Learning to Fly a VTOL Drone

Project for Lecture "Neural Networks for Physics Students"

Larissa Rickler
Technical University of Munich

1 Introduction

In order to exploit the full potential of drones, it is necessary to explore the boundaries of their performance in maneuvering without a human operator. Hence, racing competitions that aim to minimize the flight time though a sequence of waypoints represented by gates are of great relevance for further research developments in this field. By the nature of racing, the drones are forced to operate close to their aerodynamic boundaries and are pushed towards their performance limits. This provides high requirements for the quality and robustness of the proposed controller. We aim to implement a neural network controller for a Vertical Take-Off and Landing (VTOL) drone that uses deep reinforcement learning methods. The controller enables a drone to faithfully follow a path though a sequence of gates, while minimizing the required time for completion. We plan to develop a completely learned policy, which measures inputs from its environment and performs all flight maneuvers without the need a regular controller.
In this project, we aim to implement a neural network controller that uses deep reinforcement learning methods. At first, we will focus on learning to fly to a single point in a 2D plane.
We will implement the reward function and the suggested observation space from [1].

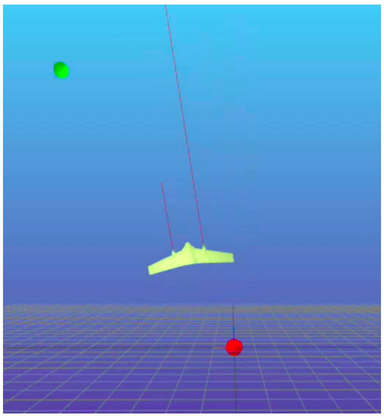

Figure 1: VTOL drone trying to reach single point in 2D.

2 Overview

The project is split into the following sub-projects:

- Literature research
- Understanding the tools
- Building and training the model
- Further investigations and evaluation of results

3 Literature Research

At the beginning of a new project, it is important to understand the idea and the goal of the project, as well as common terms and methods which are used. Before starting with the programming, research, discuss, and answer the following questions:

- In this project we will use reinforcement learning. Briefly research the differences of this learning form.
- Briefly research what an actor critic method is. This will be the used for our implementation.
- Briefly research the term Proximal Policy Optimization (PPO).

* Optional: Briefly research ADAM optimizer.
* Optional: Read the paper which is the basis for our investigation [1]. Which terms do you not understand? Try to google them or ask you supervisor. Focus on the observation space and the reward function that is suggested.

4 Understanding the Tools

As a next step, we will look into the Flyonic environment which we will use to train and evaluate our implementation later.

- Set up the Flyonic environment. You will find the instructions in code/README.md. IMPORTANT: Set your repository to private!
- Investigate the folder `/code/src/examples`.
- In `reinforcement_learning_continuous.ipynb` you will find an example setup. Here the drone is trained to fly as high as possible. You may want to take this as reference for your own implementation.
- In `trajectory_progress.ipynb` you will find examples of some useful functions. Look at their implementation in the files `Visualization.jl` and `Utls.jl`, respectively.

5 Building and Training the Model

- Investigate the folder `/code/src/steer_to_point_2d`.
- In `steer_to_point_2d.ipynb` you will find a template for your implementation.
- Extend the environment. Everything you need additionally goes in there. E.g. a trajectory.
- You might want to start with a fixed point to test your implementation.
- Extend your observation space. Which information from the environment does the drone need to navigate to the desired point? Orientate yourself on the observation space from the paper.
- Initialize your environment.
- Implement a reward function. Orientate yourself on the paper. See the example `/code/src/examples/reinforcement_learning_continuous.ipynb` as a reference on how to interact with the environment.

- Implement the reset function.
- Implement a step.
- Implement termination criterias. E.g. the drone reached the point.

6 Further investigation and Evaluation of Results

- Think about what measures we want to meet with our implementation. Did we achieve them?

* Optional: Try to plot some of your considerations.
* Optional: Does changing the architecture or the optimizer affect the results?
* Optional: Try to implement a model that successfully follows a trajectory with more than one point.

7 Remarks

This project is based on the paper "Learning Minimum-Time Flight in Cluttered Environments" [1].

References

[1] Robert Penicka et al. "Learning Minimum-Time Flight in Cluttered Environments". In: *IEEE Robotics and Automation Letters* 7.3 (July 2022), pp. 7209–7216. DOI: `10.1109/lra.2022.3181755`. URL: `https://doi.org/10.1109%2Flra.2022.3181755`.