Lars Grunske, Janet Siegmund, Andreas Vogelsang (Hrsg.): Software Engineering 2022,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2022 11

Build Your Own Training Data – Synthetic Data for Object Detection in Aerial Images

Lea Laux [1], Sebastian Schirmer [1], Simon Schopferer [1], Johann C. Dauer [1]

Abstract: Machine learning has become one of the most widely used techniques in artificial intelligence, especially for image processing. One of the biggest challenges in developing an accurate image processing model is to collect large amounts of data that are sufficiently close to the real-world scenario. Ideally, real-world data is therefore used for model training. Unfortunately, real-world data is often insufficiently available and expensive to generate. Therefore, models are trained using synthetic data. However, there is no standardized method of how training data is generated and which properties determine the data quality. In this paper, we present first steps towards the generation of large amounts of data for human detection based on aerial images. To create labeled aerial images, we are using Unreal Engine and AirSim. We report on first impressions of the generated labeled aerial images and identify future challenges – current simulation tools can be used to create realistic and diverse images including labeling, but native support would be beneficial to ease their usage.

Keywords: Machine Learning; Synthetic Data; Simulation Environment; Unmanned Aircraft; Human Detection

1 Introduction

Future unmanned aircraft system that operate beyond visual line of sight will need a trustworthy perception of the environment to carry out their missions without endangering others. State-of-the-art perception algorithms are based on machine learning (ML) techniques, where the perception function is learned from data. This learning process is called training and real-world data as well as synthetic data can be used for it. Training is an essential but unfortunately error-prone task. Therefore, after training, the function needs to be validated to understand whether it generalizes and therefore can be used for yet unseen real-world scenarios. In fact, European Union Aviation Safety Agency (EASA) published its Artificial Intelligence Roadmap and Concepts of Design Assurance for Neural Networks that entails a W-shaped development cycle for learning assurance where data management is one of the first steps [EA]. The objective of the data management is to evaluate the completeness, correctness, and quality of the data for the respective task. It is clear that real-world data achieves the highest quality. But, since real-world data is difficult to generate, especially in case of unmanned flights, it has its downside in terms of completeness. Synthetic data in contrast to real-world data can be generated automatically and data properties like photorealism, fidelity, and variability depend on the current state of the tools and technologies used where improvements are foreseeable.

[1] German Aerospace Center (DLR), Braunschweig, Germany, <firstname>.<lastname>@dlr.de

The use case considered in this paper is humanitarian aid with an unmanned aircraft. Figure 1 shows the research helicopter superARTIS with a drop system for humanitarian needs. The drop system shall only be activated if there is no human close to the release area. To detect humans within the release area, we use a trained YOLOv4 model on board. For training, we are using real-world as well as synthetic data.

Fig. 1: Unmanned DLR superARTIS research helicopter with drop system for humanitarian needs.

In this paper, we present the current state of our ongoing research regarding synthetic data generation for human detection on aerial images. To create synthetic data, we report our experience using Unreal Engine and AirSim. Further, we use segmentation images to automatically create labels for the supervised training. Our current setup requires a user to manually set up an environment and define parameters such as the desired number of images, then aerial images showing labeled humans on the ground can be generated automatically.

In the following, we state requirements on our simulation environment, then present the developed toolchain, and discuss our experience with the generation of a first set of image data for the use case described above. Furthermore, we disclose current challenges of the image generation approach and discuss both conceptual and technical issues.

2 Related Work

The mix between synthetic training data and real-world data has shown promising results for person detection [Yu10]. Furthermore, synthetic training data was successfully used to train neural networks for unmanned aerial vehicle applications [KBK19] – aerial images were used for fire detection and house counting. Yet, the authors report that it is unclear whether this success generalizes to different use cases. In this work, we investigate whether synthetic images can be used to train neural networks to detect humans in aerial images.

To generate synthetic training data, simulation environments can be used where not only fidelity is desirable but also the possibility to create diverse scenarios [Ma18; Yu10] in an automated manner. In fact, a large variety of simulation environments exist that range from simulators for controller development like Gazebo[2] to photorealistic flight simulators like X-Plane[3] for pilot training. Another simulation environment that can be found in between is AirSim [Sh17]. AirSim is a simulation environment for autonomous vehicles using the game engine Unreal Engine[4]. The Unreal Engine marketplace can be used to access a

[2] http://gazebosim.org/

[3] https://www.x-plane.com/

[4] https://www.unrealengine.com/

variety of different environments and human models. Further, AirSim has already been used for deep learning based classification of pedestrians [Sc19]. Yet, the purpose of none of the mentioned tools is the creation of synthetic training data. Therefore, in this paper, we examine how well AirSim can be used to generate a diverse set of aerial images.

Other tools bridge the gap between simulation, machine learning training, and verification. One example is VERIFAI that can be used for the design and analysis of machine learning components [Dr19]. The initial version of VERIFAI builds on top of existing simulators and uses simulation runs to verify deep neural networks for perception tasks. It incorporates SCENIC [Fr] – a modeling language for automatic scene generation. Currently, to our knowledge, these tools do not support in-air scenarios that would be an useful future extension. Most similar is the work of [ED20] that uses Unity 3D[5] and a formal description language to generate desirable aerial scenarios.

With regards to certification, EASA points to risks and mitigation of using synthetic data. They state that synthetic data should never be used without proper analysis and mitigation of the domain biases, no matter how realistic it looks. Also, testing using synthesized data is only supplementing testing using actual data and is not replacing it – otherwise derived learning assurances would not apply. However, they also acknowledge the benefits of synthetic data as it helps to find edge cases that almost never happen in the real world or that would be difficult or very costly to reproduce [EA]. This work represents first steps towards data generation for a specific aerial operation with a defined concept of operation (ConOps) specifying operational scenarios and an operation design domain (ODD) representing operation conditions and limits [EA21]. We present the toolchain that generates images, efficient ways to achieve ODD coverage requirements are part of future work.

3 Requirements for the Simulation Environment

The use of a simulation environment is a prominent and promising way for generating training data that satisfies the requirements for quality and scalability. To meet the desirable requirements of high quality data, it is necessary to define the term of good synthetic training data. In fact, there are different opinions on how to balance fidelity, i.e., photorealism, on the one hand and data diversity on the other hand. According to the findings of Mayer et al, photorealism is often overrated [Ma18]. Whereas others state the lack of fidelity of simulation environments as one of the major challenges [Af20]. In this work, we decided to prioritize diversity of the generated training data while also aiming for a high level of photorealism. The HERIDAL data set, which consists of a series of real-world aerial images, serves as a guide for ideal training data [BMG19]. The data set includes aerial images of humans in different environments like mountains, parks, forests, and other typical middle European landscapes. One example image of HERIDAL is shown in Figure 2.

[5] https://unity.com/

Considering the necessary quality of the training data and the use case for detecting humans in aerial images, we identified the following requirements:

Fig. 2: Real-world aerial image from the HERIDAL data set [BMG19], CC BY 3.0 Unported License.

Realistic and Plausible Drone Flight The simulation environment is able to fit the requirements of a realistic and plausible drone flight. Hence, images are taken at a flight level of approximately 50 meters. Further, the camera on board the drone should capture different scenarios in different camera positions and angles during flight.

Automation Capability The toolchain should be automatable. This includes the setup, choice of environment, placement of objects like humans, configuration of parameters described in the previous requirement, labeling of detected humans, and storage of data in a format suitable for further processing. Automation is important since it promises cost savings and a scalable data generation process.

Time Requirements The simulation environment is able to create data in a reasonable time. A large amount of data is usually required for the task of machine learning. Therefore, our definition of a reasonable time to meet the requirement of scalability suitable for machine learning starts with at least 1000 images per hour.

Open Source The relevant tools of the simulation environment are available under an open source license and are developed by an active community. It is desirable to have an exchange of knowledge and support by the community. Particularly with regard to the sustainability of the simulation environment and its usage in the future, a highly maintained software is preferable.

4 Synthetic Data Generation Toolchain

The toolchain is based on Unreal Engine and AirSim. Unreal Engine is used to create virtual worlds and place assets such as humans in them. Then, AirSim is used to interact with this virtual world. Technically speaking, AirSim is a plugin for Unreal Engine. Typically, it is used to simulate drone flights and car driving. AirSim can be also used to create images during simulation runs. Before these generated images can be used for ML training, we process the images to augment them with labels. In the following, we present the components of the toolcain in more detail.

4.1 Unreal Engine

Unreal Engine and its marketplace have a variety of available projects that include large virtual worlds. In principle, all the worlds and environments of the Unreal Engine and the given marketplace can be used for the purpose of gathering data. We chose environments based on photorealism and free availability. In future, it would also be possible to design own or buy other worlds. Currently, the worlds we are using are: *Blocks*, *City Park Environment Collection*, *Megascans Goddess Temple*, and *A Boy and His Kite*. *Blocks* directly comes with AirSim and consists of one plain layer with cubes, cones, and balls. The *City Park Environment Collection* is a large park that includes playgrounds, green areas, lakes, cafes, roads, and many more. This world is mainly used in this work. *Megascans Goddess Temple* represents a temple in the mountains with large rocks and very bold cliffs. Last, *A Boy and His Kite* incorporates large green areas, mountains, lakes, river, and caves.

In the Unreal Engine marketplace, there are also asset packages with different objects that can be placed in the environment. In this project, we use two different asset packages for humans: Scanned 3D People Pack[6] and Twinmotion Posed Humans[7]. These two packages alone already include a total of 161 different humans. The number of humans in the world can be configured at the beginning of the data generation. Currently, we spawn humans at random positions. The Unreal Engine also provides a list of all available objects in the current environment like trees. This list of objects can be used to guide the distribution and placement of humans. Either all objects or a filtered subset of objects are used for this purpose. To filter the list of objects, a blacklist that entails objects where a humans should not be placed and a whitelist that contains objects where a human should be placed at random is used. The blacklist and the whitelist are maintained by the user.

4.2 AirSim

AirSim is a plugin for Unreal Engine and allows to access Unreal Engine data. By accessing virtual cameras, AirSim is able to generate standard images and segmentation images. An example is shown in Figure 3. It is possible to use different modes for the car, the unmanned aerial vehicle, or the environment itself in a computer vision mode. The relevant mode for our use case is the computer vision mode, simulating a drone flight in bird's eye perspective without the overhead of simulating the drone itself.

The position of the humans can be extracted out of the Unreal Engine. Based on this position, the correct coordinates for generating the images can be derivated and used by AirSim. To increase the randomness and diversity of the images, a random altitude between 30 meters and 60 meters and a random perimeter between 10 meters and 50 meters are chosen. Concerning the virtual camera, the camera angle and twist randomly vary between

[6] https://unrealengine.com/marketplace/en-US/product/9c3fab270dfe468a9a920da0c10fa2ad
[7] https://unrealengine.com/marketplace/en-US/product/twinmotion-posed-humans

Fig. 3: Aerial images by AirSim: camera image on the left and its segmentated version on the right.

-15° and 15° and between 0° and 360°, respectively. Further, a random change in date and time ensures different lightning conditions. Also, it is possible to customize weather and environment conditions like rain, snow, leafes, fog, and dust. The probabilistic nature ensures to generate a diverse set of training data that also includes corner-cases.

4.3 Segmentation

AirSim generates images of the environment and so called segmentation images for ground truth labeling. The segmentation is configured by AirSim itself, so it is possible to assign different colors to different objects based on their ID in the Unreal Engine. In our setting, humans are colored white and other objects black as can be seen on the right side of Figure 3.

For the chosen machine learning framework YOLOv4, a specific format for the training data is used. The training data needs to specify a bounding box that captures the borders of the desired object. Figure 4 depicts such a bounding box.

Fig. 4: Generated image of a human by AirSim labeld by a bounding box.

The bounding box is calculated based on the segmentation image taken by AirSim. Here, white and black pixels stand for pixels of humans and other objects, respectively. Note that visible objects are segmented, meaning humans hidden behind other objects are only partially visible. Next, human pixels are clustered such that one cluster represents either a single human or a group of humans. If a cluster covers too small an area, it will be discarded since the human is not visible enough.

5 Discussion of the Results and Current Challenges

In this section, we discuss our experience with AirSim and Unreal Engine. Further, we give first results whether the generated images can be used for neural network training. The experiments focus on understanding the quality of the generated data – they do not represent an evaluation of the use case or the detection algorithm. The toolchain has been used in our experiments to create synthetic images with the described Unreal Engine worlds and AirSim for drone simulations and humans on the ground. They are processed to a format readable for the first tests with a machine learning framework. The results indicate that further improvements of the toolchain are required. Generated images are of high quality but often show humans hidden in shadows or concealed by obstacles. These data represent good edge cases, but are too difficult for the initial training.

5.1 AirSim/Unreal Engine Toolchain

An advantage of the presented toolchain is the possibility to partially automate the process itself. After choosing, deploying, and configuring a virtual world once, the toolchain generates one to two images per second. Creating and configuring an environment of Unreal Engine takes around 15-30 minutes, human generation in the world takes one to three minutes, creating the images itself with AirSim takes two to five minutes for all images, and post processing takes two to ten minutes. Our hardware setup includes an Intel i7-6700 CPU @3.40GHz (8.4 CPUs) with 16 GB of main memory and an GPU NVIDIA Quadro K620. Figure 5 shows some of the generated images using the City Park Environment Collection. Currently, due to the random placement of the humans, a manual review of the generated images is recommended to assure that humans are visible from the aerial camera perspective, i.e., they are not hidden in shadows.

Fig. 5: Sample images generated by the proposed toolchain.

5.2 Experimental Results

This section gives preliminary and inconclusive results for using the generated data for ML training. We have created 2780 synthetic images in different environments for human detection in aerial images in roughly half a day using the presented toolchain. The duration also includes some random manual checks of the generated images. The manual checks suggested that the captured scenes are often too difficult for training since the placed humans are often hidden in shadows or concealed by other obstacles. Hence, we designed an experimental test to validate this presumption. In the test, the results of a neural network trained exclusively on real-world data were compared with the same network further trained with the generated images – if the synthetic images are close to the real-world data, we expect only small deviations in the network performance. As the neural network, we have used a YOLOv4-tiny neural network trained with default parameters[8] and the HERIDAL data set [BMG19]. The validation was based on 54 synthetic and 27 real-world images. Further, we used the Intersection over Union (IoU) [Ev15] as metric to compute the similarity of the predicted bounding boxes. We differentiate between three classes of results: fully correct, partially correct, and incorrect. A result is correct if the calculated bounding box achieves an IoU with the ground truth greater than 50%. Otherwise, the calculated bounding box is considered as partially correct. In case the IoU is zero, the result is considered incorrect. The results showed that our presumption was correct. Many images were too difficult and did not help training, but even worsened the network performance – the network without training on synthetic data performed better. It achieves 30% fully correct, 37% partially correct, and 33% incorrect. The network further trained with synthetic data achieves only 24% fully correct, 4% partially correct, and 72% incorrect. When comparing the synthetic and the real-world data, humans are more visible in the real-world data due to better positioning, lightning conditions, and better differentiation between their surroundings. In fact, it is an open question to define the ratio between simple and difficult synthetic data. For training purposes, simpler data may be beneficial, while for testing difficult data that considers edge cases may be more beneficial. Next, we formalize further challenges that address these problems.

Challenge of Object Visibility
The approach of using simulations and 3D render engines to generate synthetic training data holds the promise of delivering an arbitrary amount of high quality training data sets. However, the setup described in this paper, uncovers some key challenges with this approach. With random positions of the camera and humans in the scene, it regularly occurred that humans were partially occluded by trees or other objects as seen from the camera's viewpoint. Similary, humans standing in the shadows of trees or other objects were sometimes barely or not at all visible in the rendered image.

[8] `https://github.com/AlexeyAB/darknet/blob/master/cfg/yolov4-tiny.cfg`

This raises the questions to which degree a detection algorithm should be expected to detect a partially visible human and how to quantify this degree of visibility in the training data. As detecting humans with 0% visibility is infeasible, a requirement should be stated to limit the degree of object visibility the algorithm must reliably cope with. This limit could be specified with corresponding scenarios in the ConOps and operational constrains in the ODD. For example, potential release areas for box dropping could be restricted to open fields without large objects such as trees which would obstruct the view or cast shadows.

However, even if a limit regarding the degree of visibility is specified in the ODD, there is no straight-forward way to assess the degree of occlusion and visibility when generating labels based on the segmentation images. To solve this technical issue, an approach was taken to generate control images alongside the rendered images with the humans removed from the scene. A comparison of both images would – in theory – allow the decision of whether or not the human should be classified as visible and hence its label should be included in the training data or not. However, with the AirSim render engine used in this work, it proved to be difficult to generate control images that differed from the original image only in the existence of certain objects in the scene. Seemingly random differences in rendering artifacts, such as shadows and moving leafs, made it impossible to reliably calculate a difference metric for the control images.

6 Conclusion

For machine learning, data from the real-world as well as synthetically generated data can be used. Synthetic data have the advantage that no cost intensive real-world experiments are required, for which corner-cases can also be safety-critical. Further, the generation of data based on simulations offers the flexibility needed to create a variety of scenarios that help improve the completeness of the data set. In this paper, we presented on-going work on a toolchain to create synthetic data for detecting humans in aerial images. The Unreal Engine plugin AirSim is used as simulation environment: AirSim offers the flexibility to capture different aerial scenarios whereas the Unreal Engine provides a range of worlds and objects. The generation of synthetic data was mostly automated – only the initial scenario must be defined manually. The current toolchain can efficiently generate a large amount of synthetic data, but often the captured scenes are too difficult for the initial learning process, e. g., humans are often hidden in shadows, under water, or concealed by obstacles. This work showed that the use of simulations is promising, but does not simply mean placing humans in a virtual world. In the future, we plan to no longer place humans randomly but rather to use operational limits, operational conditions, and scenarios defined in the ConOps to guide the process. This not only helps to control the level of difficulty, but also to ensure that the data generation achieves a certain coverage of the requirements. Here, one challenge is that simulation tools are not designed to support machine learning in the first place. Further, after updating the human placement algorithm, we plan to conduct more experiments with more synthetic data to show more conclusive results.

Literatur

[Af20] Afzal, A.; Katz, D. S.; Goues, C. L.; Timperley, C. S.: A Study on the Challenges of Using Robotics Simulators for Testing./, 15. Apr. 2020, arXiv: 2004.07368, URL: http://arxiv.org/abs/2004.07368, Stand: 17. 03. 2021.

[BMG19] Božić-Štulić, D.; Marušić, Ž.; Gotovac, S.: Deep Learning Approach in Aerial Imagery for Supporting Land Search and Rescue Missions. International Journal of Computer Vision 127/9, S. 1256–1278, 1. Sep. 2019.

[Dr19] Dreossi, T.; Fremont, D. J.; Ghosh, S.; Kim, E.; Ravanbakhsh, H.; Vazquez-Chanlatte, M.; Seshia, S. A.: VerifAI: A Toolkit for the Formal Design and Analysis of Artificial Intelligence-Based Systems. In: Computer Aided Verification. Cham, 2019, ISBN: 978-3-030-25539-8 978-3-030-25540-4.

[EA] EASA; AG, D.: Concepts of Design Assurance for Neural Networks I and II./, URL: https://www.easa.europa.eu/, Stand: 17. 01. 2022.

[EA21] EASA: First usable guidance for Level 1 machine learning applications./, Dez. 2021, URL: https://www.easa.europa.eu/, Stand: 17. 01. 2022.

[ED20] Ellis, O. S.; Durak, U.: Simulation Based Verification of Drogue Detection Algorithms for Autonomous Aerial Refueling. In: AIAA SciTech Forum. Jan. 2020, URL: https://elib.dlr.de/138636/.

[Ev15] Everingham, M.; Eslami, S. M. A.; Gool, L. V.; Williams, C. K. I.; Winn, J. M.; Zisserman, A.: The Pascal Visual Object Classes Challenge: A Retrospective. Int. J. Comput. Vis. 111/1, S. 98–136, 2015.

[Fr] Fremont, D. J.; Yue, X.; Dreossi, T.; Ghosh, S.; Sangiovanni-Vincentelli, A.; Seshia, S. A.: Scenic: Language-Based Scene Generation./, S. 28.

[KBK19] Kamilaris, A.; Brink, C. v. d.; Karatsiolis, S.: Training Deep Learning Models via Synthetic Data: Application in Unmanned Aerial Vehicles. arXiv:1908.06472 [cs, eess]/, 18. Aug. 2019, arXiv: 1908.06472.

[Ma18] Mayer, N.; Ilg, E.; Fischer, P.; Hazirbas, C.; Cremers, D.; Dosovitskiy, A.; Brox, T.: What Makes Good Synthetic Training Data for Learning Disparity and Optical Flow Estimation? International Journal of Computer Vision/, 2018.

[Sc19] Schleusner, J.; Neu, L.; Behmann, N.; Blume, H.: Deep Learning Based Classification of Pedestrian Vulnerability Trained on Synthetic Datasets. In: 2019 IEEE 9th International Conference on Consumer Electronics (ICCE-Berlin). Berlin, Germany, 8. Sep. 2019.

[Sh17] Shah, S.; Dey, D.; Lovett, C.; Kapoor, A.: AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles./, 18. Juli 2017, arXiv: 1705.05065, URL: http://arxiv.org/abs/1705.05065, Stand: 17. 03. 2021.

[Yu10] Yu, J.; Farin, D.; Kruger, C.; Schiele, B.: Improving person detection using synthetic training data. In: 2010 IEEE International Conference on Image Processing. Hong Kong, Hong Kong, Sep. 2010.