IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 7 August 2023, accepted 27 September 2023, date of publication 5 October 2023, date of current version 11 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3322374

APPLIED RESEARCH

Sim-to-Real Transfer for Object Detection in Aerial Inspections of Transmission Towers

AUGUSTO J. PETERLEVITZ[1], MATEUS A. CHINELATTO[1], ANGELO G. MENEZES[2], CÉZANNE A. M. MOTTA[1], GUILHERME A. B. PEREIRA[1], GUSTAVO L. LOPES[1], GUSTAVO DE M. SOUZA[1,2], JUAN RODRIGUES[1], LILIAN C. GODOY[1], MARIO A. F. F. KOLLER[1], MATEUS O. CABRAL[1], NICOLE E. ALVES[1], PAULO H. SILVA[1], RICARDO CHEROBIN[1], ROBERTO A. O. YAMAMOTO[1], AND RICARDO D. DA SILVA[1]

[1]Computer Vision Department, Eldorado Research Institute, Campinas, São Paulo 13083-898, Brazil
[2]Institute of Mathematics and Computer Sciences, University of São Paulo, São Carlos, São Paulo 13566-590, Brazil

Corresponding author: Augusto J. Peterlevitz (augusto.peterlevitz@eldorado.org.br)

This work was supported by Transmissora Aliança de Energia Elétrica S.A. (TAESA) through Agência Nacional de Energia Elétrica (ANEEL) Research and Development Program under Project PD-07130-0059/2020.

ABSTRACT Training deep learning models for object detection usually requires a large amount of data, a condition that is not common for most real-world applications, especially in the context of aerial imagery. One possible solution is the use of simulators to generate synthetic data. For a good generalization, the model must be able to learn on the simulated data and perform correctly on the real data, process known as sim-to-real transfer. In this work, we analyze the generation of synthetic data to account for a data-scarce real-world scenario, which includes aerial imagery and object detection of transmission towers and their components. We evaluate the impact of image-to-image translation methods as domain adaptation techniques. In this analysis, we explore training strategies to mitigate the domain shift between synthetic and real data. According to our experimental results, the use of domain-adapted data through image-to-image translation could slightly improve the detection performance in real test data when compared to training with raw synthetic images only or with small datasets of real data, although it was noted through a visual analysis that objects with small bounding boxes, like clamp, anchoring clamp and ball link, could be distorted or vanished by the application of image-to-image translation methods. Additionally, when only a small subset of real data is available, training with both real and synthetic data at once led to better detection results, surpassing combinations of pre-training on synthetic and fine-tuning on real data.

INDEX TERMS Aerial inspection, domain adaptation, object detection, sim-to-real transfer, domain adaptation, image-to-image translation.

NOMENCLATURE

$\mathcal{D}_r$	Real dataset.
$\mathcal{D}_s$	Synthetic dataset.
AP	Average Precision.
AR	Average Recall.
CUT	Contrastive Unpaired Translation.
CycleGAN	Cycle-Consistent Adversarial Networks.
GANs	Generative Adversarial Networks.
IOU	Intersection Over Union.
LPIPS	Learned Perceptual Image Patch Similarity.
mAP	Mean Average Precision.
TSIT	Two-Stream Image-to-image Translation.
UAV	Unmanned Aircraft Vehicle.
YOLO	You Only Look Once.

The associate editor coordinating the review of this manuscript and approving it for publication was Giambattista Gruosso.

I. INTRODUCTION

Deep Learning models have been applied throughout the years in a wide range of computer vision tasks, such as image classification, image segmentation, and object detection. These models usually require a great volume of samples with good variance in order to reach a good performance and

avoid overfitting, which can be a challenging assignment [1]. Obtaining a great number of pictures in some scenarios can be a difficult task, not to mention that the annotation of those images can be a time-consuming and laborious task. This is the case of object detection in aerial scenarios [2], [3], [4]. In general, large sets of real images taken from aircrafts are very costly. In this regard, the popularization of UAV's (Unmanned Aircraft Vehicles) has made a positive impact, giving access to cheap airborne devices capable of shooting high-quality photos. However, even with access to drones, this is still a hard task since it requires a trained operator, is limited by the device's battery life (still not great nowadays), and depends on favorable weather.

For computer vision scenarios in which data can be scarce, some common techniques have been used to deal with the limited volume of images by generating synthetic data. This has been achieved by the application of image transformation techniques (e.g., flipping, rotation, shifting, cropping), generative models, and the use of simulators, which will be described more deeply in Section II.

The use of synthetic data generated by computer simulations has been a game changer for AI development for several reasons. Reducing costs and increasing time efficiency are the main factors since a limited effort of setting up a realistic simulation or a generative model can be worth a large volume of high-quality data [5]. Moreover, restrictive licenses for data usage and privacy are not a problem for synthetic data generated internally.

Nevertheless, the use of simulated data might not be the ''final'' step for a successful application of computer vision in the real world since there is often a large gap in visual features between the generated and real data [6], [7]. Although the quality of computer-generated images has improved greatly in the last few years, the photorealistic simulation of light and texture still poses a great challenge for computer graphics research [5]. For situations where there may be differences in synthetic and real data distributions, researchers have explored the application of style transfer and domain adaption techniques (techniques that aim to improve the performance of a model trained on data that belongs to a different domain) to compensate for these differences and improve results for applications in the real-world [8], [9], [10], [11].

For this study, we are interested in investigating how to overcome the lack of training data available in a real-world aerial inspection scenario that involves detecting electrical transmission towers and their components in real-time. Considering that gathering enough data samples in this context can be quite costly, we explored the possibility of generating synthetic data by computer simulations (here referred as ''synthetic data'') to expand our initial dataset. For this purpose, we designed a photorealistic environment using the AirSim simulator [5], which is based on the Unreal Engine [12]. To further compensate for the appearance differences between synthetic and real images, we explored the use of domain adaption techniques based on image-to-image translation and evaluate two practical scenarios for training the model: mixed training with all available data and fine-tuning on a real-data subset.

To better describe our methodology and findings, we organize the paper as follows: Section II gives a short technical background and discusses some of the related work that involves this paper's main topics, Section III describes in detail the data processing methods, and proposed experiments, followed by Section IV with the presentation and discussion of the results. Finally, Section V provides our final considerations regarding the findings of this paper and possible research directions to improve the image-to-image translation methods usage as ready-to-use domain adaptation techniques for specific tasks such as object detection in UAV images collected for transmission lines inspections.

II. BACKGROUND AND RELATED WORK

This section briefly describes the base knowledge necessary for understanding the discussions in the next sections and contextualizes related works on the fields explored by this paper.

A. AERIAL INSPECTION OF TRANSMISSION TOWERS

The process of performing a preventive inspection of transmission towers is important for evaluating the lifecycle of components and keeping the safety of those around them. During the inspection, several tower components, such as lightning rods, insulator strings, spans, and conductor cables, must be checked for defects or abnormalities.

Considering the large number of items to be inspected and the difficulty of evaluating them individually, which is mostly done by climbing the structure or with the use of binoculars, several companies have invested in the use of UAVs for both increase in speed and quality, and a decrease of costs in each inspection. However, the assessment must still be properly planned and supervised by people with the right site safety skills and equipment to avoid causalities around the electrical transmission structure.

With the advent of using a UAV in an inspection, the technician has the ability to take high-quality pictures of electrical components, archive them for comparisons and then list their corrosion and degradation levels, which is another laborious task prone to automation with the use of machine learning techniques. To achieve this, the drone can run an object detection model to detect and identify each electrical accessory and guide its flight toward acquiring photos with similar framing and conditions to aid the standardization of the assessment.

B. OBJECT DETECTION

The object detection task involves identifying an object alongside its position in an image. Traditional techniques have explored the use of feature descriptors and other handcrafted features [13] as reference for object localization. However, with the recent advances on deep learning and the improvement of hardware capabilities, most of the

state-of-the-art methods in this field has been built on top of deep neural networks [14], [15].

Detectors based on deep learning can have several different architectures and perform the detection on one or multiple stages. Two-stage detectors usually work by first identifying regions of interest or possible bounding boxes and then running a classifier on those regions with sometimes a post-processing step for filtering out duplicated detections [16]. Methods such as R-CNN [17] and its extensions, Fast R-CNN [18], Faster R-CNN [19] and Mask R-CNN [20], are all two-stage models and have been largely employed as successful strategies for real-world applications [21], [22], [23]. Alternatively, one-stage models treat the whole detection as a single task. The YOLO network, for example, interprets object detection as a single regression problem with the pixels of the image as inputs and the bounding boxes coordinates and class probabilities as outputs, making it simpler and extremely fast [24]. The tradeoff commonly involved when choosing to use one-stage over two-stage detectors is based on the preference for the former when speed is necessary, with the caveat of a decrease in performance. However, new one-stage detectors, such as FCOS [25] and YOLOv5 [26], have shown that one-stage solutions are also suitable for performance-dependent tasks.

The detections are evaluated by calculating how much a predicted bounding box for an object hits or misses the ground truth annotation based on a specified threshold. The hits and misses are calculated in terms of true positives (hits), false positives (mistakes) and false negatives (misses). The IOU (Intersection Over Union) value is used to measure the ratio between the intersection area and union area of the predicted and ground truth bounding boxes, so as the intersection area grows, this value also grows. In (1), B_{pred} refers to the predicted bounding box and B_{gt} to the corresponding ground truth, Fig.1 exemplifies this. The defined threshold specifies the minimal overlap needed to consider a prediction as hit, a miss or a mistake.

$$Jaccard\ Index = IOU = \frac{area(B_{pred} \cap B_{gt})}{area(B_{pred} \cup B_{gt})} \quad (1)$$

From individual prediction metrics, the average precision (AP), average recall (AR), and mean average precision for all classes (mAP) can be calculated and used as means of comparing the detection capabilities of a model. Also, the AP can be calculated by varying the IOU threshold from 0.5 to 0.95, with a step size of 0.05, and then the mean value within these thresholds is obtained, followed by the mean value between the classes, usually reported as $mAP@[.5 : .95]$ [27].

The successful training of these detectors is somewhat dependent on the availability of large datasets. Transfer learning and the fine-tuning of models trained initially on publicly large datasets such as VOC [28] and COCO [29] is often the preferred direction for practitioners. Yet, when the domain for the task is too different from the one originally used for training the base model, their use comes down to only serving as a better weight initialization for the neural network.

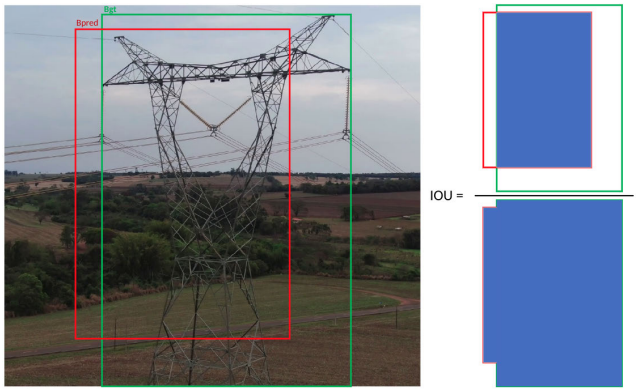


FIGURE 1. IOU calculation from predicted and ground truth bounding boxes - made by the authors.

C. SYNTHETIC DATA

Lack of data is one of the most challenging problems to overcome when training deep learning models. For several situations, mainly in real-world applications, it might not be possible to acquire large real datasets due to cost, time, or lack of variance on the available data sources.

The popularization of game engines has made it possible to create environments with great fidelity based on physical simulations. In such scenarios, all parameters for the scene can be adjusted to assemble the highest photorealistic appearance that mimics the real scene and its objects. One example of both a similar scene in the real world and simulator can be seen in Fig. 2.

(a) Real **(b)** Simulation

FIGURE 2. Real and simulation driving scenes from CityScapes [30] and GTAV [31] datasets.

From one simulated environment, thousands or even millions of images can be generated for different framing and positions. In this way, game engines have currently become the go-to tool for leveraging applications that involve the training of deep learning models when the number of available labeled data is nonexistent or small, which is the case for several applications in the inspection industry due to costs to generate and annotate data [32], [33].

In this context, Hwang et al [34] used the Unreal Engine to develop ElderSim, a platform for generating synthetic data for human action recognition of elderly people. Hu et al. [35] used the engine of the game Grand Theft Auto V (GTAV) to create a dataset called SAIL-VOS 3D for object detection and 3D mesh reconstruction from videos. Also using the GTAV environment, Angus et al. [36] created the URSA

dataset for semantic segmentation of road-scene images. For the purpose of training and validating autonomous cars systems, the CARLA simulator was developed by Dosovitskiy et al. [37], which included the generation of semantic segmentation, object detection, depth, and LiDAR (Light Detection and Ranging) data.

For creating reliable environments with aerial views of scenes, the AirSim plugin for the Unreal Engine was proposed by Shah et al. [5]. With this tool, large datasets such as the Mid-Air [38] and TartanAir [39] benchmarks were created to advance research in robot navigation and visual understanding. Both datasets present RGB frames, surface normal orientation, depth, and semantic information for each scene captured in several drone flights, which make them useful for solving other related visual tasks.

D. SIM-TO-REAL TRANSFER

Although synthetic data generation is a promising approach when data is lacking, the performance of a deep learning model trained on synthetic data may not be satisfactory when testing on real data due to the domain shift between the two datasets. The distribution of the visual features present in the synthetic data domain may differ from those in the real domain distribution [10] as texture and light reflectance are hard to be represented computationally. The process in which a model is trained on simulated data and tries to perform on real data is referred as ''sim-to-real transfer'' (here, ''sim'' standing for ''synthetically generated images'').

To successfully solve the sim-to-real generalization issue, diverse domain adaptation, randomization, and transfer techniques have been proposed. Tobin et al. [40] explored the concept of domain randomization by training an object detection model on geometric objects with non-realistic random textures and demonstrated the model trained on the simulated data has better generalization when tested in the real domain. Tremblay et al. [41] applied domain randomization on object detection of cars by generating scenes with random position, rotation, and textures of the vehicles, as also including in the scene random geometric shapes to be used as negative examples and randomized lighting and position of the camera. The models were trained on the resulting unrealistic synthetic dataset and then fine-tuned on real data, improving the results when testing on the real domain.

Tang et al. [42] explored the usage of synthetic data generated with Blender [43] and studied a handful of aspects regarding the usage of such data for downstream segmentation tasks. The results suggested that a pre-training performed on synthetic data can be better than pre-training on real data for model generalization. It also remarked that synthetic images should be generated with variations on object scale, textures, camera viewpoint and image background in order to increase their contribution to the model performance.

Some specific works have focused on making synthetic data more realistic through the use of GANs. Shrivastava et al. Reference [8] proposed the SimGAN, an adversarial network that learns to improve the realism of synthetic data for gaze estimation using unsupervised learning. Bousmalis et al. [44] used a GAN approach on the Synthetic Cropped LineMod [45] dataset to make the synthetic 3D models closer to the target domain for object classification and position estimation. [46] proposed the GeneSIS-RT, an approach that explores the use of CycleGAN, [47], for image-to-image translation, transforming simulated images into more realistic ones for segmentation and obstacle avoidance tasks for the flight of a quadcopter. Also based on the CycleGAN, Hoffman et al. [9] presented CyCADA with the addition of a semantic consistency loss to constrain the image-to-image translation and pay attention to differences in pixel and feature levels with the use of segmentation masks.

Despite the growth in sim-to-real transfer research, to the best of our knowledge, there is still a lack of experiments that specifically apply techniques to improve aerial inspection object detection, with most works focused on the image segmentation task for more common objects. Besides that, the application of those kinds of methods to translate simulated aerial images to approximate them to real UAV images and improve downstream tasks has yet to be explored. The purpose of this work is, therefore, to experiment with these techniques as ready-to-use tools to improve object detection by translating simulated UAV images closest to real, and then give some insights for further research for improvements on the application of these methods with this aim.

III. METHODOLOGY

This section describes the experiments performed within the context of object detection in an aerial inspection setting, the arrangements to augment the original real dataset, and the evaluation of image-to-image translation strategies with the objective of discussing the following questions:

- In a real-world situation with scarce aerial inspection real data, how well can an object detection model trained only on synthetically generated images perform?
- Given that there is a gap between these synthetically generated images and the real images domain, can we apply image-to-image translation methods to approximate the simulated images to the real world and improve the detector metrics using only this synthetically generated data?
- If we have a setting where there is a small amount of real data and a larger amount of synthetically generated images, how can we combine those subsets to improve the detector's performance?

The following subsections dive into the datasets generation process, choice of object detection model, and the experiments performed to answer the questions. Fig. 3 presents a summary of the pipeline of our study.

A. DATASETS

A collection of annotated images is essential for training an object detection model. In this work, we use datasets from

FIGURE 3. Summary of the proposed methodology to study and evaluate methods to improve object detection in aerial inspections of transmission towers within scarce data scenarios. A larger dataset with synthetically generated images and a smaller dataset of collected real images are used to train Image-to-Image translation methods to approximate the simulated environment to the real target environment. Then, variations on dataset composition and training strategies are studied and compared based on the performance metrics achieved by the object detection model on real collected data - made by the authors.

two distinct sources: a small dataset created from videos captured in real life around transmission towers, here referred to as $\mathcal{D}_r$; and a second synthetic dataset, generated from an environment created on the AirSim simulator, here referred to as $\mathcal{D}_s$. Both datasets were annotated and contained the classes tower, insulator, yoke plate, ball link, clamper, and anchoring clamp. These objects were chosen for their visual characteristics and their relevance in the structure of the transmission towers.

After gathering all the images for each dataset, we performed the subset division between train, validation, and test, containing 80%, 10%, and 10% of the data, respectively. The training and evaluation procedures used only their respective subsets for the studies carried out in this methodology section. The test data was reserved exclusively to generate the final performance metrics presented in the results section. This procedure was adopted to reduce the authors' bias in choosing the final model and offer a more formally rigorous performance evaluation. Table 1 shows the number of images in each dataset and each subset.

As we aim to study the performance of a detector for the aerial inspection scenario and how the domain shift between real and synthetic data affects it, the following subsections provide more details about the characteristics of each data subset.

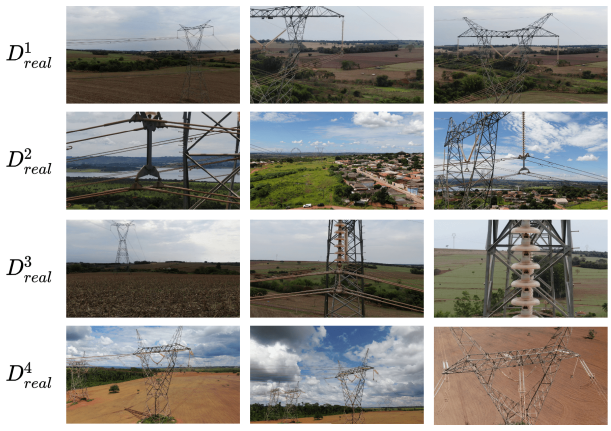


FIGURE 4. Examples of images from the training set of each real dataset. Those images were captured by the UAV - made by the authors.

1) REAL IMAGES

The real dataset $\mathcal{D}_r$ is formed by the union of 4 different datasets ($\mathcal{D}_r = \mathcal{D}_r^1 \cup \mathcal{D}_r^2 \cup \mathcal{D}_r^3 \cup \mathcal{D}_r^4$) which were obtained from videos captured on different field trips to power transmission towers. Fig. 4 contains example images of each dataset.

The images were collected flying from transmission towers and their components, varying different distances and angles. Each dataset was acquired in a different city with its particular biome and environment. The variability of vegetation,

TABLE 1. Quantity of images in each dataset: one large dataset of synthetic images and four small datasets of real images. Inside the parenthesis, the percentage of images relative to each split.

Source	Dataset		Quantity of Images on each Subset				
			Train	Val	Test	Total	
Sim	$\mathcal{D}_s$		5323 (0.8)	625 (0.1)	626 (0.1)	6574 (0.90)	
Real	$\mathcal{D}_r$	$\mathcal{D}_r^1$	45 (0.8)	6 (0.1)	6 (0.1)	57 (0.1)	700 (0.10)
		$\mathcal{D}_r^2$	62 (0.8)	8 (0.1)	8 (0.1)	78 (0.1)	
		$\mathcal{D}_r^3$	291 (0.8)	36 (0.1)	37 (0.1)	364 (0.5)	
		$\mathcal{D}_r^4$	160 (0.8)	20 (0.1)	21 (0.1)	201 (0.3)	
	Total		558 (0.8)	70 (0.1)	72 (0.1)	700 (0.10)	
				142 (0.2)			

climate, and lighting in different cities provided a greater diversity of features for the images, which can be used to evaluate the model's power for generalization.

Considering that the set of collected images could have other images very similar to each one due to the frame extraction procedure from the videos and leading this way to overfitting the training, we used a script based on the Learned Perceptual Image Patch Similarity (LPIPS) [48] to select the frames of interest. When running LPIPS, we compare consecutive frames, f and f+1. If they have a similarity smaller than 50%, then f+1 is selected. If not, then the script evaluates f+2, and so on, until reaching this minimum threshold of difference. The algorithm provided us with a dataset containing images sufficiently distinct from each other.

2) SYNTHETIC IMAGES

Based on the videos captured during the field trips, we set up a 3D virtual environment in the Unreal game engine [12]. We used the open-source software AirSim to control the UAV and capture images and Unreal to modify and create a map with the desired characteristics.

We modeled the objects of interest and inserted them into the virtual environment. Fig. 5 shows each of the considered classes, its modeled 3D object, and a real reference image.

A series of randomization rules were established to capture a set of images representative of the locations that the UAV would be flying relative to the tower. We tried to offer the same perspectives and framing that appear in our real datasets.

B. AVOIDING DATA LEAKAGE

When using the same video frames on the train, test, and validation subsets, the resulting images will contain very similar visual aspects. The videos, in this scenario, are usually recorded around a central tower. Therefore the vegetation, the tower itself, and the daylight will be the same for all frames.

To avoid the bias toward evaluating the model on frames for the training and test set that are originally from the same video, in the following studies, we do not use real images from the same video for training and testing. For example,

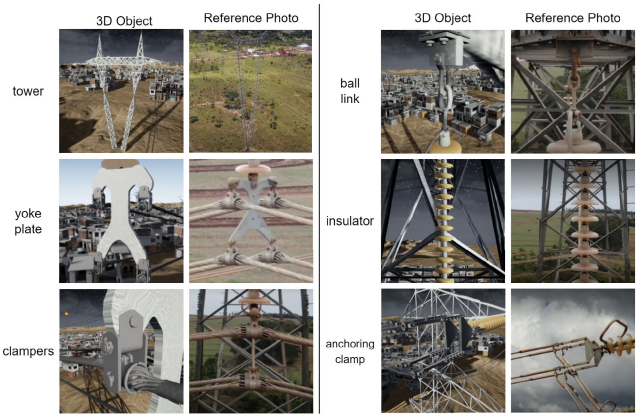


FIGURE 5. Examples of the 3D objects modeled to generate synthetic images for the dataset. Each object was based on reference photos present in the real dataset. From top to bottom, on the left column: tower, yoke plate, and clampers; and on the right column: ball link, insulator, and anchoring clamp - made by the authors.

if a certain model is trained using $\mathcal{D}_r^2 \cup \mathcal{D}_r^3 \cup \mathcal{D}_r^4$, then the test subset will only contain images from $\mathcal{D}_r^1$. Models that do not contain any real image on their training and validation steps will be tested on the whole real dataset ($\mathcal{D}_r = \mathcal{D}_r^1 \cup \mathcal{D}_r^2 \cup \mathcal{D}_r^3 \cup \mathcal{D}_r^4$), since there is no chance that the model has already seen any of those images. In addition to guaranteeing that there is no similar image leaks from training to test subsets, this process also favors models that can achieve a better generalization. Further details regarding which images are used in each experiment can be seen in Appendices A and B.

C. CHOICE OF DETECTOR

The model chosen for this project is the YOLOv5[1] [26]. It is a widely used one-stage detector based on convolutional neural networks and is specifically designed for high performance and low inference times. The version used (YOLOv5s) is adequate to run inside real-time applications, obtaining a good trade-off between size, consumed memory, processing

[1] https://github.com/ultralytics/yolov5

time, and precision. The YOLOv5s training took from 8 to 12 hours to complete on a single NVIDIA TITAN X GPU, following the number of images and the parameters shown on Appendices A, B and C. The inference time on the same GPU takes around 10ms, and around 60 ms on a SAMSUNG S10 Lite with a Snapdragon 855 chipset, a hardware similar to the one present on some UAV controllers.

The metric used for evaluating the model performance is the $mAP@[.5; .95]$ for its generability. This value shows us how well the model is performing with respect to finding the correct objects in the correct positions. This is the standard metric used in the YOLOv5 and the official metric for the COCO dataset [29] object detection challenge, which was the dataset used as the base for the weight initialization in our model. This also has been the metric reported on most papers to compare state-of-the-art object detection models, for example on [17], [18], [19], [20], [24], and [49]. The hyperparameters for all training procedures can be seen in Appendix C.

D. TRAINING WITH REAL DATA (REAL BASELINE)

We first trained and tested the YOLOv5 model with all the scarce available real data, following what would usually be done when training an object detection model in a new task with sufficient data for transfer learning. This model established a baseline for the possible improvements in performance when using real and synthetic images in combination, as explored in the following experiments. The baseline performance was affected mainly by the diverse environments and backgrounds represented in each subset of $\mathcal{D}_r$ and the dissimilarity in appearance between the base classes from COCO and the transmission line accessories.

E. STUDY A - EVALUATING THE SYNTHETIC DATA GENERABILITY AND DOMAIN GAP

Earlier works identified the difficulty of reaching state-of-the-art model performance when only synthetic data is used for training the network [50]. However, as this might be the case due to difficulties in image acquisition and annotation, a solution could be to use the scarce real data only on the testing set to emulate real-world scenarios and therefore fit only on synthetic data.

In this section, we describe experiments to identify the presence of the domain gap between simulated and real images, set a baseline for performance comparison, and try to close this gap using domain adaptation techniques based on image-to-image translation.

1) ESTABLISHING A BASELINE WITH ONLY SYNTHETIC IMAGES (STUDY A1)

In order to evaluate how well we could identify the objects of a real situation when training only with simulated data, we trained the YOLOv5 model for 300 epochs on 100% synthetic data. Then, we evaluated this model's performance on all real data available on our datasets.

2) SIM-TO-REAL TRANSLATION (STUDIES A2-A4)

Considering the availability only of simulated images and the apparent success of unsupervised image-to-image translation techniques based on GANs, we focused on their investigation to close the gap between our applications' real and synthetic domains. The following techniques were evaluated:

- CycleGAN [47] (Study A2): this method introduced the cycle consistency loss to regularize the mapping from one domain to another and back again. The architecture was based on two discriminators, which aim to differentiate the real image from the translated image to the other domain, and two autoencoders, with the objective of translating images from one domain to another. Most of the state-of-the-art domain adaptation techniques report results against this technique and consider it as a strong baseline.[2]
- Constrative Unpaired Translation - CUT [51] (Study A3): this method explores contrastive learning through the use InfoNCE [52] loss for conditional image synthesis, enforcing the network to learn an encoder that approximates image patches that correspond to each other and disassociate not corresponding patches. The proposed framework acts on unpaired images through a multilayer approach using a generator composed of two components, an encoder and a decoder, which avoids the use of inverse auxiliary generators and discriminators and simplifies the training procedure. The method is a follow-up work from the same authors of the CycleGAN.[3]
- Two-Stream Image-to-image Translation - TSIT [11] (Study A4): this approach uses a two-stream network dedicated to integrating content and style information into the generated image. The proposed method introduces two feature transformations: feature adaptive denormalization (FADE) to preserve the content spatial structure and feature adaptive instance normalization (FAdaIN) to transpose the style information, both acting in multi-scale feature levels. Also, the method claims to be simpler than other strategies since it does not depend on external constraints such as the cycle consistency loss and presents state-of-the-art results for image-to-image translation task.[4]

Since the objective of image-to-image translation models is to learn the mapping from one domain to another, and the quantity and variety are two important factors in this task, the three image-to-image translation models were trained using the training and validation subsets from the synthetic and real datasets. It was noticed that the TSIT model has more restrictions to image and crop size due to its architecture,

[2]https://junyanz.github.io/CycleGAN/
[3]https://github.com/taesungp/contrastive-unpaired-translation
[4]https://github.com/EndlessSora/TSIT

and we used the value 512 for both variables. Also, TSIT uses pairs of images from both domains, but since the real domain has fewer images that the synthetic one, we ensured that the synthetic images that did not have a pair got a random image that was already used from the real domain. As noticed, this disparity may have led to poorer results. Both CycleGAN and CUT used a value of 720 as image and crop sizes for training. All three methods were used to train translation models for 100 epochs. More details on the hyerparameters used for each experiment can be seen in the Appendix D.

After the training, an inference was performed on the whole set of synthetic images from the simulator to map them closer to the real domain, and then the object detection models were trained on 100% on the synthetic images generated by the image-to-image translation techniques and tested on 100% of the real images.

F. STUDY B - FINE-TUNING AND TRAINING WITH MIXED DATA

Another option when having only a small number of real images is to combine those images in the training step using some strategy to optimize the contribution of this data to the training. The proposed experiments explored mainly two ways to combine simulated images, adapted or not, with real images to improve the performance of the object detection model:

- Train the model first on synthetic data and fine-tune it on real data.
- Train the model with all available data at once, without a pre-training step.

1) FINE-TUNING FROM SYNTHETIC DATA (STUDY B1)

A common approach to training deep learning models with little data is to apply transfer learning. This process consists of reusing a model trained on a different but similar domain that contains a larger amount of data and then fine-tuning this model on your target domain, to incorporate the knowledge obtained from the similar task the model was previously trained to perform.

The simulated environment was modeled based on the reference objects present in real scenes and contains almost 10 times more images than the real dataset, so a pre-training on this larger dataset and fine-tuning on the smaller real dataset could improve the performance of the model and decrease the chance of overfitting.

In this experiment, the model obtained in Experiment A1, which had been trained for 300 epochs solely on synthetic data, was fine-tuned for 30 additional epochs on real data subsets and evaluated on real images. We followed the process described in Section III-B to avoid data leakage. The exact number of images used for each training step is shown in Appendix B.

In this process, all hidden layers were re-trained and no weight value was left frozen.

2) FINE-TUNING FROM ADAPTED SYNTHETIC DATA (STUDY B2)

Since applying domain adaptation techniques should approximate the synthetic and real domains, we hypothesize that it should also contribute to improving the performance in the pre-training stage.

To evaluate the contribution of the domain adaptation to this scenario, we used the weights of the network trained on Study A2, which employed synthetic data transformed with CycleGAN, and fine-tuned the YOLOv5 on real data, following the same process as Study B1.

The CycleGAN-adapted dataset was chosen based on a visual inspection and the fact that a qualitative analysis of the translation results indicated that those images were the most photo-realistic and presented fewer distortions. More on that will be discussed in Section IV, and visual examples can be seen in Appendix E.

3) TRAINING WITH MIXED DATA (STUDY B3)

Mixed training is defined as when the training set is composed of a mix of real and synthetic data. In this study, we planned to verify how adding real images to a synthetic dataset for training impacts the model's performance on the test set. In order to do so, we combined all available data for training the network.

Different from the previous experiment, where the training was performed in two stages (pre-training on synthetic and fine-tuning on real data), this experiment mixed images from both datasets on the training and validation steps. This means that for each forward pass, the neural network received as input either a real or a synthetic image, randomly defined.

The final performance evaluation was performed on the same test images as the previous experiments in order to avoid bias and compare the results following the considerations described in Section III-B and the image distribution shown in Appendix B.

4) TRAINING WITH MIXED DATA AND DOMAIN ADAPTATION (STUDY B4)

Following the same assumption that the domain adaptation on synthetic images could lead to images more closely related to real scenarios, we also performed a training mixing the real data with the simulated images adapted with CycleGAN.

IV. RESULTS

A. DOMAIN SHIFT WITH SYNTHETIC IMAGES (STUDY A)

The presented Real Baseline experiment result is the mean between the tests of four trained models, following the process detailed in Subsection III-B to avoid leakage. In this experiment, each model was trained and validated on three real datasets and then tested on the data subset that was left out. The other four models (A1-A4) were all trained and validated on sets of synthetic images and then tested on all available real images. The synthetic data from A2-A4 were translated toward the real domain. The overall results can be seen in Fig. 6.

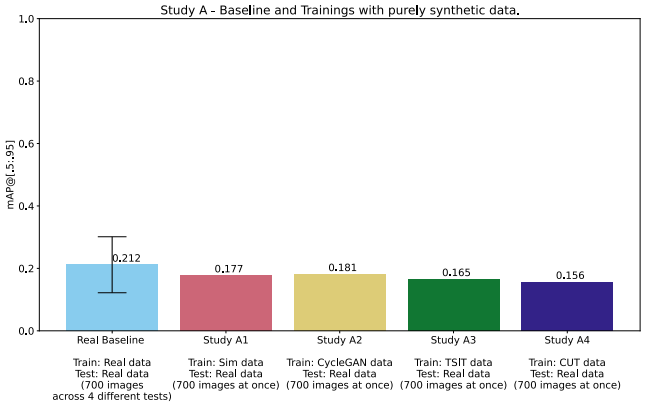


FIGURE 6. Study A results for mAP@[.5;.95] on the test set - made by the authors.

The difference between the *mAP*@[.5; .95] value achieved by the baseline training on real data, 0.212, and simulated data without any domain adaptation techniques, 0.177, makes explicit the domain gap between real and synthetic domains and reinforces the difficulty to improve a deep learning model only training on simulated data. Model A1 was trained on as much as 10 times more data than the real baseline model and performed worse, suggesting that even when increasing the number of images, training in a dataset without any resource of the real domain might not be a viable solution. The small values, even for the models fully trained on real images, also corroborate the need for a more complete dataset and put in evidence the contrast between the ideal and the real application of object detection for scenarios with scarce data.

Regarding the attempt to reduce the shift using image-to-image translation as domain adaptation techniques, approximating the simulated images to real ones, we can compare the results of the models trained only on simulated images taking Study A1 as a baseline, since it used synthetic images without any additional processing. In this scenario, the objective is to consider the influence of only the domain adaptation techniques on the model performance. As shown in Fig. 6, only the CycleGAN-adapted model presented an overall slightly better result than 0.177, with a *mAP*@[.5; .95] of 0.181. Some samples of the images generated by the domain adaptation methods can be seen in Fig. 7.

By observing the generated images, the CycleGAN method was able to approximate some background characteristics, such as vegetation and ground, without distorting too much the objects and structures that need to be detected. The other methods presented a number of artifacts that could have led the model to performing worse than it performed whereas using purely synthetic images from the simulator. When looking at images generated by CUT, we observed that the network introduced objects like insulators and tower structures into the images and mixed the objects with the background, even transforming structures into clouds. TSIT-generated images also presented instabilities with important power line components disappearing, getting blurred, or being merged with the background. Since the annotations

FIGURE 7. Images generated using image-to-image translation. From left to right, the first column represents the original synthetic images provided by the AirSim simulator, followed by the results of the CycleGAN, CUT, and TSIT methods - made by the authors.

used for each version of the synthetic images are still the same, the disappearance or creation of new objects leads to the presence of false positives and negatives that influence the *mAP* metric and have an effect on the updates of the network weights during training.

Through a visual analysis of the synthetic datasets, it was noted that the smaller objects like anchoring clamps, clamper, and ball links were the most affected in the generated images, presenting distortions, hallucinating artifacts, and even vanishing from the image. Some examples of this analysis are shown in Figures 12, 13 and 14 in Appendix E. The distributions of bounding boxes shapes (height and width percentage of the image) for each dataset were also analyzed (Figure 11) and it was confirmed that these classes (anchoring clamps, clamper and ball link) correspond to smaller objects, having the height and/or the width of their bounding boxes concentrated within the range of 5%.

Additionally, it is worth noting that YOLOv5 filters small objects after the image augmentation on the training phase: bounding boxes with width or height less than 2 pixels, aspect ratio width/height greater than 20, and less than 10% of the area before the pre-processing are disregarded), what could also be interfering on the detection performance of these classes as shown by the Recall and Precision achieved on each experiment for each class in Fig. 8. A more extensive analysis on the general filtering and pre-processing techniques commonly used by popular detectors could be interesting to be investigated by further studies. Nevertheless, for objects that take a bigger part of the images, like towers, insulators, and yoke plates, we can see that the results improved for one or multiple techniques.

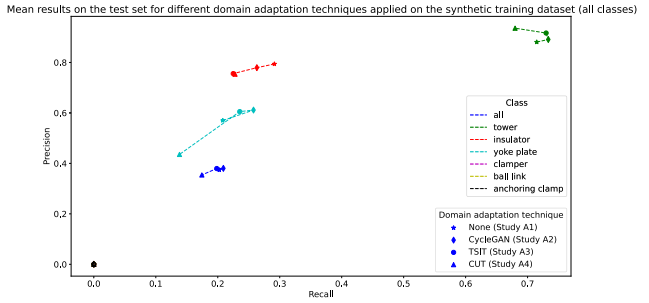


FIGURE 8. Study A Recall x Precision results for each class. The Precision and Recall presented are the mean values for each experiment considering the discussion on Section III-B - made by the authors.

A more in-depth study of how each method modified the images and their optimization for the specific case of aerial images might lead to better results, such as the proposal of heuristics to find better image-to-image pairs and the exploration of other hyperparameter settings. Besides that, a combination of images generated by different domain adaptation techniques may also be a solution to improve the variability of images present on the dataset and increase the generalization capacity of the trained models.

B. FINE-TUNING AND TRAINING WITH MIXED DATA (STUDY B)

This study analyzed what would be the best practical scenario for involving the scarce real data in training in order to maximize its contribution when allied with the simulated images or their domain-adapted versions. The *mAP*@[.5; .95] results for Study B are shown in Fig. 9.

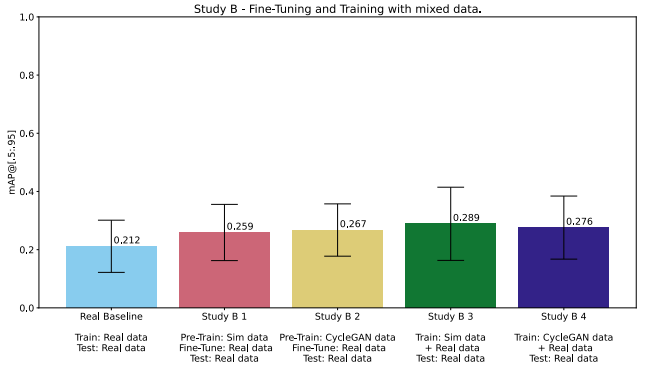


FIGURE 9. Study B results of mAP@[.5;.95] on the test set - made by the authors.

The two settings where the model was fine-tuned with real data after being pre-trained on synthetic data, Studies B1 and B2, presented a better performance than the baseline, where only real data was available. The results also indicated that the mixed training procedure from Studies B3 and B4, which combined the synthetic data and the real data for training, presented a better performance than the fine-tuning experiments.

Comparing the two mixed training experiments, Study B4, which used synthetic data adapted with CycleGAN, had a lower *mAP*@[.5; .95] than Study B3, which introduced synthetic data without any image-to-image translation technique. One possible explanation for these results is that the approximation of synthetic to the real images generated by the CycleGAN has harmed the dataset variability, as opposed to keep the domain randomization aspect that is already embedded into the simulated data.

Fig. 10 shows the Precision x Recall for all classes in Study B. The yoke plate and the insulator classes were better detected using the settings from Study B2, whereas the classes clamper and anchoring clamp had better results with the mixed training using the synthetic data presented in Study B3. Since these two classes belong to objects with smaller sizes than the yoke plate and the insulator, we assume that the CycleGAN image-to-image translation might have generated image artifacts that led to a deterioration in performance for the mentioned classes, and that the proportion of real to synthetic images used on the dataset was not enough to compensate for this deterioration.

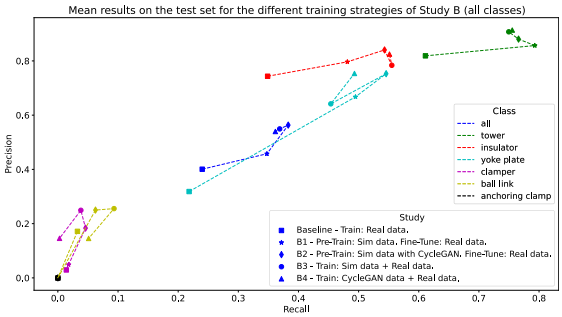


FIGURE 10. Study B Recall x Precision results for each class and training combination. The Precision and Recall presented are the mean values for each experiment considering the discussion on Section III-B - made by the authors.

One hypothesis for the non-satisfactory performance of image-to-image methods is that the diversity of picture angles and zoom on the objects could disturb their training, which can lead them to produce the previously mentioned artifacts. However, it is interesting to notice that using a proportion of only 10% real images to 90% synthetic (adapted or not), the object detection model greatly improved both precision and recall for the tower, insulator, and yoke plate, and showed at least a small gain for clamper and ball link, leading to better overall results. A more in-depth and foundational study could be performed in the future, with more sets of real and simulated data, to find the optimal proportion between synthetic and real samples in mixed dataset training.

V. CONCLUSION

In this work, we evaluated the application of object detection with scarce data for the UAV inspection of transmission towers. For mitigating the lack of data, we explored the generation of synthetic images using a simulated environment, and the application of several image-to-image translation

TABLE 2. Number of images from each dataset used on each experiment from Study A.

Experiment	Train (t)							Validation (v)							Test (e)						
	$\mathcal{D}^1_{r,t}$	$\mathcal{D}^2_{r,t}$	$\mathcal{D}^3_{r,t}$	$\mathcal{D}^4_{r,t}$	$\mathcal{D}_{r,t}$	$\mathcal{D}_{s,t}$	$\mathcal{D}_t$	$\mathcal{D}^1_{r,v}$	$\mathcal{D}^2_{r,v}$	$\mathcal{D}^3_{r,v}$	$\mathcal{D}^4_{r,v}$	$\mathcal{D}_{r,v}$	$\mathcal{D}_{s,v}$	$\mathcal{D}_v$	$\mathcal{D}^1_{r,tve}$	$\mathcal{D}^2_{r,tve}$	$\mathcal{D}^3_{r,tve}$	$\mathcal{D}^4_{r,tve}$	$\mathcal{D}_{r,e}$	$\mathcal{D}_{s,tve}$	$\mathcal{D}_e$
A 1 (simulated images)	0	0	0	0	0	5323	5323	0	0	0	0	0	625	625	57	78	364	201	700	0	700
A 2 (CycleGAN)	0	0	0	0	0	5323	5323	0	0	0	0	0	625	625	57	78	364	201	700	0	700
A 3 (TSIT)	0	0	0	0	0	5323	5323	0	0	0	0	0	625	625	57	78	364	201	700	0	700
A 4 (CUT)	0	0	0	0	0	5323	5323	0	0	0	0	0	625	625	57	78	364	201	700	0	700

TABLE 3. Number of images from each dataset used on each experiment from Study B.

Experiment	Train (t)							Validation (v)							Test (e)						
	$\mathcal{D}^1_{r,t}$	$\mathcal{D}^2_{r,t}$	$\mathcal{D}^3_{r,t}$	$\mathcal{D}^4_{r,t}$	$\mathcal{D}_{r,t}$	$\mathcal{D}_{s,t}$	$\mathcal{D}_t$	$\mathcal{D}^1_{r,v}$	$\mathcal{D}^2_{r,v}$	$\mathcal{D}^3_{r,v}$	$\mathcal{D}^4_{r,v}$	$\mathcal{D}_{r,v}$	$\mathcal{D}_{s,v}$	$\mathcal{D}_v$	$\mathcal{D}^1_{r,tve}$	$\mathcal{D}^2_{r,tve}$	$\mathcal{D}^3_{r,tve}$	$\mathcal{D}^4_{r,tve}$	$\mathcal{D}_{r,e}$	$\mathcal{D}_{s,tve}$	$\mathcal{D}_e$
B 1 1 fine-tuning	-	70	328	181	579	0	579	-	8	36	20	64	0	64	57	-	-	-	57	0	57
B 1 2 fine-tuning	51	-	328	181	560	0	560	6	-	36	20	62	0	62	-	78	-	-	78		78
B 1 3 fine-tuning	51	70	-	181	302	0	302	6	8	-	20	34	0	34	-	-	364	-	364		364
B 1 4 fine-tuning	51	70	328	-	449	0	449	6	8	36	-	50	0	50	-	-	-	201	201		201
B 2 1 fine-tuning (CycleGAN)	-	70	328	181	579	0	579	-	8	36	20	64	0	64	57	-	-	-	57	0	57
B 2 2 fine-tuning (CycleGAN)	51	-	328	181	560	0	560	6	-	36	20	62	0	62	-	78	-	-	78		78
B 2 3 fine-tuning (CycleGAN)	51	70	-	181	302	0	302	6	8	-	20	34	0	34	-	-	364	-	364		364
B 2 4 fine-tuning (CycleGAN)	51	70	328	-	449	0	449	6	8	36	-	50	0	50	-	-	-	201	201		201
B 3 1 mixed training	-	70	328	181	579	5323	5902	-	8	36	20	64	625	689	57	-	-	-	57	0	57
B 3 2 mixed training	51	-	328	181	560	5323	5883	6	-	36	20	62	625	687	-	78	-	-	78		78
B 3 3 mixed training	51	70	-	181	302	5323	5625	6	8	-	20	34	625	659	-	-	364	-	364		364
B 3 4 mixed training	51	70	328	-	449	5323	5772	6	8	36	-	50	625	675	-	-	-	201	201		201
B 4 1 mixed training (CycleGAN)	-	70	328	181	579	5323	5902	-	8	36	20	64	625	689	57	-	-	-	57	0	57
B 4 2 mixed training (CycleGAN)	51	-	328	181	560	5323	5883	6	-	36	20	62	625	687	-	78	-	-	78		78
B 4 3 mixed training (CycleGAN)	51	70	-	181	302	5323	5625	6	8	-	20	34	625	659	-	-	364	-	364		364
B 4 4 mixed training (CycleGAN)	51	70	328	-	449	5323	5772	6	8	36	-	50	625	675	-	-	-	201	201		201

techniques with their default settings to close the domain gap. Additionally, we investigated different training regimes that could lead to better detection performance in the real world.

The initial experiments, in which models were trained only using synthetic data and assessed on real-world data, showed that there could be a gap in visual information from the simulated and real images, since performance was lower than when only training with a small subset of real data. To solve for that, the image-to-image translation methods CycleGAN, CUT and TSIT were applied to the synthetic images for improving their relatedness with real images and the performance of the object detection model. It was observed that CycleGAN introduced fewer artifacts on the adapted images compared to the other methods, and aided the detector for reaching a better mAP performance than the model trained solely on the synthetic data without any image-to-image translation. However, the performance was still behind the training baseline with the subset of real samples. These results lead to the conclusion that the use of such methods is a promising approach to create more adequate synthetic data, but might not be sufficient to solely replace the need of at least a small amount of real data for training.

The main contribution of this work is the demonstration that, for the scenario where synthetic data can be generated and a small subset of real-world data is available in the context of aerial inspection, mixing both of them for training has shown to be more prone for better practical results than the application of fine-tuning. This approach increased around 36% the mAP value when compared with the baseline of only real images. The obtained results mixing domain adapted synthetic images with real data also suggest that there is still room for more optimizations on the image-to-image translation methods that could further improve the performance of object detectors. Thus we conclude that the mixing of real and synthetic data, with the addition of image-to-image translation or not, can have a great impact on scenarios with scarce real data.

TABLE 4. Hyperparameters used on YOLOv5 trainings.

NAME	VALUE
lrf	0.01
momentum	0.937
weight_decay	0.001
warmup_epochs	3
warmup_momentum	0.8
warmup_bias_lr	0.1
box	0.05
cls	0.0375
cls_pw	1
obj	0.4225
obj_pw	1
iou_t	0.2
anchor_t	4
fl_gamma	0.
hsv_h	0.015
hsv_s	0.7
hsv_v	0.4
degrees	0.0
translate	0.1
scale	0.5
shear	0.0
perspective	0.0
flipud	0.0
fliplr	0.5
mosaic	1
mixup	0.0
copy_paste	0.0
label_smoothing	0.0

As future directions for investigating the practical use of unsupervised domain adaptation in this context, a promising venue can be the proposition of heuristics for filtering the training images used by the translation models to avoid artifact generation, as well as the use of similarity metrics, like LPIPS, to choose better pairs of images to be given as input to paired models. Studies on proportion and volume of synthetic images on mixed datasets could also be interesting to improve downstream tasks as object detection and segmentation.

APPENDIX A
DATASETS USED ON STUDY A

Number of images and respective datasets used on each experiment of Study A is shown on Table 2.

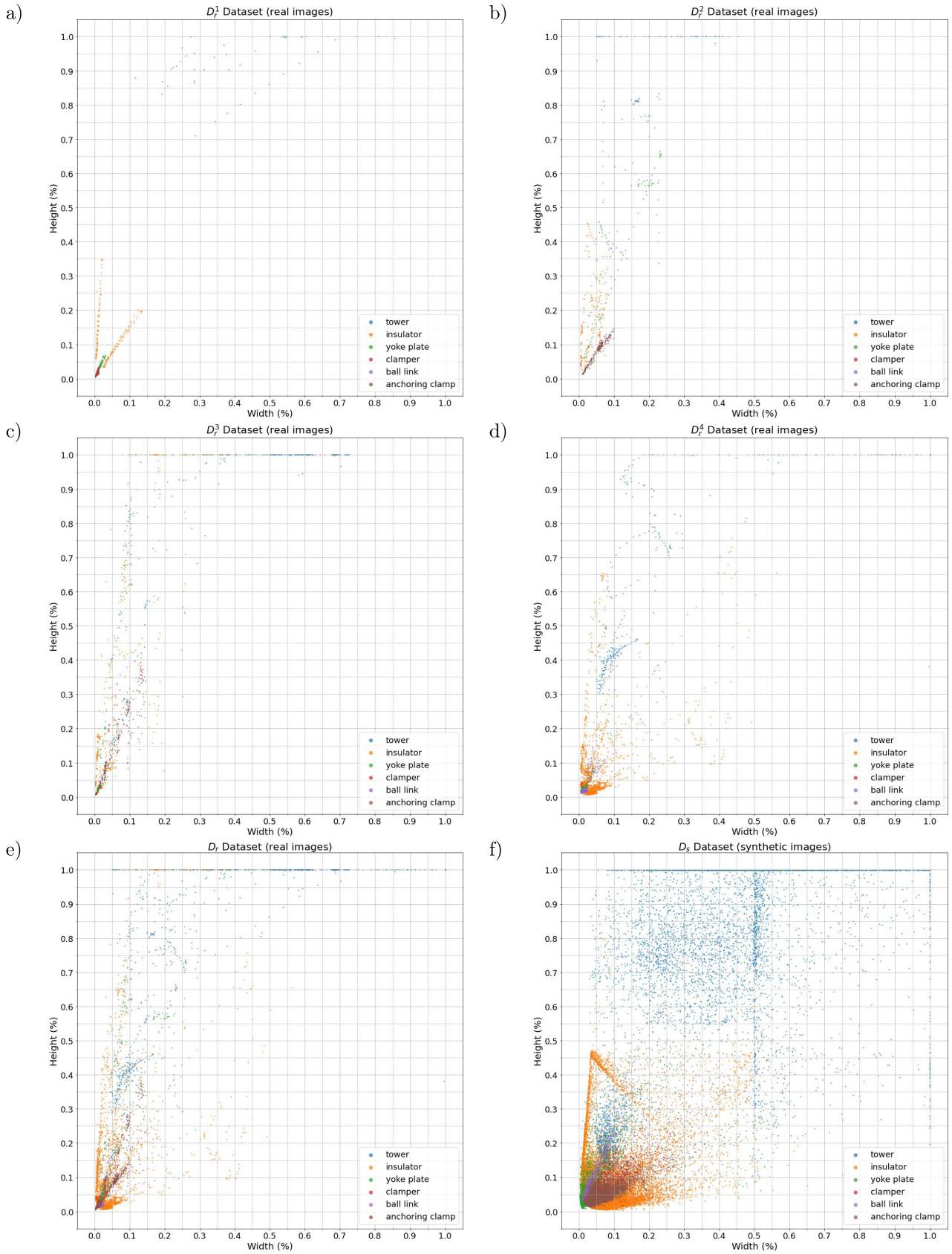


FIGURE 11. Bounding boxes' shape distribution (height and width percentage of the image) per class for each of the real datasets $\mathcal{D}_r^1$ (a), $\mathcal{D}_r^2$ (b), $\mathcal{D}_r^3$ (c) and $\mathcal{D}_r^4$ (d), for all real datasets together $\mathcal{D}_r = \mathcal{D}_r^1 \cup \mathcal{D}_r^2 \cup \mathcal{D}_r^3 \cup \mathcal{D}_r^4$ (e) and for the synthetic dataset $\mathcal{D}_s$ (f) - made by the authors.

APPENDIX B
DATASETS USED ON STUDY B

Number of images and respective datasets used on each experiment of Study B is shown on Table 3.

APPENDIX C
YOLOV5 TRAINING HYPERPARAMETERS

All YOLOv5s models were trained with the same hyperparameters in order to evaluate only the dataset influence on the results. The most relevant parameters are shown below, all other are presented on Table 4.

- Batch size: 128.
- Optimizer: SGD with initial learning rate $'lr0'$ 0.01 following a linear scheduler.
- Image size: 416.
- Epochs: 300 for training and pre-training, 30 for fine-tuning.

APPENDIX D
IMAGE-TO-IMAGE TRANSLATION TRAINING HYPERPARAMETERS

The hyperparameters described below are relative to the execution script of the official implementations [47], [51] and [11] of all three methods.

- CycleGAN
  - Epochs: 100
  - Batch size: 1
  - Optimizer: Adam
  - Beta1: 0.5
  - Learning Rate: 0.0002 with linear policy and decay after 50 iterations.
  - Pool size: 50
  - GAN loss: LSGAN
- CUT
  - Epochs: 100
  - Batch size: 1
  - Optimizer: Adam
  - Beta1: 0.5

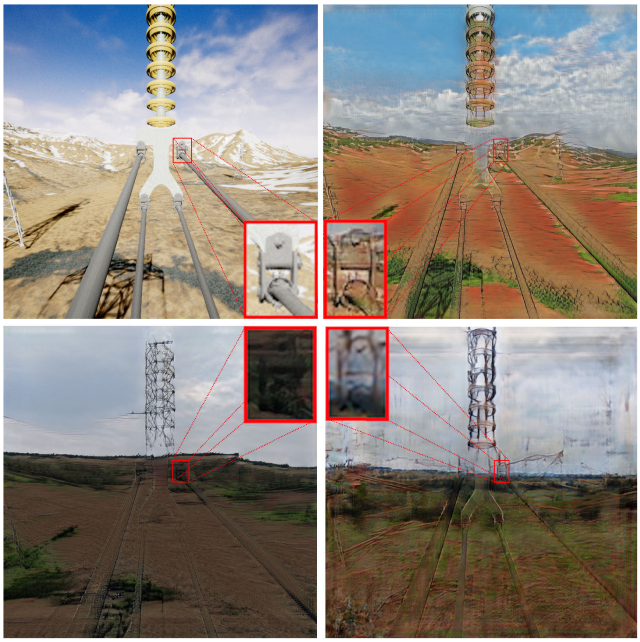


FIGURE 13. Example of distortion and vanishing of the clamper class in the outputs of the image-to-image translation methods - made by the authors.

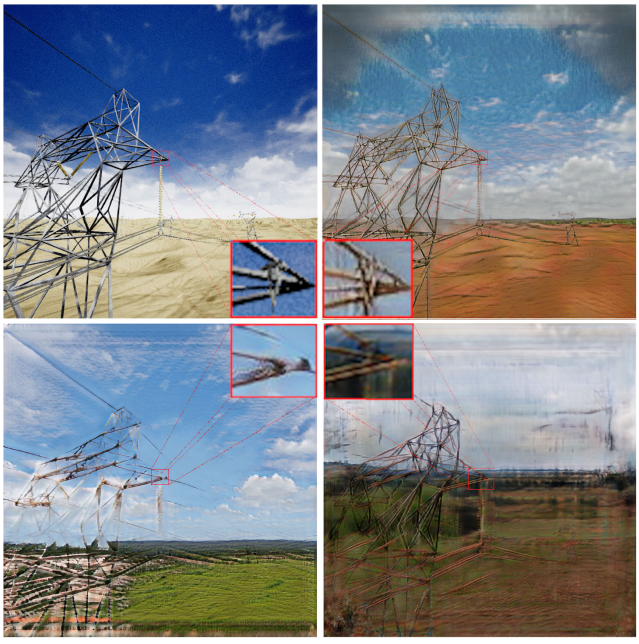


FIGURE 12. Example of distortion and vanishing of the ball link class in the outputs of the image-to-image translation methods - made by the authors.

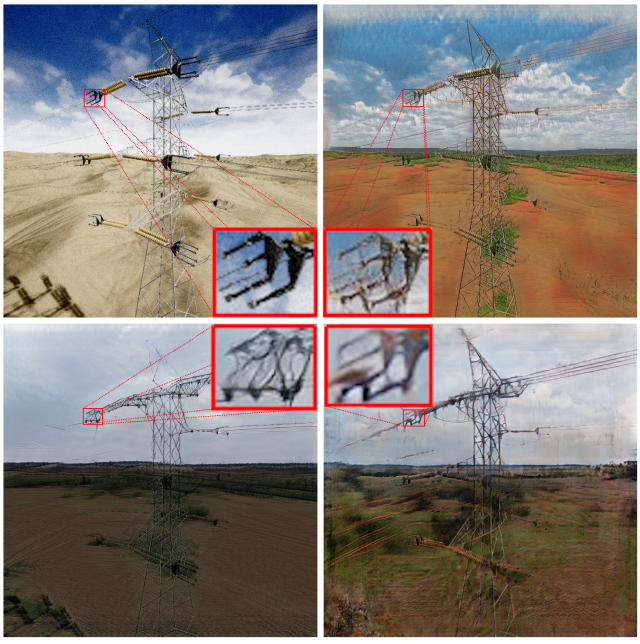


FIGURE 14. Example of distortion and vanishing of the anchoring clamp class in the outputs of the image-to-image translation methods - made by the authors.

  - Beta2: 0.999
  - Learning Rate: 0.0002 with linear policy and decay after 50 iterations.
  - Pool size: 50
  - GAN loss: LSGAN
- TSIT
  - Epochs: 100
  - Batch size: 1
  - Optimizer: Adam
  - Beta1: 0.5
  - Beta2: 0.999
  - Learning Rate: 0.0002
  - Lambda VGG: 1
  - Lambda Feat.: 1
  - Alpha: 1
  - GAN loss: Hinge
  - Number of discriminator filters in the first convolutional layer: 64
  - Discriminator iterations per Generator iterations: 1

APPENDIX E
ANALYSIS OF SMALL BOUNDING BOXES COMPONENTS

See Figures 11–14.

REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[2] J. Bian, X. Hui, X. Zhao, and M. Tan, "A monocular vision–based perception approach for unmanned aerial vehicle close proximity transmission tower inspection," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 1, p. 1729881418820227, 2019.

[3] R. Abdelfattah, X. Wang, and S. Wang, "TTPLA: An aerial image dataset for detection and segmentation of transmission towers and power lines," in *Proc. Asian Conf. Comput. Vis.*, Nov. 2020, pp. 601–618.

[4] X. Tao, D. Zhang, Z. Wang, X. Liu, H. Zhang, and D. Xu, "Detection of power line insulator defects using aerial images analyzed with convolutional neural networks," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 4, pp. 1486–1498, Apr. 2020.

[5] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "AirSim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics: Results of the 11th International Conference*. Cham, Switzerland: Springer, 2018, pp. 621–635.

[6] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3234–3243.

[7] M. Fabbri, G. Brasó, G. Maugeri, O. Cetintas, R. Gasparini, A. Ošep, S. Calderara, L. Leal-Taixé, and R. Cucchiara, "MOTSynth: How can synthetic data help pedestrian detection and tracking?" in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10829–10839.

[8] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2242–2251.

[9] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "CyCADA: Cycle-consistent adversarial domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1989–1998.

[10] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool, "Domain adaptive faster R-CNN for object detection in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018.

[11] L. Jiang, C. Zhang, M. Huang, C. Liu, J. Shi, and C. C. Loy, "TSIT: A simple and versatile framework for image-to-image translation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 206–222.

[12] Epic Games. *Unreal Engine*. Accessed: Oct. 6, 2023. [Online]. Available: https://www.unrealengine.com/en-US

[13] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proc. IEEE*, vol. 111, no. 3, pp. 257–276, Mar. 2023.

[14] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.

[15] X. Wu, D. Sahoo, and S. C. H. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing*, vol. 396, pp. 39–64, Jul. 2020.

[16] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *Proc. 18th Int. Conf. Pattern Recognit. (ICPR)*, vol. 3, 2006, pp. 850–855.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[18] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 91–99.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[21] J. Hung and A. Carpenter, "Applying faster R-CNN for object detection on malaria images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 808–813.

[22] Q. Fan, L. Brown, and J. Smith, "A closer look at faster R-CNN for vehicle detection," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2016, pp. 124–129.

[23] H. Jiang and E. Learned-Miller, "Face detection with the faster R-CNN," in *Proc. 12th IEEE Int. Conf. Autom. Face Gesture Recognit. (FG)*, May 2017, pp. 650–657.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[25] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9626–9635.

[26] G. Jocher. *Ultralytics YOLOv5*. Accessed: Oct. 6, 2023. [Online]. Available: https://github.com/ultralytics/yolov5

[27] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Jul. 2020, pp. 237–242.

[28] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.

[30] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 3213–3223.

[31] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 102–118.

[32] H. Liang, C. Zuo, and W. Wei, "Detection and evaluation method of transmission line defects based on deep learning," *IEEE Access*, vol. 8, pp. 38448–38458, 2020.

[33] Q. Mei and M. Gül, "A cost effective solution for pavement crack inspection using cameras and deep neural networks," *Construct. Building Mater.*, vol. 256, Sep. 2020, Art. no. 119397.

[34] H. Hwang, C. Jang, G. Park, J. Cho, and I.-J. Kim, "ElderSim: A synthetic data generation platform for human action recognition in eldercare applications," *IEEE Access*, vol. 11, pp. 9279–9294, 2023.

[35] Y.-T. Hu, J. Wang, R. A. Yeh, and A. G. Schwing, "SAIL-VOS 3D: A synthetic dataset and baselines for object detection and 3D mesh reconstruction from video data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 3359–3369.

[36] M. Angus, M. ElBalkini, S. Khan, A. Harakeh, O. Andrienko, C. Reading, S. Waslander, and K. Czarnecki, "Unlimited road-scene synthetic annotation (URSA) dataset," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 985–992.

[37] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, pp. 1–16.

[38] M. Fonder and M. Van Droogenbroeck, "Mid-air: A multi-modal dataset for extremely low altitude drone flights," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 553–562.

[39] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "TartanAir: A dataset to push the limits of visual SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 4909–4916.
[40] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 23–30.
[41] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018.
[42] H. Tang and K. Jia, "A new benchmark: On the utility of synthetic data with blender for bare supervised learning and downstream domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 15954–15964.
[43] *Blender—A 3D Modelling and Rendering Package*, BO Community, Blender Found., Stichting Blender Found., Amsterdam, The Netherlands, 2018. [Online]. Available: http://www.blender.org
[44] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 95–104.
[45] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3D pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3109–3118.
[46] G. J. Stein and N. Roy, "GeneSIS-RT: Generating synthetic images for training secondary real-world tasks," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 7151–7158.
[47] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2242–2251.
[48] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 586–595.
[49] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[50] S. I. Nikolenko, *Synthetic Data for Deep Learning*, vol. 174. Cham, Switzerland: Springer, 2021.
[51] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, "Contrastive learning for unpaired image-to-image translation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 319–345.
[52] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

AUGUSTO J. PETERLEVITZ received the B.Sc. degree in physics from the University of São Paulo (USP), São Carlos, Brazil, in 2017, and the M.Sc. degree in applied physics from the University of Campinas (UNICAMP), Campinas, Brazil, in 2020. He is currently a Software Analyst with the Eldorado Research Institute, Campinas, working on research and development projects. His research interests include computer vision, artificial intelligence, and synthetic data.

MATEUS A. CHINELATTO received the B.Sc. degree in electrical engineering from the University of Campinas (UNICAMP), in 2020. He has professional experience with optimized digital image processing, signal processing, and deep learning solutions. He is currently a Software Analyst with the Eldorado Research Institute, Campinas, Brazil, working on research and development projects. His research interests include AI, computer vision, and deep learning applied to image and signal processing.

ANGELO G. MENEZES received the B.S. degree in mechatronics engineering from Tiradentes University, in 2017, and the M.S. degree in computer science from the Federal University of Sergipe, in 2019. He is currently pursuing the Ph.D. degree with the Computer Science and Computational Mathematics Program, Institute of Mathematics and Computer Science, University of São Paulo, Brazil. His research interests include computer vision, neural networks, and continual learning.

CÉZANNE A. M. MOTTA received the B.Sc. degree in computer science from the University of Tocantins, Brazil, in 2017, and the M.Sc. degree in computer sciences and computational mathematics from the University of São Paulo, Brazil, in 2022. He is currently a Researcher with the Eldorado Research Institute. His research interests include algorithm design, machine learning, and computer vision.

GUILHERME A. B. PEREIRA received the B.Sc. degree in electrical engineering-robotics and automation from the Federal University of Juiz de Fora (UFJF), in 2017, where he is currently pursuing the M.Sc. degree with the Graduate Program in Electrical Engineering. He is a Software Analyst with the Eldorado Research Institute, Campinas, Brazil, working on research and development projects. His main research interests include remotely piloted aircraft (RPAs), robotics, and computer vision.

GUSTAVO L. LOPES received the B.Sc. degree in control and automation engineering from the Federal University of Itajubá (UNIFEI), in 2017, where he is currently pursuing the M.Sc. degree in computer science. He is a Software Developer in AI and robotics with the Eldorado Research Institute, Campinas, Brazil, working on research and development projects. His research interests include robotics, computer vision, deep learning, and machine learning in general.

GUSTAVO DE M. SOUZA received the B.S. degree in information systems. He is currently pursuing the M.S. degree in computer science and computational mathematics with the University of São Paulo, Brazil. From 2019 to 2020, he was a Research Fellow with the University of Campinas. He is a Researcher with the University of São Paulo. His main research interests include intelligent agents, evolutionary computing, deep learning, computer vision, synthetic data, and reinforcement learning.

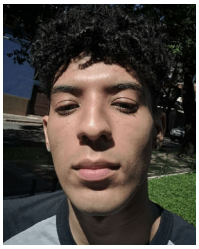
JUAN RODRIGUES is currently pursuing the degree in analysis and systems development with the Faculty of Technology and Sciences of Northern Paraná, UniFatecie, Brazil. He is an Intern in the area of visual computing with the Eldorado Research Institute, Campinas, Brazil, working on research and development projects. His research interests include mobile development and artificial intelligence.

LILIAN C. GODOY received the degree in systems analysis from Centro Universitário Salesiano, in 2007, and the Specialization degree in strategic management of information technology from the Business School São Paulo, in 2011, with double certification in international business from Universitat Politècnica de Catalunya, in 2014. She has experience managing research and development projects. She is currently the Project Leader with the Eldorado Research Institute.

MARIO A. F. F. KOLLER received the B.S. degree in industrial design from UniverCidade/RJ, in 2011, and the M.B.A. degree in strategic design from the Infnet Institute in Rio de Janeiro, in 2014. He received the Professional Specialization Certificate in UX/UI design with UX Unicorn, in 2021. He has more than 12 years of experience as a Graphic Designer. He has professional experience in design thinking, UX research, UX strategy, designing user-centered interfaces, wireframing, prototyping, and testing usability with users. He is currently focused on user-centered design. His research interests include AI, metaverse, accessibility, and gamification.

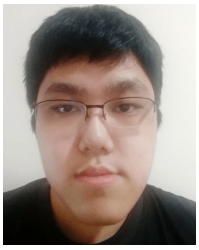
MATEUS O. CABRAL received the B.S. degree in engineering physics from the University of Campinas, in 2019, and the M.S. degree in digital system engineering from École des Mines de Paris, in 2020. He has professional experience in optimized digital image processing, deep learning solutions, and radar data processing. He is currently working with 3D visualization for multiphysics simulation. His research interests include computer graphics, computational photography, and numerical methods for physical simulation.

NICOLE E. ALVES received the B.Des. degree in game design from Anhembi Morumbi University, Brazil, in 2014, and the B.Eng. degree in computer engineering from the National Institute of Telecommunications (INATEL), Brazil, in 2021. She is currently a Software Analyst with Eldorado Research Institute, Campinas, Brazil, working on research and development projects and pursuing the M.Sc. degree. Her main research interests include web and mobile development, and robotics.

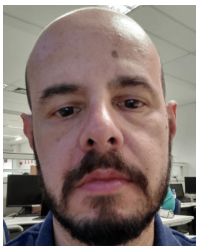


PAULO H. SILVA received the B.S. degree in computer science from the University of São Paulo, in 2022. From April 2022 to July 2022, he was a Software Engineer Intern with the Eldorado Research Institute, generating and improving synthetic datasets. From August 2022 to November 2022, he was a Software Engineer Intern with Google, contributing to the UI development in the search engine. His research interest includes character animation.

RICARDO CHEROBIN received the B.S. degree in digital games and the M.S. degree in computer science from the University of Vale do Itajaí, Brazil. He is currently a Senior Software Analyst with the Eldorado Research Institute, Campinas, Brazil, working on research and development projects. He has more than ten years of experience in developing and managing computer science projects. His main research interests include AI, game development, cyber security, and mobile development.



ROBERTO A. O. YAMAMOTO received the B.S. degree in computer science from the Federal University of São Carlos, São Carlos, Brazil, in 2021. He is currently a Software Analyst with the Eldorado Research Institute, Campinas, Brazil, working on research and development projects. His research interests include web development and AI.



RICARDO D. DA SILVA received the B.Sc. and M.Sc. degrees in electronic and mechanical engineering from the University of São Paulo (USP), Brazil, in 1990 and 2005, respectively. He has more than 20 years of experience in managing computer science projects, particularly in embedded and telecommunication systems. He is currently the Project Leader with the Eldorado Research Institute, Brazil.