Curating Datasets for Visual Runway Detection

Joakim Lindén
Saab Aeronautics
Järfälla, Sweden
Email: joakim.linden@saabgroup.com

Håkan Forsberg, *Senior Member, IEEE*
School of Innovation, Design and Engineering
Division of Intelligent Future Technologies
Mälardalen University
721 23 Västerås, Sweden
Email: hakan.forsberg@mdh.se

Josef Haddad
Saab Aeronautics
Järfälla, Sweden
Email: josef.haddad@saabgroup.com

Emil Tagebrand
Saab Aeronautics
Järfälla, Sweden
Email: emil.tagebrand@saabgroup.com

Emil Gustafsson Ek
Saab Aeronautics
Järfälla, Sweden
Email: emil.gustafssonek@saabgroup.com

Masoud Daneshtalab, *Senior Member, IEEE*
School of Innovation, Design and Engineering
Division of Intelligent Future Technologies
Mälardalen University
721 23 Västerås, Sweden
Email: masoud.daneshtalab@mdh.se

Erasmus Cedernaes
Saab Aeronautics
Järfälla, Sweden
Email: erasmus.cedernaes@saabgroup.com

***Abstract*—In Machine Learning systems, several factors impact the performance of a trained model. The most important ones include model architecture, the amount of training time, the dataset size and diversity. In the realm of safety-critical machine learning the used datasets need to reflect the environment in which the system is intended to operate, in order to minimize the generalization gap between trained and real-world inputs. Datasets should be thoroughly prepared and requirements on the properties and characteristics of the collected data need to be specified. In our work we present a case study in which generating a synthetic dataset is accomplished based on real-world flight data from the ADS-B system, containing thousands of approaches to several airports to identify real-world statistical distributions of relevant variables to vary within our dataset sampling space. We also investigate what the effects are of training a model on synthetic data to different extents, including training on translated image sets (using domain adaptation). Our results indicate airport location to be the most critical parameter to vary. We also conclude that all experiments did benefit in performance from pre-training on synthetic data rather than using only real data, however this did not hold true in general for domain adaptation-translated images.**

Keywords— avionics, safety-critical, machine learning, deep neural networks, dataset, synthetic data, domain adaptation

I. Introduction

Data-driven development methods show great promise in producing accurate models for perception functions such as object classification, detection and semantic segmentation, however most of them lack the holistic view needed for being implemented in dependable systems. Within the field of machine learning (ML), deep neural networks (DNN) are increasingly used for this purpose. A substantial part of the accuracy and robustness of a trained model is due to the data it was trained on, yet most research today focuses on model architecture development. It is therefore the intention of this paper to emphasize the dataset side of the problem by regarding the way datasets (used for model training) are being created. In this paper, we focus primarily on synthetically generated data (SD), for two reasons. Firstly, because it is a more controlled environment in which experiments can be performed, without the risk of creating hazardous aviation manoeuvres. Secondly, due to the high cost of capturing real world data (RD), alternative more practical sources of data need to be found. To affirm the relevance of these excavations, we note that it has been shown in previous work by Gaidon et al. [1] that introducing synthetically generated data into the dataset trained upon increases the accuracy of the model. There are several ways in which to structure a training scheme, e.g., initial training on synthetic data, then finalizing the training in a second step on real captured data, but one can imagine several other ways.

We propose a method for creating datasets in a structured manner, for use in safety-critical systems where a visual perception function is automated by the introduction of data-driven methods such as DNNs. This is achieved by using real-world aviation data, extracting relevant statistical artefacts, apply sampling methods, render synthetic images to include in the dataset, train and evaluate performance and finally iterate this loop to build a more complete dataset resulting in a robust model.

The overarching quest of this paper is to find solid ground for dataset creation and curation for robust model development in systems where visual perception is automated. The appropriate setting for such a system would likely be a support system from which the output is displayed for a pilot, to alleviate some of the workload during the intensive approach phase. We are not including here any tracking function where time series of outputs from this type of model would be aggregated into trackable objects, i.e. a limitation similar to that made in the CoDANN reports [21], [22]. It is highly conceivable that multiple sensor modalities (e.g. visual, infrared) could work complementary, as infrared sensors work especially well in after-dark conditions as well as in weather situations including light to moderate fog. We have decided to restrict our study to simulated visual sensors only, because of lack of sufficiently realistic sensor simulation, and hence no weather conditions including fog, nor nighttime operations have been part of this study's operating design domain.

This paper is organized as follows: In Section II, we present the steps of the curating method we devised to gain insights into the intricacies of different data sampling

This work is partially supported by Vinnova within the project SafeDeep: Dependable Deep Learning for safety-Critical Airborne Embedded Systems and partially by the Swedish Knowledge Foundation within the project Dependable Platforms for Autonomous systems and Control.

978-1-6654-3420-1/21/$31.00 ©2021 IEEE

methods and model training schemes. In Section III, we present the results of the experiments we performed to validate the method. In section IV we introduce related work and in Section V we discuss the strengths and weaknesses of this particular implementation. Finally, Section VI concludes the paper.

II. Curating Method

In this section, the different steps of the method are explained.

The first step is to identify a source of data describing the operating design domain (ODD) of interest. In our case we introduce Automatic Dependent Surveillance-Broadcast (ADS-B) system data for this purpose. For more information regarding ADS-B and how it works, refer to e.g., Ali et al. [5]. ADS-B contains lots of data from the real-world air traffic domain which is suitable for the use-case of visual runway detection.

The second step entails performing statistical analysis of the data to extract relevant statistical properties related to the ODD. In this paper we investigate three different sampling methods for acquiring these properties.

The third step is to sample synthetic data from simulated environments using parametrically controlled simulator configurations (called scenes) according to the statistical properties acquired is previous step.

The fourth step includes the training of the model, including evaluation of accuracy using a relevant metric. In our case, the task being object detection, the appropriate metric is the commonly referred AP score used by MS-COCO [13] and variants thereof.

The fifth step is to introduce variations of parameters to discern whether the trained model is robust towards these variations. In our study we vary parameters we hypothesize might impact performance. These variations are used to produce new datasets in which we vary each parameter individually, creating a test dataset for each parameter. We use these test sets to determine how far from the originally produced training set our model will work.

The sixth and last step includes using the results from the variational experiments and expand the training dataset in the appropriate dimensions. This step is preferably iterated to build a subsequently more complete training dataset, eventually resulting in a more robust model.

Up until this point, we have been working entirely in the simulated domain. There is a serious question one need to ask here, is this of any use in the real world?

To answer this question we have put effort into validating the applicability of using simulated visual imagery in real-world scenarios. This investigation is further expanded to include domain-translated synthetic images using domain adaptation techniques [2]. In the following sections, we go into detail of each of these areas to describe the method in the context of our use-case.

A. Sampling methods for accurate data distributions

There are several simulated environments that can be used for sampling of images. One of them is X-Plane [3]. In X-Plane many parameters are configurable including scenery (latitudes from 60 degrees south to 74 degrees north), weather, aircraft position and attitude to name a few. To decide what subspace of images to sample from in this rich simulated world, real-world references are needed. We therefore start by collecting positional data from real-world commercial aircraft, ADS-B, where positional information is emitted regularly from the majority part of commercial aircraft. OpenSky Networks [4] hosts a historical database with a convenient querying interface. ADS-B data is not directly useful; it needs to be filtered to only include landing scenarios (for the scope of this work). Firstly, a cylindrical crop-out centered on the desired runway was defined, with a radius of about 16.6 km (9 nautical miles, NM). Secondly, a vertical descent of at least 2.5 m/s (500 ft/min) was added along with a maximum altitude of 3000 m (10000 ft) to exclude aircraft taking off and cruising. Finally, an ILS-condition discarding tracks not inbound for the particular runway was applied, along with a final outlier filter for anomalous height data, resulting in a clean set of ADS-B points from inbound aircraft using Instrument Landing System (ILS) or Visual Flight Rules (VFR) approaches. With this data point cloud, we analyze the distribution of these points to understand how to accurately draw samples from it. Three different methods are investigated. For each method, we create a dataset of images, train a model and measure its accuracy.

The sampling of different scenes is achieved using the Scenic probabilistic programming language developed by Fremont et al. [6], where scenes can be defined in a parametric way. Scenic can then be asked to draw samples from the defined scene, whilst randomizing the parameters of the scene according to set intervals and distributions. For example, the aircraft position can be one such parameter; weather and time-of-day are other examples. Scenic communicates directly with the appropriate simulator backend, X-Plane 11 for aviation or CARLA [7] for automotive in our case, which is responsible for rendering the scenes.

a) Sampling method 1: Continuous Normal

In this method, we divide the filtered dataset into bins based on distance to runway, each bin spanning 1NM of distance. A total of eight bins are created. For each bin i we estimate the mean and standard deviation of the altitude and lateral position.

Looking at the lateral position (i.e. perpendicular distance to extended centerline), see Fig. 1, we can fit a continuous polynomial function for the mean $\mu_{Lat}(d)$ and standard deviation $\sigma_{Lat}(d)$ as a function of distance to runway d using linear regression. When we create the dataset we randomize d using uniform sampling, and from this we use said functions to find the representative $\mu_{Lat}(d)$ and $\sigma_{Lat}(d)$ for this distance and then finally draw the aircraft's lateral and vertical position from a normal distributions $N(\mu_{Lat}(d), \sigma_{Lat}(d))$ and $N(\mu_{Alt}(d), \sigma_{Alt}(d))$ respectively. A total of 8000 samples are drawn using Scenic, which configures X-Plane with a scene for each sample, resulting in a unique image each time. Annotation information of the runway's position in the image is calculated at the time of generation and included in the dataset.

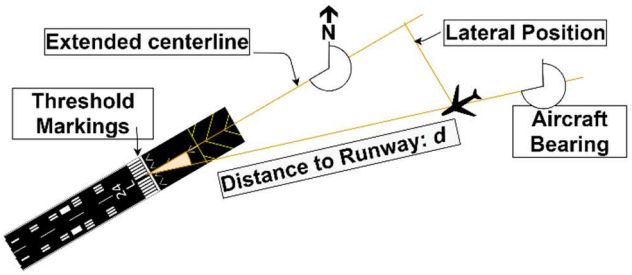

Fig. 1 Definitions of used notation for aircraft position and distance to runway.

b) Sampling method 2: Discrete Normal

This method divides the airspace into the previously described 1 NM bins. Similar to method 1 we calculate the discrete (μ^k_{Lat}, σ^k_{Lat}) as well as (μ^k_{Alt}, σ^k_{Alt}) for each bin *k*. The difference is in using the discrete μ^k and σ^k values to draw samples from bin *k*, so for each bin we:

- Randomize runway distance *d* using uniform sampling within the bin.
- Use the representative μ^k_{Lat} and σ^k_{Lat} for bin *k* and then draw the aircraft's lateral position from the normal distribution $N(\mu^k_{Lat}, \sigma^k_{Lat})$.
- Similarly, use the representative μ^k_{Alt} and σ^k_{Alt} for bin *k* and draw the vertical position from a normal distribution $N(\mu^k_{Alt}, \sigma^k_{Alt})$.
- Repeat until a sufficient number of samples has been created for bin *k*, such that the total number of samples is equal to that of other sampling methods, 8000 images in total.

c) Sampling method 3: Perturbation

This method randomly selects an existing ADS-B point and perturbs its position with a random offset (uniformly distributed within a box of size 40 m vertically and 40 x 40 m in the horizontal plane).

TABLE I DATASETS FOR SAMPLING METHOD EXPERIMENTS.

ID	Dataset context	Size	ADS-B data source
DS1	M1: Continuous Normal sampling	8k	**January** 2021, LPPT (Lisbon Airport)
DS2	M2: Discrete Normal sampling	8k	**January** 2021, LPPT
DS3	M3: Perturbation	8k	**January** 2021, LPPT
DS4	Reference: Raw (Raw ADS-B points)	8k	**January** 2021, LPPT
DS5	Small Test (raw ADS-B points)	4k	**February** 2021, LPPT
DS6	Large Test (raw ADS-B points)	24k	**March-April** 2021, LPPT

d) Reference Method: Raw

For reference, an equal sized dataset was created from the raw ADS-B data points, without any augmentation or randomization. In all these sampling methods (used for creating DS1-DS6), other parameters (listed in Table II) were kept fixed at the following values:

- Time of Day: 14.00 (local time)
- Pitch: 0 degrees
- Roll: 0 degrees
- Yaw: Aligned with runway
- Clouds: None
- Number of sampled images: 8000
- Aircraft Type: AirBus A320

Datasets used in this experiment are listed in Table I.

One model was trained for each dataset DS1-DS3 corresponding to sampling methods 1-3 and one model was trained on DS4 corresponding to the reference method. These four models were then tested on two datasets, DS5-DS6, both using the reference sampling method where ADS-B points are taken without augmentation.

B. *Parameter impact scan*

The parameters considered in this paper are, apart from the geospatial coordinates discussed above, time of day, attitude angles, weather (clouds), and finally airport location. The impact of said parameters was evaluated using the following approach.

TABLE II DATASETS USED FOR PARAMETER SCAN EXPERIMENTS.

ID	Parameter varied	Parameter Range	Distribution type
DS7	Time of Day	06.30 to 19.00 local time	Uniform
DS8	Attitude Roll	-7 to +7 degrees	Uniform
DS9	Attitude Pitch	0 to 10 degrees above horizon	Uniform
DS10	Attitude Yaw	-20 to +20 degrees from runway heading	Uniform
DS11	Weather Clouds	Category 0 to 5 (X-Plane's definition)	Discrete Uniform
DS12	Airport	OTHH (Hamad International Airport)	N/A
DS13	Airport	LFPO (Paris Orly)	N/A

First, two models were trained on datasets from experiment A yielding the highest overall accuracy (in our case method 3 (DS3)), and using the reference method (Raw, DS4). Both these models were then exposed to the datasets DS7-DS13 listed in Table II in turn and the accuracy measures on each set were collected. Note that the trained models have not been shown any image data containing these augmentations during training, meaning we expect to see a drop in performance if the test dataset deviates significantly from what it has been trained on.

C. Synthetic real data mixing

As previous work has shown, synthetic data may be used to improve the performance of deep learning models on real-world applications [1]. We leverage the SD in our experiments for object detection using the Faster RCNN [8] detector with Feature Pyramid Network backbone [9] in the aviation and automotive domain for detecting runways respective car detection. The result from the automotive domain is included in this paper since some of the experiments could not be performed in the aviation domain due to limitations of the used datasets. Also, detecting control ground vehicles are of importance for diversification of the runway environment.

The SD is used to pre-train the detector which is then fine-tuned and evaluated on RD. Our SD in the automotive domain was sampled from the CARLA car simulator [7] with domain randomization in mind and furthermore automatically generated 2D amodal bounding boxes for the cars in each scene. The RD data in the automotive domain consisted of the Cityscapes [10] and KITTI [11] datasets with accompanying amodal 2D bounding box annotations. In the aviation domain, we had 10 video sequences of an aircraft landing at Stockholm Skavsta Airport (ESKN), recorded from the perspective of the aircraft. One frame per video sequence second was extracted and the runways were manually annotated with 2D bounding boxes.

In the automotive domain, we used a total of 8000 synthetic images that were used to pre-train the detector. This pre-trained state was then fine-tuned on a smaller portion of the RD. For the Cityscapes dataset, we evaluated the effect of training with the addition of SD by performing a cross-validation schema across the 21 cities we had annotations for. 5 cities were used for training, 3 for validation and 2 for testing in each iteration. We performed a similar cross-validation schema for the data in the aviation domain but across the 10 video sequences, where 7 were used for training, 2 for validation and 1 for testing. For the KITTI dataset, we split the train, validation and test sets using cross-validation over all samples, where 800 samples were used for training, 748 for validation and 748 for testing. The validation set was used to checkpoint the weights during training that performed the best on that portion of the data, and the size of the training set were intentionally smaller than the size of the available samples in the automotive domain to better reflect the dataset size one may have available in a domain where a large dataset is difficult to come by.

D. Domain adaptation methods for data augmentation

Due to the reality gap between the synthetic and real-world data, we use a cycle consistent generative adversarial network framework to increase the realism of the images as a means of domain adaptation (DA). In our data preparation phase, the CycleGAN framework [12] is trained to map the domain of the SD to the RD domain, which can be used to create a domain adapted synthetic dataset, which we denote as SD(DA) in this paper. This results in two generators, generator A which learns a mapping from the synthetic domain to the real-world domain, and generator B which learns the mapping in the opposite direction. These mappings were learned for both real-world data sets in the automotive domain but were omitted in the aviation domain due to the limitations of the datasets in that domain. For the Cityscapes dataset, we utilized the extended version of the dataset which does not contain bounding boxes to get a better representation of the real-world domain. While it may seem counterintuitive to use a large real-world dataset when the original problem is the difficulty of collecting real-world data, the data used for learning the domain adaptation does not have to be annotated.

III. Results

A. Sampling methods for accurate data distributions

Sampling method 1 (Continuous Normal) measured the mean and standard deviation in each of the eight bins of ADS-B data points, for both the altitude and the lateral displacement from the extended centerline. For lateral displacement it is easy to find a suitable model for how the mean and standard deviation changes over distance *d* to the runway, see Fig. 2. We note here that the mean displacement converges to about -8 m, which seems to be an offset issue with accurately representing the centerline position. For the altitude parameter the mean follows a steady linear decline, whereas the standard deviation does not show a clear correlation with distance, see Fig. 3.

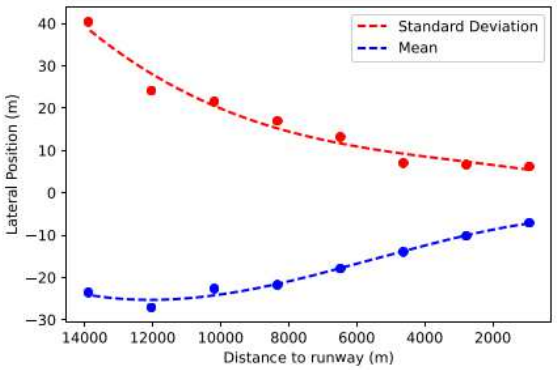

Fig. 2 Third degree polynomial Regression fit of standard deviation (top) and mean (bottom) of lateral displacement (w.r.t. extended centerline) as functions of distance to runway.

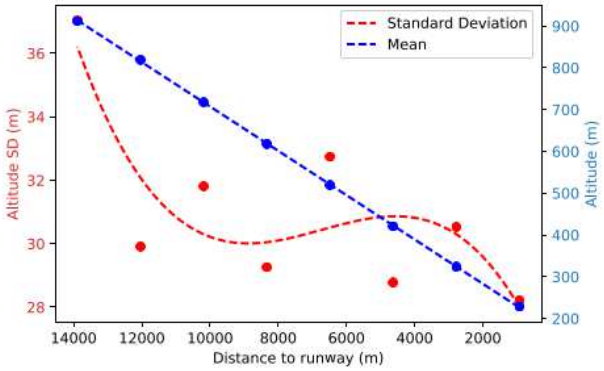

Fig. 3 Polynomial Regression fit of mean altitude (first degree polynomic) and third degree polynomial fit of standard deviation of aircraft altitude as functions of distance to runway.

Fig. 4 - Fig. 6 show the lateral spread of the three generated datasets as well as the underlying ADS-B data set used as input for all sampling methods. Note that sampling method 3 (Perturbation) shows a wider spread close to the runway than the other ones. It also captures some of the late turn approaches, which is not the case in methods 1 and 2. Method 2 (Discrete Normal) is showing a greater spread at large distance compared to method 1 (Continuous Normal). Method

3 does not generate the mirrored points (large negative values at great distances) since they are not found in the original dataset. Methods 1 and 2 implicitly assume a symmetric spread since we model the spread with a normal distribution.

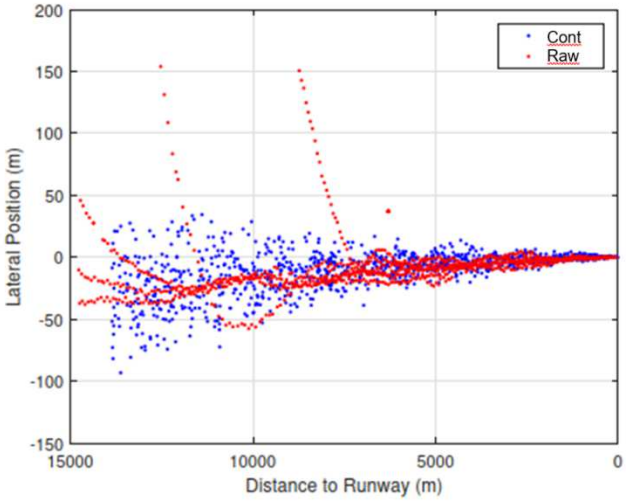

Fig. 4 Sample method 1 (Continuous Normal) generated dataset (blue) compared to the reference method (Raw) generated dataset (red). Diagram shows lateral spread as a function of distance to runway.

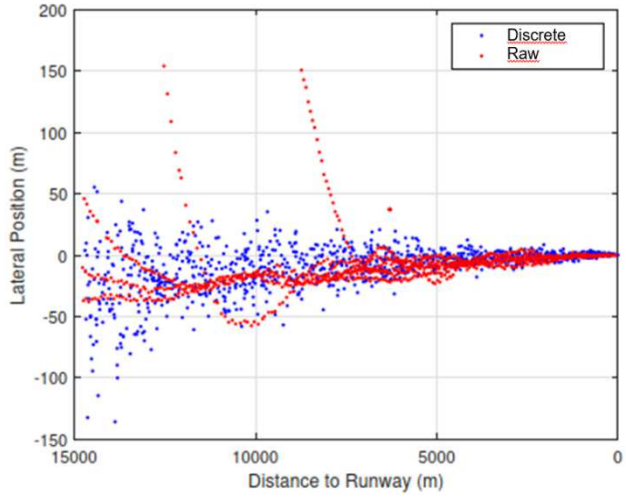

Fig. 5 Sample method 2 (Discrete Normal) generated dataset (blue) compared to the reference method (Raw) generated dataset (red). Diagram shows lateral spread as a function of distance to runway. Note the large lateral spread far from runway.

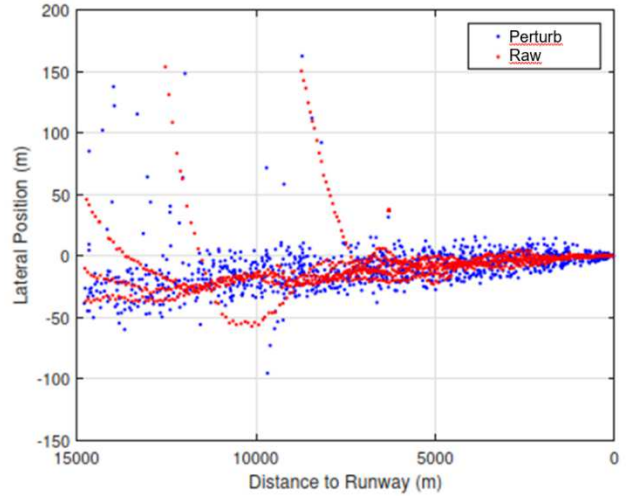

Fig. 6 Sample method 3 (Perturbation) generated dataset (blue) compared to the reference method (Raw) generated dataset (red). Diagram shows lateral spread as a function of distance to runway. Note the large lateral spread close to runway.

Model Performance

The neural network used throughout this work is the Faster RCNN two-stage detector described earlier. The model was initialized in the same way for all experiments in this section (i.e. the ResNet-50 [14] backbone is pretrained on ImageNet dataset [15]). *Fig. 7* and *Fig. 8* show the detection accuracy results measured by the MS-COCO [13] Average Precision (AP) metric commonly used for object detection. AP is the average precision over multiple Intersection-over-Union (IoU) levels spanning from 0.5 to 0.95 in 0.05 increments. AP(distant) refers to objects of small size (< 32x32 pixels), i.e. the detections made at great distance from the runway. Similarly, AP(close) refers to the AP of objects of large size (> 96x96 pixels) and hence include detections made in closer proximity to the runway. There is a small increase in accuracy for sampling method 3 (Perturb) in both datasets when focusing on the close detections, AP(close), which could be due to its comparatively wider spread of data points in this distance range. This effect is similarly seen in the AP(distant) for method 2 (Discrete Normal) which shows the widest spread of data of all sampling methods at large distances. This leads us to believe that great variety in the dataset is important for the model to generalize.

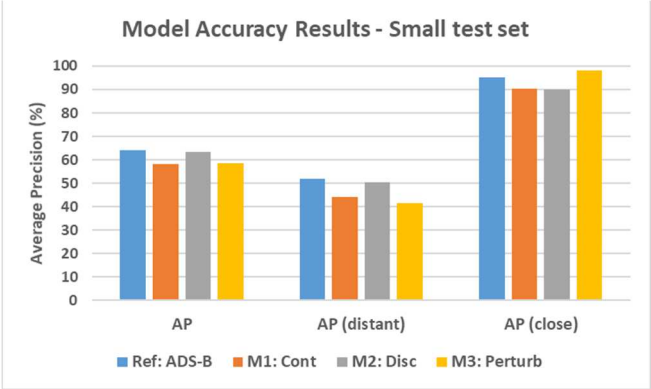

Fig. 7 Comparison of the four models (same network trained on datasets DS1 – DS4) tested on dataset DS5, which is similar to DS1 but independently prepared.

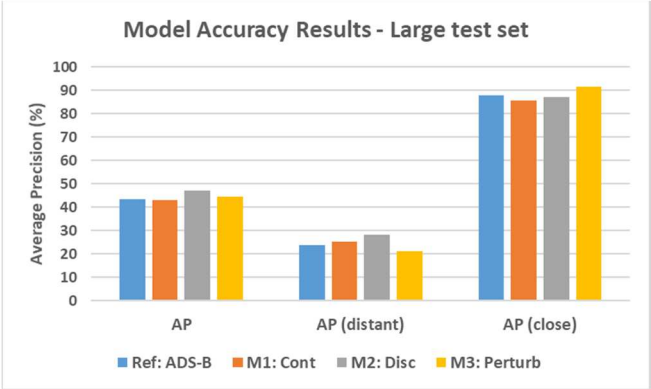

Fig. 8 Comparison of the four models (same network trained on datasets DS1 – DS4) tested on dataset DS6, which is similar but larger than DS1, independently prepared.

It should also be mentioned that in general our trained detection models perform better with larger objects, which is most likely due to the IoU threshold criteria being harder to meet with smaller objects (1 pixel offset on a small object reduces the IoU more than in the case of a larger object).

B. *Parameter impact scan*

The impact of varying different parameters is shown in the diagrams below, Fig. 9 and Fig. 11. Note that we are testing our two trained models on data with parameter variations it has never seen. Starting with the airport location, we see a *dramatic* drop in performance (AP) when shifting from Lisbon Airport (LPPT) to Paris-Orly (LFPO) and even worse when testing with Hamad International Airport (OTHH), where performance plummets to fractions of that of the reference airport. In Fig. 10 we include visual samples of the three airports as rendered in X-Plane and subjectively one might argue that LPPT and LFPO are more similar than LPPT and OTHH.

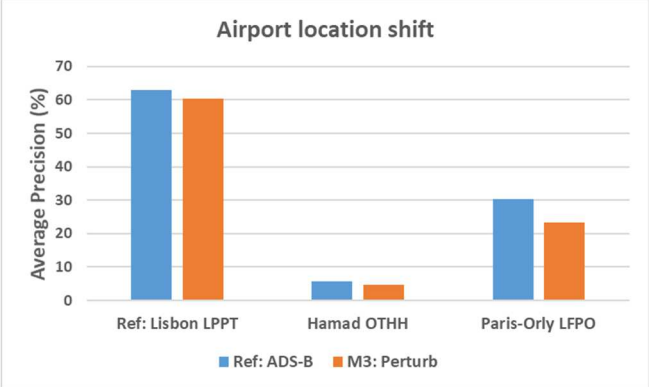

Fig. 9 Two models have been trained on datasets DS3 and DS4 respectively and then both tested on datasets DS12 (Hamad Airport) and DS13 (Paris-Orly Airport).

Considering the other parameter variations, the results are far less dramatic. Although several parameters are clearly affecting the performance, not all are equally important from an AP score drop point of view. If breaking this down further the attitude variations were found to contribute to dataset diversity at close distances, whereas environmental conditions (clouds, time of day) had a greater diversifying effect over larger distances. For more details, see Tagebrand and Gustafsson Ek [16].

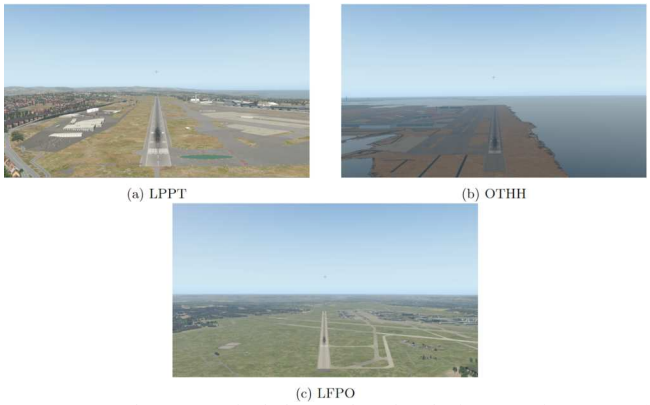

Fig. 10 For reference, we include samples of each airport environment. (a) is sampled from Lisbon Airport, (b) from Hamad International Airport and (c) from Paris-Orly Airport.

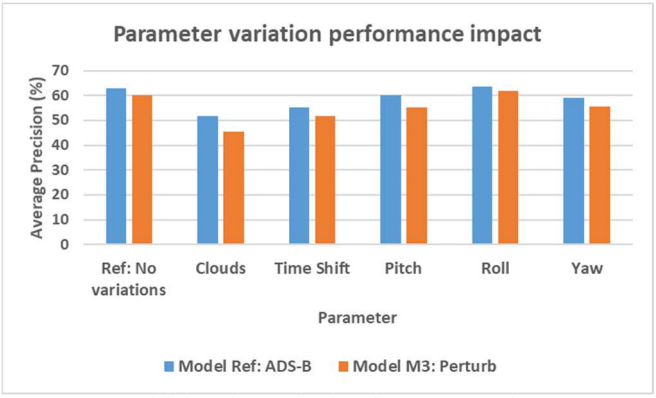

Fig. 11 Two models have been trained on datasets DS3 and DS4 respectively and then both tested on datasets DS7 – DS11.

C. *Synthetic real data mixing*

The results of mixing the real-world data with the corresponding synthetic data can be seen in Table III represented as the mean AP score achieved by the detector. Table III also contains the results from training with only real-world data as a baseline reference. We observe an increase in performance across all datasets when pre-training on the synthetic datasets before fine-tuning, averaged on the 10 cross validation iterations. Fig. 12, Fig. 13 and Fig. 14 illustrate the distribution of the performance (from the cross-validation scheme) of the detector using boxplots for the two car datasets and the data set in the aviation domain. We can conclude that the performance difference in the automotive domain in terms of the median AP is statistically significant under a 2-sided Wilcoxon signed-rank test at $p = 0.05$. The null hypothesis that their median performance is the same can therefore be rejected with confidence. However, the performance improvement that we observe in the aviation domain is not statistically significant under the same statistical hypothesis test. More details on these results are presented by Haddad [17].

TABLE III MEAN AVERAGE PRECISION RESULTS OF SYNTHETIC DATA AND DOMAIN ADAPTED DATA EXPERIMENTS.

Dataset	**RD**	**SD + RD**	**SD (DA) + RD**	**Legend:** **RD:** Real Data **SD:** Synthetic Data **DA:** Domain Adaptation transformed
Aviation data	78.53	**79.47**	N/A	
Cityscapes	45.36	**47.17**	46.19	
KITTI	62.97	63.41	**63.63**	

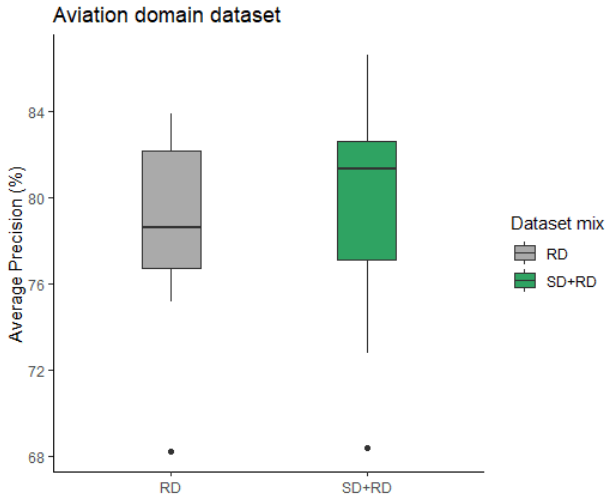

Fig. 12 Average Precision results for aviation domain dataset (across all 10 experiment folds in cross-validation). For this dataset, it was not possible to assert the performance increase with statistical significance.

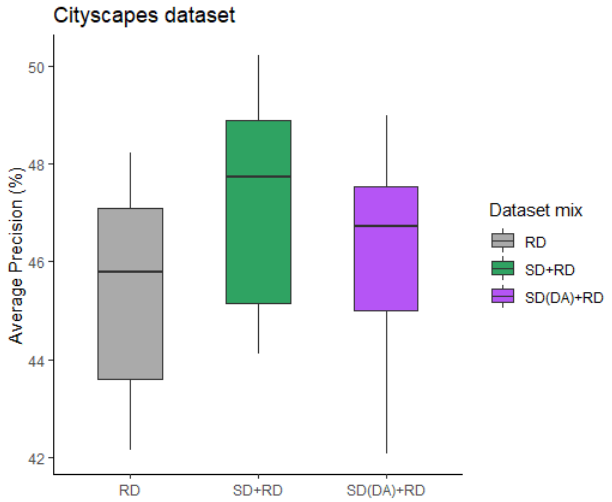

Fig. 13 Average Precision results for automotive domain Cityscapes dataset (across all 21 experiment folds in cross-validation). For this dataset, it was possible to assert the performance increase with statistical significance when comparing RD and RD+SD models (gray vs green box).

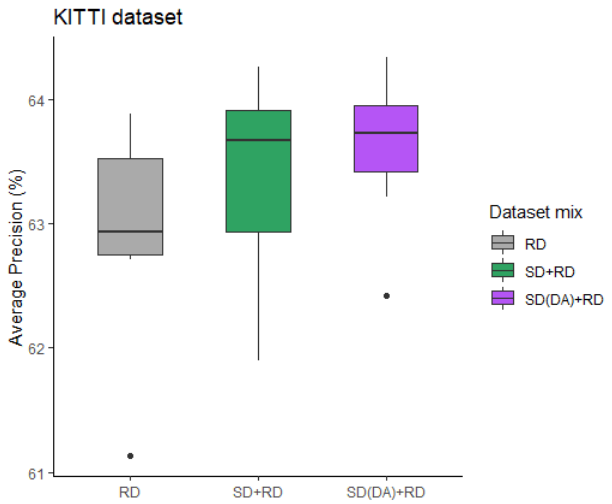

Fig. 14 Average Precision results for automotive domain KITTI dataset (across all 10 experiment folds in cross-validation). For this dataset, it was possible to assert the performance increase with statistical significance when comparing RD and RD+SD models (gray vs green box).

D. Domain adaptation methods for data augmentation

Table III shows how using the domain adapted synthetic data (SD(DA)) for pre-training compares to not using any DA (SD). We observe a decrease in performance on the Cityscapes dataset and an improvement in performance on the KITTI dataset, which is also observed in Fig. 13 respective Fig. 14 when comparing the green and the purple box plots. Although we observe an improvement when utilizing DA on the KITTI dataset, we cannot reject the null hypothesis that the median performance between using and not using DA on the synthetic data is the same.

Some visual examples of domain adaptation are shown for Synthetic CARLA domain to real-world KITTI domain (Fig. 15) as well as Synthetic CARLA domain to real-world Cityscapes domain (Fig. 16) for visual reference.

Fig. 15 Domain adaptation transformation from simulated environment with the CARLA simulator (top) to the real-world captured KITTI domain. Presumably, black cars are more common in real-world German cities than in our simulations where vehicle color was uniformly randomized.

Fig. 16 Domain adaptation transformation from simulated environment with the CARLA simulator (left) to the real-world captured Cityscapes domain (German cities).

IV. Related Work

Fremont et al. [6] propose the Scenic probabilistic language and describe the use of this to find corner cases in the automotive domain. Schäfer et al. [4] describe the development of OpenSky, which is a network of ADS-B sensors, based on low-cost hardware and built with the intention of helping researchers perform big data analysis within the aviation domain. Adrien et al. [1] propose a real-to-virtual cloning method resulting in the Virtual KITTI dataset. They also address training on synthetic data, fine-tuning on real-world data and find it to improve performance

in their automotive experiments. Domain adaptation is investigated in several fields. Wang et al. [18] show successful adaptation of visible images to thermal infrared in automotive traffic scenarios, Palladino et al. [19] use the same technique for adapting related image domains in MR medical imaging, whereas Choi et al. [20] presents StarGAN, allowing multi-domain image-to-image translations. Richter et al. [23] use domain adaptation to enhance the photo-realism of computer-generated graphics by extracting extra information from the generated images such as surface normals, distance, albedo, glossiness and extra lighting information.

V. Discussion

It is clear from our results that the airport location is the one parameter most important to vary to achieve a generalizing detection model for our use-case. Other parameters include weather effects like clouds and the dusk-and-dawn-like effects we see when varying the time of day of our flights.

It should be noted that the ADS-B data sourced for generating these datasets is applicable to Airbus A320 aircraft only. It should be tested whether these results still hold for other aircraft types. Weather and time of day show greater impact on distant detections, attitude more so on close-up scenarios. Other environmental conditions than clouds and time-of-day should be considered, in particular rain and fog. Rain was in the scope of our initial effort, but due to unforeseen simulator issues, rainy images did not render correct textures and hence this part had to be discarded. Extending the cloud variation to include more sinister weather conditions like different types of fog would add more important data to our parameter scan.

Although the investigated sampling methods did not incur large differences in performance when compared, they importantly provide the means for applying the curating method to our use-case by parametrizing the input data. Paired with parametrically controlled simulation scenes we can start the iterative process of completing the dataset for our use-case. As found in our parameter scan, the airport location parameter was the lowest hanging fruit, and with relative ease we can teleport to a new location, in the sense that we use the statistical properties (distributions) drawn from one ADS-B dataset (belonging to one airport) and then apply on a different airport, enabling the use of this method for data generation also where ADS-B data is more scarce. We were also able to, using the results from the sampling method models, hypothesize that great variation in position is important for training a more general model.

Synthetic data collected from simulators showed promising results in both aviation and automotive domains, even though the performance improvement in the aviation domain was not statistically significant in our experiments.

We hypothesize that we would observe more significant performance improvements if we evaluated the detector on real-world aviation data from a runway that we have not fine-tuned the detector on already for example. One can argue that doing so is a more realistic real-world scenario since we cannot assume that we will have access to images of all runways in our training set if we were to deploy such a model in the future.

Domain Adaptation as a data augmentation method did not show improved results, which might be due to problems getting the model to converge well enough for the domain translation step. We did observe problems with some images being translated into non-existing scenes, such as cars melting into the pavement or vanishing backgrounds and buildings. We did explore other methods for DA initially but resorted to CycleGAN due to availability and its success in previous work. DA using GANs is nevertheless a promising field and deserves being studied in the future in the context of increasing realism of synthetic images for object detection.

VI. Conclusions

The method of scene randomization works really well in the automotive domain, due to the ease of controlling the simulator environment. The Scenic tool for handling scene randomization is an idea worth expanding into other domains. We have successfully done so for the aviation domain, by including several aircraft and environment parameters, however extending this to also control ground vehicles, other aircraft and other scene-impacting parameters would allow further diversification of the runway environment.

An obvious next step is to generate a diverse training dataset where many of these parameters are varied, and verify whether this in fact does boost detector performance. The sampling methods investigated in this paper have parameters such that we can control the width of the approach flight paths to increase diversity where needed.

Using this more diverse synthetic dataset for the aviation domain would enable revisiting some of the other experiments, resulting in more precise outcomes. Sourcing more real-world data from the aviation domain would greatly benefit such future experiments.

We proposed a data curating method in this paper with steps to systematically and iteratively build a complete dataset for a specific use-case's ODD in the synthetic data domain. We showed that training on synthetic data always improves the performance of the model, but real-world data will still always be needed as a last fine-tuning step. Domain adaptation is a cumbersome tool for augmenting images and this technique did not show significant improvements to our detection results, but further investigation is needed before any firm conclusion can be reached regarding this method.

VII. References

[1] Gaidon, A., Wang, Q., Cabon, Y., & Vig, E. (2016). Virtual worlds as proxy for multi-object tracking analysis. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4340-4349).

[2] Wang, Z., She, Q., & Ward, T. E. (2021). Generative adversarial networks in computer vision: A survey and taxonomy. ACM Computing Surveys (CSUR), 54(2), 1-38.

[3] X-plane, [Accessed July 9, 2021], Laminar Research, 2021. [Online]. Available: https://www.x-plane.com/.

[4] Schäfer, M., Strohmeier, M., Lenders, V., Martinovic, I., & Wilhelm, M. (2014, April). Bringing up OpenSky: A large-scale ADS-B sensor

network for research. In IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (pp. 83-94). IEEE.

[5] Ali, B. S., Ochieng, W., Majumdar, A., Schuster, W., & Chiew, T. K. (2014). ADS-B system failure modes and models. The Journal of Navigation, 67(6), 995-1017.

[6] Fremont, D. J., Dreossi, T., Ghosh, S., Yue, X., Sangiovanni-Vincentelli, A. L., & Seshia, S. A. (2019, June). Scenic: a language for scenario specification and scene generation. In Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation (pp. 63-78).

[7] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. (2017, October). CARLA: An open urban driving simulator. In Conference on robot learning (pp. 1-16). PMLR.

[8] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28, 91-99.

[9] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).

[10] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., ... & Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3213-3223).

[11] Geiger, A., Lenz, P., & Urtasun, R. (2012, June). Are we ready for autonomous driving? the kitti vision benchmark suite. In 2012 IEEE conference on computer vision and pattern recognition (pp. 3354-3361). IEEE.

[12] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.

[13] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

[14] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[15] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

[16] Tagebrand, E., & Gustafsson Ek, E. (2021). Dataset Generation in a Simulated Environment Using Real Flight Data for Reliable Runway Detection Capabilities. Master's Thesis.

[17] Haddad Josef (2021). Data Synthesis in Deep Learning for Object Detection. Master's Thesis, unpublished work.

[18] Wang, P., Sun, H., Bai, X., Guo, S., & Jin, D. (2021). Traffic thermal infrared texture generation based on siamese semantic CycleGAN. Infrared Physics & Technology, 116, 103748.

[19] Palladino, J. A., Slezak, D. F., & Ferrante, E. (2020, November). Unsupervised domain adaptation via CycleGAN for white matter hyperintensity segmentation in multicenter MR images. In 16th International Symposium on Medical Information Processing and Analysis (Vol. 11583, p. 1158302). International Society for Optics and Photonics.

[20] Choi, Y., Choi, M., Kim, M., Ha, J. W., Kim, S., & Choo, J. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 8789-8797).

[21] Cluzeau, J.M. et al., "Concepts of design assurance for neural networks (CoDANN)," Public Report Extract, EASA AI Task Force and Daedalean AG, Version 1.0, March 31, 2020.

[22] Giovanni, C et al., "Concepts of design assurance for neural networks (CoDANN) II," Public Report Extract, EASA AI Task Force and Daedalean AG, Version 1.0, May 19, 2021

[23] Richter, S. R., AlHaija, H. A., & Koltun, V. (2021). Enhancing Photorealism Enhancement. arXiv preprint arXiv:2105.04619.