GARD - Gustavo's Awesome Runway Dataset - A Synthetic Runway Image Dataset built with Diffusion Models

Gustavo de Paula

Contents

Introduction

The template I chose for this project is “Gather your own dataset”.

This project intends to use state-of-the-art image generation models such as Latent Diffusion Models popularized by Stable Diffusion to generate a synthetic aircraft runway image dataset to be used in vision-based landing tasks. This approach can greatly increase the availability of runway images to be used in other research projects, and, by sharing the models weights freely, allow

other researchers to generate their own synthetic dataset with their own desired characteristics.

Motivation

Currently, there is an increasing interest in autonomous systems in the aerospace field. Commercial airplanes already have autopilot when the aircraft is in cruise mode, but takeoff and landing are still heavily in the pilots' hands. Machine learning models are a promising approach to help in vision-based landing, specially in the runway detection component.

Contemporary machine learning techniques usually require 10s or 100s of thousands of items in a dataset. But there are few open-source datasets of aerial images that can be used in the task of runway detection. Synthetic image dataset generation can be a viable strategy to bridge this gap of data availability.

Literature Review

In this section, prior work on building a runway dataset is reviewed and critiqued. Four well-known, publicly available runway images datasets are reviewed. Additionally, papers relevant to the methodology are reviewed as well.

LARD – Landing Approach Runway Detection [1] is a dataset built using images from Google Earth. It uses geometry to to simulate the viewpoint of a camera in the airplane's nose seeing the runway. Along with the public dataset, code to generate images using Google Earth studio is also provided. Currently, the dataset has over 17,000 images.

FS2020 Runway Dataset [2] is a public runway dataset available on Kaggle built using extracted video footage from Microsoft Flight Simulator. The researches recorded videos of airplanes landing on 60 different runways across the world in different light conditions. Then, the videos are split into several frames, and each one annotated with segmentations labels. The dataset contains 5,587 images.

BARS: A Benchmark for Airport Runway Segmentation [3] is a public dataset built using images from the X-Plane, a FAA-certified flight simulator. Their dataset contains 10,256 images, collected from different airports and weather conditions, to simulate real-world scenarios.

Runway Landing Dataset (RLD) [4] is also a dataset built using X-Plane. The authors built RLD to use in VALNet, a runway image segmentation network. In the paper, the researchers had access to BARS and other non-publicly available datasets. They claim to have built RLD to *"alleviate the deficiency of large-scale, high-quality, publicly available datasets for runway segmentation"*. The dataset includes images covering entire landing scenarios and different weather conditions. The dataset has 12,239 images.

LARD, the biggest dataset, is also the most recent one. The images generated by it are easily reproducible, as all the code to generate new scenarios and new images is publicly available. In the synthetic images of LARD, there are no variations in the weather conditions and night view is simulated with simple reduction of lightness in the photos, both of which create unrealistic images.

The datasets built using flight simulators (FS2020, BARS, RLD) have far better data diversity as they are able to produce realistic images in different weather and light conditions. But, as they use flight simulator games, they are not easily reproducible and extendable.

These trade-offs indicate the potential of the novel approach of this paper. A runway image generating model has the potential to be easily reproducible and extendable as well as being capable of generating images for different realistic scenarios. It could also help in generating edge-cases that are hard to replicate in flight simulators and also generate images of novel and unseen airports and runways.

The paper **Runway Detection and Localization in Aerial Images Using Deep Learning** [5] trains a two-stage pipeline: first, there is a model that detects if there's a runway in the image and then there is another model that does localization. Their approach to runway detection is using a Resnet50 model, fine-tuned with a runway image dataset. They are able to achieve an accuracy of around 90% in detection, although the dataset used in evaluation uses images with a top-down or satellite-like camera perspective, which is not realistic in the vision-based landing scenario explored in this paper.

A hard problem in building a dataset in general is evaluating the quality of this dataset. Ideally, the best way would be to train a runway detection and segmentation model on this new dataset and evaluating the performance of this dataset. Because of time constraints, this won't be covered in this paper. But training a simple model to detect if there's a runway in the image can be a helpful proxy to the quality of the dataset.

In recent years, diffusion models have become the state-of-the-art solution in the field of generative image modelling. Diffusion models work by progressively destructing data adding noise and then training a model that learns how to progressively remove that noise, reconstructing the original data.

In the paper ***"Diffusion Models: A Comprehensive Survey of Methods and Applications"*** [6], the authors provide an overview of the foundations of diffusion models and several related techniques such as *inpainting*, *DreamBooth*, and text-to-image generation. They also review several applications for diffusion models, such as *"Generating Data from Diffusion Models"*, which is the application explored in this project. The authors talk positively about diffusion models for computer vision tasks, citing that synthesizing datasets using them can effectively advance tasks such as classification and data augmentation.

Design

Methodology

The background research has shown the importance of datasets for researchers working on the field of detecting runways for the task of automated aircraft vision-based landing. Real-image datasets are too small to be used in deep-learning model training. Synthetic datasets bridge the gap of data availability and use simulated images from programs such as X-Plane, Microsoft Flight Simulator and Google Earth to generate the runway images.

This project's primary research question is how can a suitable synthetic image dataset be built for computational vision tasks without the need of images extracted from simulators or similar. To answer this question, the project uses the field of vision-based landing and builds a synthetic runway image dataset.

The primary users of this project are researchers that use the dataset provided in this project to build models for runway detection. Secondary users might be researchers interested in synthesizing their own datasets for computational vision tasks such as classification or segmentation.

The proposed research question begs the question of what constitutes a "suitable synthetic image dataset". Here, we rank the following characteristics that make an image dataset suitable:

1. **Human-judged realism**: humans subjectively judge the images as credible and realistic. When comparing side-by-side the images of the generated dataset with other datasets, the perceived quality of the images are the same or better.
2. **Detection by existing models**: fine-tuned models on available datasets can detect the presence and segmentation of the desired data (runways in this project) in the images.
3. **Data diversity**: the dataset has good data diversity, including edge cases and several data variations.
4. **Model training**: an existing architecture can be trained on the dataset and achieve a reasonable performance or a pre-trained model can be fine-tuned and have their performance improved.

Hypothesis and experiments

The emergence of diffusion models for image generation creates a golden opportunity to apply this state-of-the-art technique to the problem at hand. Thus, the project will evaluate the following hypothesis: *"Can diffusion models generate a suitable runway image dataset?"*. This novel idea would be ground-breaking for the runway detection and segmentation field, because it would mean that researchers would be able to use existing techniques in the field of image generation such as *inpainting*, *DreamBooth*, and *LoRAs* to generate their own extensive datasets with images in diverse, specific and complex scenarios.

Suitable image generation To evaluate the hypothesis, a Python project using the *Diffusers* library will be written. The first part of the project will evaluate what main techniques can be used to generate suitable images: Unconditional image generation, text-to-image generation and image-to-image generation.

Secondary to the issue of image generation is the question if diffusion models are able to generate diverse, specific and complex scenarios, such as scenes that vary in relation to runway orientation, weather conditions (e.g., fog, rain, snow), lighting (e.g., dusk, dawn, night, heavy overcast), and background scenery (e.g., airport buildings, natural terrain, distant cityscapes),

Unconditional image generation Unconditional image generation is a good starting point because it allows generation of images that look like those in the training dataset in a simple manner. In the context of generating runway images, we can train models using two approaches:

- Train with images of runways in their background scenery, with, for example, the airport buildings and the terrain;
- Train with cropped images that contain only or primarily the runway;

The hypothesis to be explored here is if it's possible to generate suitable images when training with the original images, because there is a possibility that the model doesn't learn the main concept of the runway and just learns to generate scenery that is found around the runway.

Training with cropped images has a better chance for the model to learn the concept and features of a runway, but it is more limited in the usefulness, as it would require other techniques, such as *Outpainting*, that extends the original image beyond its original boundaries, to generate a complete and realistic image for the dataset.

Text-to-image generation Text-to-image generates an image from a text description, also known as *prompt*. Text-to-image generation is the most powerful and adaptable technique because it would allow the extension of the dataset by generating new custom images using text prompts.

The simplest way to explore the question is to start with *prompt engineering*. Prompt engineering is the technique to tweak and try different text prompts to guide the image generation process.

It is unlikely that prompt engineering alone will be able to generate suitable images for the dataset. In that case, other techniques will need to be tested:

1. **Prompt weighting**: prompt weighting is a technique that emphasize or de-emphasize certain parts of the text prompts by manipulating their embedding values. This allows for more control over the generated image as the model will focus more on certain concepts.
2. **Textual inversion**: fine-tunes the text embedding of the model to learn a new concept.

3. **DreamBooth**: fine-tunes the whole model to learn how to generate contextualized image of a given subject.

Image-to-image generation Image-to-image generation is similar to text-to-image, but in addition to passing a text prompt, an image is also passed as the starting point for the diffusion process. The two techniques selected here to be explored are:

1. **ControlNet**: provides structural guidance to shape the generation of an image. It can be a pose skeleton or segmentation masks, for example, that will have their information retained and use to guide de de-noising process. ControlNet can be used to generate images in a way where the specific position of the runway is determined beforehand.
2. **Inpainting**: Inpainting modifies an existing image. It takes a mask that shows where the model should change the image. The model then tries to generate the content in the masked out area consistent with the surrounding pixels and the given prompt. It can be used to transform a scenery image into a runway image.

And because image-to-image also uses text-to-image, there's the possibility to use techniques such as Prompt weighting, textual inversion and DreamBooth alongside ControlNet and Inpainting for better performance.

Another technique that could be further developed in another project would be to use image-to-image generation to extend existing datasets. For example, the LARD datasets doesn't have different weather conditions for the images. Thus, we could use image-to-image generation to add different weather conditions (e.g., fog, rain, snow).

Evaluation

The evaluation of the synthetic runway image dataset will be both qualitative and quantitative, focusing on the key characteristics defined in the report: realism, detectability by existing models, data diversity, and potential to improve model performance.

Human-Judged Realism:

1. **Approach**: Conduct a small-scale human evaluation. Recruit a few peers to rate the realism of a random sample of generated images compared to a sample of images from established datasets (e.g., LARD or BARS).
2. **Metrics:** Rate images on a Likert scale (1–5) for realism.
3. **Feasibility:** Small-scale user study with at least 5 evaluators, each rating at least 20 images.

Detection by Existing Models:

1. **Approach**: Take a pre-trained runway detection model (e.g., a model pre-trained on LARD or a simple ResNet50 classifier) and test it on a

subset of the synthetic dataset.
2. **Metrics**: Accuracy, Precision, Recall, and F1-score for runway detection on the synthetic images.
3. **Feasibility**: This could be done by fine-tuning a lightweight classifier if no pre-trained model is readily available.

Data Diversity

1. **Approach:** Qualitatively check scenarios: generate images under different conditions (weather, lighting, background) and visually inspect variety.
2. **Metrics**: Define a small taxonomy of conditions to be covered such as weather (Clear, Fog, Rain, Snow), lighting (Day, Dusk/Dawn, Night), and background (Urban, Rural, Coastal, Mountainous) and report the number of images in each category.

If time permits, there could be a quantitative metric:

1. **Approach:** Use embedding-based diversity metrics (e.g., CLIP embeddings) to measure intra-dataset variability.
2. **Metrics**: Calculate mean pairwise similarity between image embeddings; lower similarity implies greater diversity.

Model training:

1. **Approach**: (If time permits) Train or fine-tune a basic runway detection or segmentation model.
2. **Metrics**: Comparison in accuracy and segmentation metrics.
3. **Feasibility**: Due to time constraints, this may be done at a small scale.
4. **Critique**: There is the possibility that accuracy metrics decrease when training with the new dataset. If the images in the dataset closely mimic reality, it could be the case that performance decreases because existing models are not suitable for real-world challenges.

Project timeline

Week	Start Date	Tasks	Due Dates
1	13/10	Choose project idea	
2	20/10	Pitch rough project idea	
3	27/10	Work on project proposal;Study research ethics	
4	3/11	Work on project proposal;Study previous work	
5	10/11	Start Literature Review	Project Proposal - 11/11
6	17/11	Finish Literature Review	
7	24/11	Start working in project design	
8	1/12	Work on evaluation plan;Finish project desing	

Week	Start Date	Tasks	Due Dates
9	8/12	Prototype on Unconditional Image Generation;Start working on prototype;Record video of prototype	
10	15/12	Start working on text-to-image generation;Prompt engineering and weighting;	Preliminary Report - 16/12
11	22/12	Recess	
12	29/12	Recess	
13	5/1	Work on text-to-image generation;Textual Inversion;	
14	12/1	Work on text-to-image generation;DreamBooth;	
15	19/1	Start working on evaluation;Simple Runway classifier model	
16	26/1	Finish classifier model;Start working on image-to-image model;ControlNet	Draft Report - 27/01
17	2/2	Work on image-to-image model;Inpainting	
18	9/2	Start generating final dataset	
19	16/2	Generate final dataset	
20	23/2	Evaluate final dataset	
21	2/3	Recess	
22	9/3	Finish project report	Exam - 10/3
23	16/3	Review final report	
24	23/3		Final Report submission - 24/3

Prototype

To demonstrate the feasibility of the project, a small-scale dataset is produced using Unconditional Image Generation. Usually, in image generation models, there are inputs that guide the image generation process such as a prompt or initial starting image. This is not the case in unconditional generation, where the model starts from random noise and has no guidance. The result is that after training, the model should output images that look like the dataset it was trained on.

The training dataset was the FS2020 dataset. It was chosen because it has realistic images in diverse scenarios, that were subjectively judged as more aesthetic pleasing than the other datasets. It is also the easier to download as it is hosted on Kaggle.

How diffusion works

The model for image generation used in this prototype is a Denoising Diffusion Probabilistic Model presented in [7]. Diffusion models work by progressively adding noise to an image (called the forward process) and then training a network that learns how to remove noise from the image (the reverse process).

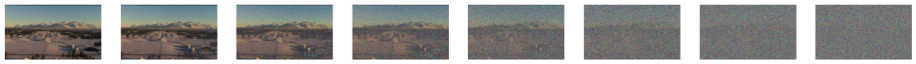

Figure 1: Diffusion forward process

The intuition of Diffusion models is that, if a model can be trained to predict the noise in an image at a timestep, we can start at pure noise, then repeatedly call this model and remove the noise from the image, at each step making a less noisy image.

U-Net

In ***"U-Net: Convolutional Networks for Biomedical Image Segmentation"*** [8], the authors introduced the U-Net architecture for image segmentation. The U-Net is composed of two parts: an encoder and a decoder. The encoder transforms the image into a compressed form that retains essential features. This compressed data is called a "latent". The decoder can then operate on this latent and output some data related to the input data. In the original paper, they used the U-Net to extract biomedical segmentation data. In Diffusion models, the U-Net is used as the model that predicts the noise from an image.

Training

Before training, the dataset was preprocessed by applying four operations: resizing to 128x128 pixels, random horizontal flipping, transforming to Pytorch Tensor and normalizing the tensor values.

Following the same methodology as in the DDPM [7] paper, the `UNet2DModel` from Diffusers was chosen. The model was trained for 1000 epochs, with batch size of 16 and using a AdamW optimizer. A `DDPMScheduler` of 1000 timesteps was used, meaning that in training, there were 1000 possible variations of progressive noise added to the images.

Results

A dataset of 256 images with the resolution of 128x128 was generated. As the goal of the prototype was to evaluate the feasibility of the research project, it is evaluated by two characteristics: subjective human criticism and data diversity.

For data diversity, a small taxonomy is used and the number of images in each category is reported:

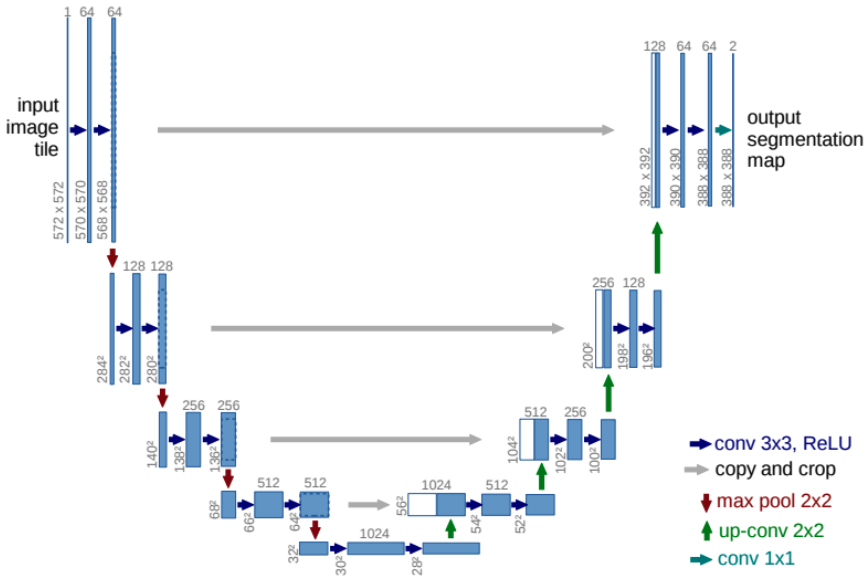

Figure 2: Figure from [8]

Figure 3: Images before and after preprocessing

Class	Subclass	Image count
Weather	Clear	116
	Fog	128
	Rain	0
	Snow	12
Lighting	Day	213
	Dusk/Dawn	25
	Night	18
Background	Urban	13
	Rural	55
	Coastal	30
	Mountainous	7
	Unable to detect	30
	Too close to runway	21
Presence	Runway clearly detectable	90
	Hard to detect runway	11
	No runway in image	155

The images (Figure 4) demonstrate the feasibility of the project: Diffusion models are in fact capable of generating suitable images for a runway dataset. The trained model was able to generate runway images with rich and diverse scenery. The model is even capable of generating images at night, a limitation of the synthetic images in the LARD [1] dataset.

Still, there are challenges to be addressed further in the research project. The images have a low-resolution (128x128), some images do not contain a runway, there's too much fog in a lot of images that make the runways unrecognizable, the demarcations and numbers in the runway should follow a specific pattern observed in real runways and the dataset itself needs to be larger. And most importantly, as it uses unconditional image generation, it's not possible to specify desired characteristics to generate an image.

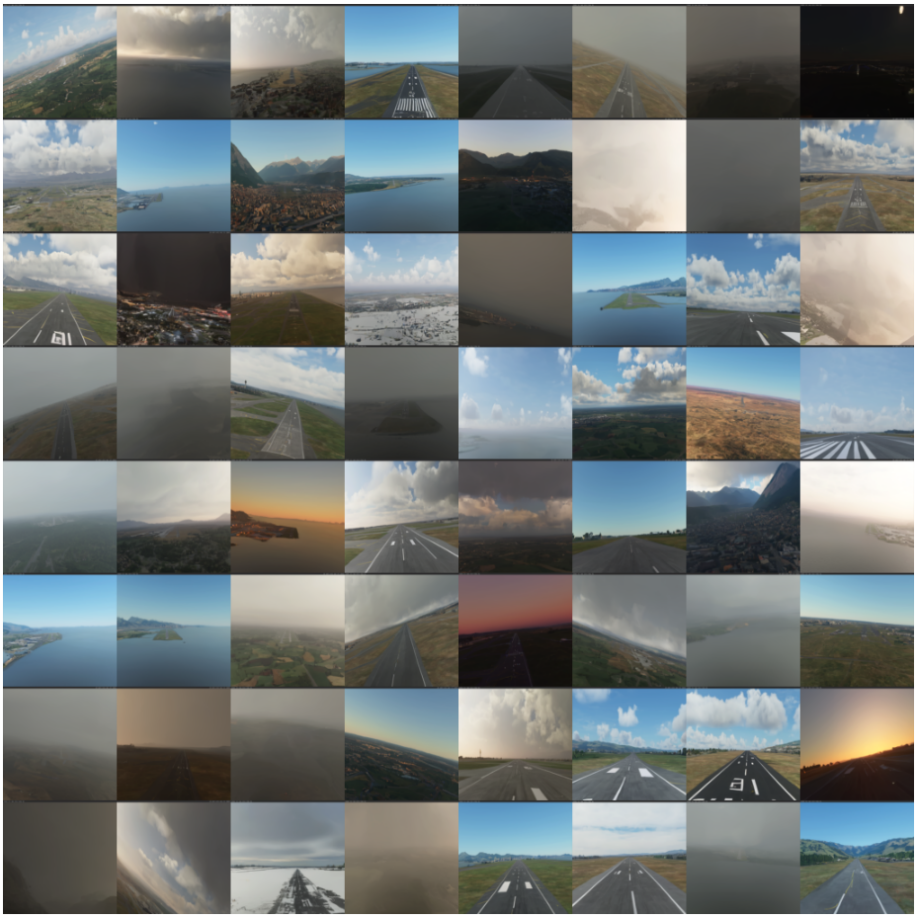

Figure 4: Images 192-256

References

[1] Mélanie Ducoffe, Maxime Carrere, Léo Féliers, Adrien Gauffriau, Vincent Mussot, Claire Pagetti, and Thierry Sammour. 2023. LARD – Landing Approach Runway Detection – Dataset for Vision Based Landing. DOI:https://doi.org/10.48550/arXiv.2304.09938

[2] Mingqiang Chen and Yuzhou Hu. 2024. An image-based runway detection method for fixed-wing aircraft based on deep neural network. *IET Image Processing* 18, 8 (2024), 1939–1949. DOI:https://doi.org/10.1049/ipr2.13087

[3] Wenhui Chen, Zhijiang Zhang, Liang Yu, and Yichun Tai. 2023. BARS: A Benchmark for Airport Runway Segmentation. DOI:https://doi.org/10.48550/arXiv.2210.12922

[4] Qiang Wang, Wenquan Feng, Hongbo Zhao, Binghao Liu, and Shuchang Lyu. 2024. VALNet: Vision-Based Autonomous Landing with Airport Runway Instance Segmentation. *Remote Sensing* 16, 12 (January 2024), 2161. DOI:https://doi.org/10.3390/rs16122161

[5] Javeria Akbar, Muhammad Shahzad, Muhammad Imran Malik, Adnan Ul-Hasan, and Fasial Shafait. 2019. Runway Detection and Localization in Aerial Images using Deep Learning. In *2019 Digital Image Computing: Techniques and Applications (DICTA)*, 1–8. DOI:https://doi.org/10.1109/DICTA47822.2019.8945889

[6] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2024. Diffusion Models: A Comprehensive Survey of Methods and Applications. DOI:https://doi.org/10.48550/arXiv.2209.00796

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 6840–6851. Retrieved December 15, 2024 from https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. DOI:https://doi.org/10.48550/arXiv.1505.04597