 2023).
34. Pham, H.X.; Sarabakha, A.; Odnoshyvkin, M.; Kayacan, E. PencilNet: Zero-Shot Sim-to-Real Transfer Learning for Robust Gate Perception in Autonomous Drone Racing. *IEEE Robot. Autom. Lett.* **2022**, *7*, 11847–11854. [[CrossRef](#)]
35. Hodaň, T.; Vineet, V.; Gal, R.; Shalev, E.; Hanzelka, J.; Connell, T.; Urbina, P.; Sinha, S.N.; Guenter, B. Photorealistic Image Synthesis for Object Instance Detection. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 66–70. [[CrossRef](#)]
36. Reway, F.; Hoffmann, A.; Wachtel, D.; Huber, W.; Knoll, A.; Ribeiro, E. Test Method for Measuring the Simulation-to-Reality Gap of Camera-based Object Detection Algorithms for Autonomous Driving. In Proceedings of the IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA; 19 October–13 November 2020, pp. 1249–1256. [[CrossRef](#)]
37. Collins, J.; Brown, R.; Leitner, J.; Howard, D. Traversing the Reality Gap via Simulator Tuning. *arXiv* **2020**, arXiv:2003.01369.
38. Nowruzzi, F.E.; Kapoor, P.; Kolhatkar, D.; Hassanat, F.A.; Laganière, R.; Rebut, J. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. *arXiv* **2019**, arXiv:1907.07061.
39. Dwibedi, D.; Misra, I.; Hebert, M. Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1310–1319. [[CrossRef](#)]
40. Georgakis, G.; Mousavian, A.; Berg, A.; Košecká, J. Synthesizing Training Data for Object Detection in Indoor Scenes. In Proceedings of the Robotics: Science and Systems XIII, Cambridge, MA, USA, 12–16 July 2017. [[CrossRef](#)]
41. Rozantsev, A.; Lepetit, V.; Fua, P. On rendering synthetic images for training an object detector. *Comput. Vis. Image Underst.* **2015**, *137*, 24–37. [[CrossRef](#)]
42. Yoo, U.; Zhao, H.; Altamirano, A.; Yuan, W.; Feng, C. Toward Zero-Shot Sim-to-Real Transfer Learning for Pneumatic Soft Robot 3D Proprioceptive Sensing. *arXiv* **2023**, arXiv:2303.04307.
43. Nikolenko, S.I. *Synthetic Data for Deep Learning*; Springer Optimization and Its Applications: Berlin, Germany, 2021.

44. Microsoft Research. Welcome to AirSim. Available online: <https://microsoft.github.io/AirSim> (accessed on 16 January 2023).
45. Chamola, V.; Kotesch, P.; Agarwal, A.; Naren.; Gupta, N.; Guizani, M. A Comprehensive Review of Unmanned Aerial Vehicle Attacks and Neutralization Techniques. *Ad Hoc Netw.* **2021**, *111*. [[CrossRef](#)] [[PubMed](#)]
46. PolyPixel. Urban City. Available online: <https://www.unrealengine.com/marketplace/en-US/product/urban-city> (accessed on 24 February 2023).
47. SilverTm. City Park Environment Collection. Available online: <https://www.unrealengine.com/marketplace/en-US/product/city-park-environment-collection> (accessed on 24 February 2023).
48. Singha, S.; Aydin, B. Automated Drone Detection Using YOLOv4. *Drones* **2021**, *5*, 95. [[CrossRef](#)]
49. Aydin, B.; Singha, S. Drone Detection Using YOLOv5. *Eng* **2023**, *4*, 25. [[CrossRef](#)]
50. Jung, H.K.; Choi, G.S. Improved YOLOv5: Efficient Object Detection Using Drone Images under Various Conditions. *Appl. Sci.* **2022**, *12*, 7255. [[CrossRef](#)]
51. Aker, C.; Kalkan, S. Using deep networks for drone detection. In Proceedings of the 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 29 August–1 September 2017, pp. 1–6. [[CrossRef](#)]
52. Mishra, A.; Panda, S. Drone Detection using YOLOv4 on Images and Videos. In Proceedings of the 7th IEEE International Conference for Convergence in Technology (I2CT), Mumbai, India, 7–9 April 2022; pp. 1–4. [[CrossRef](#)]
53. Ultralytics. YOLOv5: The Friendliest AI Architecture You'll Ever Use. Available online: <https://ultralytics.com/yolov5>, Last visit: 06.02.2023 (accessed on 6 February 2023).
54. Bochkovskiy, A.; Wang, C.; Liao, H.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
55. Lin, T.Y.; Maire, M.; Belongie, S.J.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In Proceedings of the European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2014. [[CrossRef](#)]
56. Padilla, R.; Netto, S.L.; da Silva, E.A.B. A Survey on Performance Metrics for Object-Detection Algorithms. In Proceedings of the International Conference on Systems, Signals and Image Processing (IWSSIP), Niteroi, Brazil, 1–3 July 2020; pp. 237–242. [[CrossRef](#)]
57. Hastie, T.; Tibshirani, R.; Friedman, J.H. *The Elements of Statistical Learning*; Springer: Berlin/Heidelberg, Germany, 2017.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.