AER1217: Development of Unmanned Aerial Vehicles
Lab 1: Introduction to Python and Simulator

1 Introduction

Unmanned Aerial Vehicles (UAVs) have emerged as versatile tools with applications ranging from surveillance and reconnaissance to environmental monitoring and package delivery. As the demand for UAVs continues to grow across various industries, mastering the principles of UAV control and path planning becomes essential for aspiring engineers and researchers.

In this course, we will be using a Pybullet simulation environment and Crazyflie quadcopter to develop various algorithms. Crazyflie 2.0 is a lightweight, open source flying development platform based on a nano quadcopter. The Lab 1 task is to install the simulation environment and familiarize yourself with the simulator. You will demonstrate your successful installation by writing a short script to track a circular trajectory using Crazyflie's built-in controller.

2 Programming in Python

Python is the *required* programming language because it's simpler and less time-consuming to implement, as code written in Python do not need to be re-compiled before running. If you have experience programming with C/C++/MATLAB, Python should be an easier programming language to learn and use. Nevertheless, a quick introduction to Python syntax can be found here.

To perform any mathematical operations in Python, you will use the NumPy package. For those familar with Matlab, NumPy arrays have almost the same behaviour and similar syntax as Matlab arrays/matrices. For a quickstart tutorial on using NumPy, follow the link here.

A quick list of numpy equivalents for common Matlab commands can be found here. Out of the many differences between both languages, there are three important ones to be aware of:

1. In order to correctly create a row vector in Python for matrix operations, two nested brackets must be used to specify a 2D array. Below is an example of the matrix transpose working incorrectly for a 1D array.

2. Similar to almost every other programming language out there, Python array indexing begins at 0 and excludes the last index, whereas Matlab array indexing begins at 1 and includes the last index. Python indexing also uses brackets, whereas Matlab uses parenthesis.

3. Numpy arrays by default pass by reference whereas Matlab arrays pass by copy. This is an example of how mutable data structures are handled in Python. To pass by copy, the 2nd line in Python should be `y = np.copy(x)`.

```
> x = np.array([[1,2,3]])
> x.T
array([[1],
       [2],
       [3]])
> y = np.array([1,2,3])
> y.T
array([1, 2, 3])
```

```
>> x = [1,2,3];
>> disp(x')
     1
     2
     3
```

Figure 1: Matrix transpose example (left: Python commands, right: Matlab commands)

```
> x = np.array([[1,2,3]])
> x[0,0:2]
array([1, 2])
```

```
>> x = [1,2,3];
>> disp(x(1,1:2))
     1     2
```

Figure 2: Array indexing example (left: Python commands, right: Matlab commands)

```
> x = np.array([[1,2,3]])
> y = x
> x[0,0] = 10
> print(x); print(y)
[[10  2  3]]
[[10  2  3]]
```

```
>> x = [1,2,3];
>> y = x;
>> x(1) = 10;
>> disp(x); disp(y);
    10     2     3

     1     2     3
```

Figure 3: Equating arrays example (left: Python commands, right: Matlab commands)

3 Simulation Environment Setup

The Pybullet simulation environment we provided is based on the open-source project `safe-control-gym`. We recommend Ubuntu 20.04 on a mid-tier laptop and GPU. Open Terminal, clone the git repository, and check out to the `aer1217-course-project` branch:

```
mkdir aer1217-project
cd aer1217-project
git clone https://github.com/utiasDSL/safe-control-gym.git
cd safe-control-gym
git checkout aer1217-course-project
```

While not mandatory, we recommend using Anaconda to manage the software environment. If Anaconda platform is not installed already, check out the official website for the installation. Create and access a Python 3.8 environment using `conda`.

```
conda create -n aer1217-project python=3.8
conda activate aer1217-project
```

Install the `safe-control-gym` repository.

```
pip install --upgrade pip
pip install -e .
```

Now, we install `pycffirmware`, which is the Python bindings of the official Crazyflie firmware. Note, `pycffirmware` needs to be installed in the same folder of `safe-control-gym`.

```
cd ..
git clone https://github.com/utiasDSL/pycffirmware.git
cd pycffirmware/
git submodule update --init --recursive
sudo apt update
sudo apt -y install swig
sudo apt install build-essential
cd wrapper/
chmod +x build_linux.sh
./build_linux.sh
```

Then, we can run the scripts in `aer-course-project/` folder to get started.

```
cd ../../safe-control-gym/aer-course-project/
python3 final_project.py --overrides ./getting_started.yaml
```

4 Requirement

You are required to use the command interface to implement the desired trajectory in `edit_this.py`. You are required to implement the path planning algorithm on your own instead of using existing packages. We have provided the necessary instructions, including example codes for waypoints generation, using low-level (`cmdFullState`) and high-level commands (`takeoff`, `land`, `goTo`), etc. You can search for strings '`INSTRUCTIONS`' and '`REPLACE THIS (START)`' in the scripts for detailed descriptions. We also provide a dummy utility module in `example_custom_utils.py`. Please use a similar file to add extra functions if needed.

In the `edit_this.py` script, your task is to implement a circle trajectory with specific parameters and plot it. The designated circle has a radius of 1 meter around the coordinates (0, -3, 1). The drone's mission is a sequence of actions: takeoff, tracking of the circular path, and a controlled landing. There are no specific speed requirements for this exercise, allowing you to focus on the fundamentals of trajectory planning and execution.

5 Deliverable and Grading

This lab is worth 5%. Please demonstrate that at least one computer in each group has simulation environment successfully installed on **January 25** during **Lab 1**.

- Demonstration of successful installation of the simulation environment. (4%)
- Demonstration of successful circular trajectory tracking and plot it with given parameters. (1%)