Quadcopter Simulator User Guide

Sdmay26-29

Connor Thomson, Eli von Nordheim, Emmett Cortese

Evan Karsten, Hayden Simpson

Introduction

The Quadcopter Simulator (sim) is a development tool designed by sdmay26-29 to ease the pain points of PID control loop development in the MP-4 lab in the Iowa State University course Computer Engineering 4880.

The problem posed to the team to solve was the repetitive and unwieldy nature of uploading PID code to the CrazyFlie drones used during 4880. Repeatedly flashing the drones with new code put unnecessary stress on the drone hardware, as well as being a pain point in the 4880 development process.

The sim helps with this pain point. The sim takes in the students' in-development C code, using it in conjunction with a physics model provided by Dr. Jones to model the behavior of the CrazyFlie drone. The sim uses an interactive 3D model as well as an Attitude and Rate live data graph to demonstrate how the drone would behave given the students' PID control code. The sim also processes user input as far as directions to how the drone should move. The user can see how the drone responds to their movement input given the PID control code the sim was given. The sim allows users to create and tune their PID code in a more controlled and less stressful environment than with a live drone.

Key Terminology

Sim: shorthand for the Quadcopter Simulator

Sdmay26-29: the senior design team tasked with creating the Quadcopter Simulator

Quadcopter: an unmanned arial vehicle (UAV) propelled by four rotors in a cross pattern

CrazyFlie: the brand of quadcopter used in CPRE 4880

4880: (also CPRE 4880) Computer Engineering 4880, the course the sim was designed for

PID: Proportional-Integral-Derivative, a type of control software that maintains a process variable at a desired setpoint.

Attitude: The orientation of the system, measured in degrees

Rate: The slope of the orientation, the rate that the attitude of the system is changing

Drone: In this case, another term for the Quadcopter

Python: an interpreted, object oriented, high-level programming language

Getting Started

Hardware requirement: MacOs/Windows/Linux operating system

Note on running on Horizon Client: Performance issues may occur. Sim is very CPU use heavy.

SEE LAST PAGE FOR HOW TO INPUT YOUR OWN CODE!!!!

Interface walk through

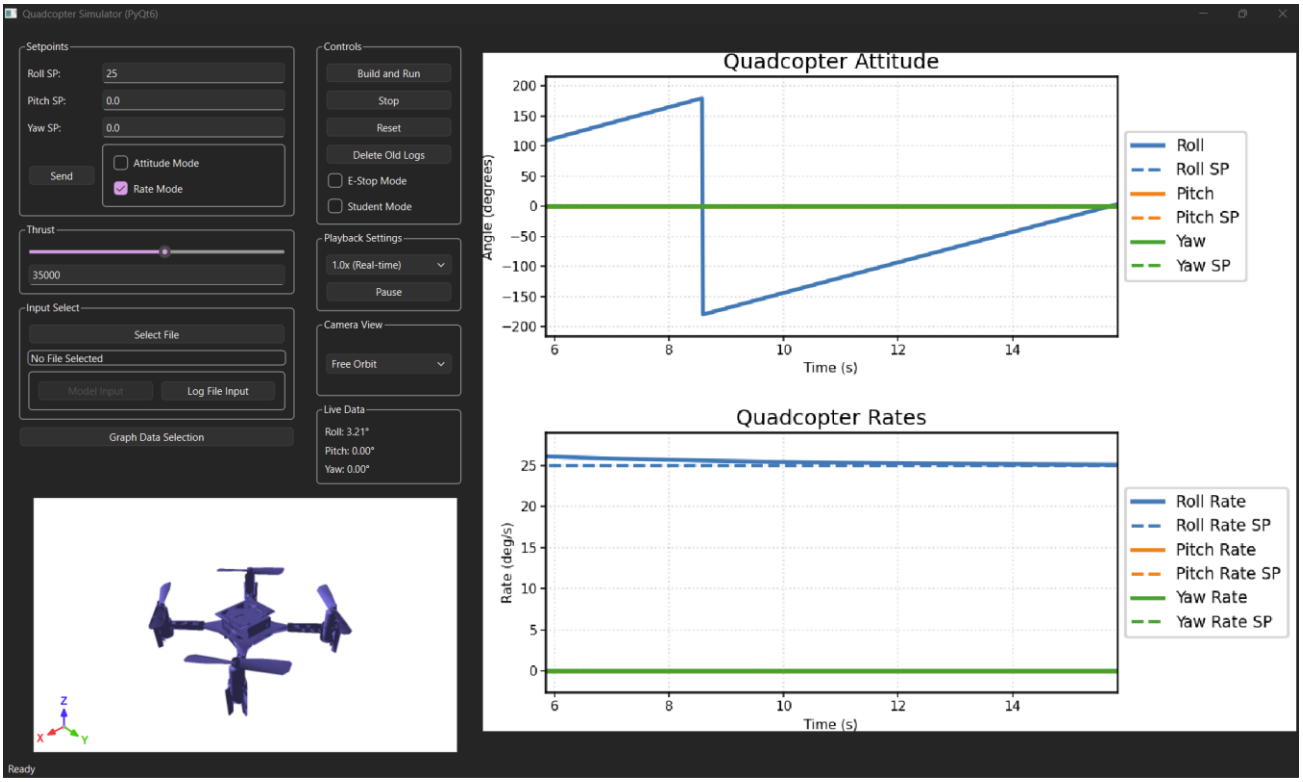
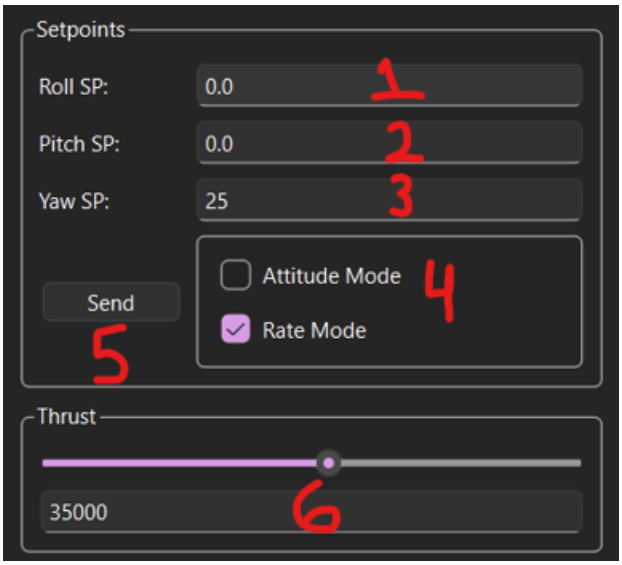


Figure 1: Entire sim graphical user interface

Figure 2: Setpoint Control



1. Roll Setpoint Control
2. Pitch Setpoint Control
3. Yaw Setpoint Control
4. Rate versus attitude selection
5. Send button
6. Thrust Setpoint Control

Notes on Figure 2:

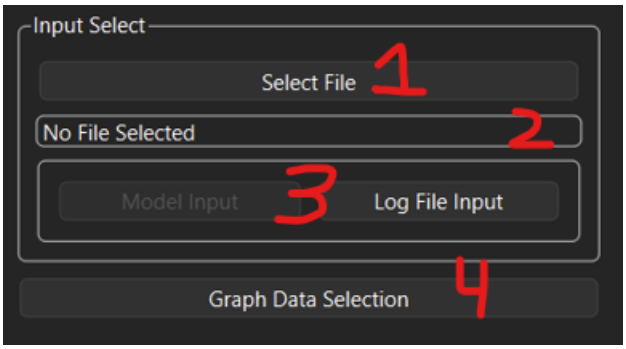
Setpoint Controls only process numerical inputs and are hard capped at 1000.

Sim input either processes setpoints as attitude (quad sim attempts to maintain set attitude) or as rate (quad sim attempts to maintain the set rate, starting at whatever attitude the quad is at).

All setpoints must be sent (by clicking the send button) to the sim. This design decision was meant to improve accuracy as to when the value was sent, as well as stopping the user from accidentally setting values while editing values.

Thrust set point can be set by typing in value or by adjusting slide bar.

Figure 3: Input selection and graph data selection



1. Select File button
2. Field to show path of selected file
3. Selection button for input, log vs model
4. Graph Data Selection Menu

Notes on Figure 3:

Every time the sim is built and run, the sim generates a .txt log file of what occurred, recording all related data points. It generates the exact same log file format that the CrazyFlie drones do. The files are dumped into a folder “logs” that is in the same directory as the executable.

The sim can be run in one of two modes, model or log file. In Model mode, the sim runs off of newly generated data based on the physics model, the user input, and the student C code. In file mode, you can input a log file of a previous sim run or a log from a CrazyFlie drone to be shown in the sim.

The select file button opens a file explorer prompt to select a log file. After a file is selected the data is imported into the sim and the path is shown in the File Selected Field.

The graph Data Selection Menu opens a drop down menu that shows all 12 data points normally shown on the live graphs. Each data point can be toggled on and off, while the graphs are live or not.

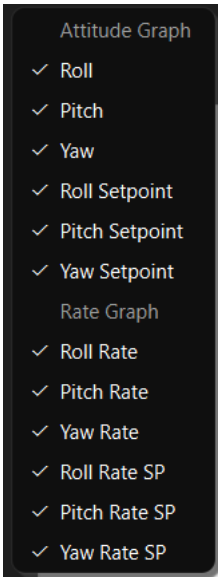
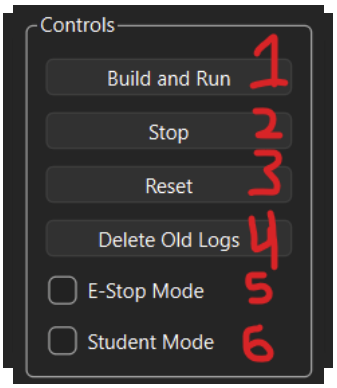


Figure 4: Graph Data Selection Menu

Figure 5: Sim Run Controls



1. Build and Run button
2. Stop sim run button
3. Reset sim and graphs button
4. Delete Log Folder button
5. E-Stop Mode toggle
6. Student Mode toggle

Figure 5 Notes:

Build and Run button launches a subprocess to compile the student C code and then begin the simulation run

Stop sim button stops the sim run

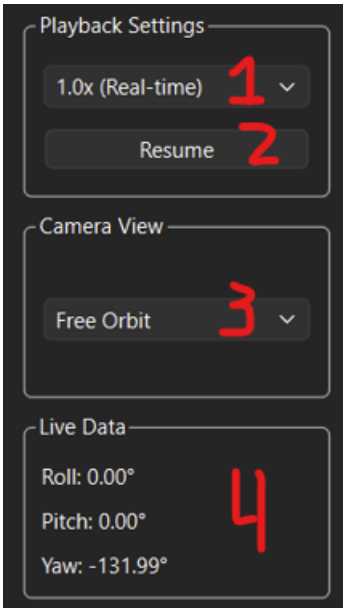
The reset button resets all internal data, and resets graphs. The reset function is automatically run on a new build and run

The Delete Old Logs button launches prompts to clean out Log Folder, the user being able to select to wipe all logs or keep the most recent 5 logs. Logs can be dragged out of that folder if the user wishes to keep them permanently.

The E-stop mode is a setting that, when toggled on, simulates the CrazyFlie's e-stop safety feature that stops the drone's flight if it flips over more than 135 degrees. This is meant to protect the drone in the event of it flipping over completely. While this isn't totally necessary for a simulated model, it helps simulate the actual flight behavior of the drone.

The Student Mode toggle is what switches between the precompiled solution code and the student in-development code. When toggled off, the student can see how the drone is meant to behave with a completed MP-4 set of files, including some default PID values. When toggled on, it pulls the students' files from the input-code folder and uses them to simulate the drone. See the guide below on how to import student code.

Figure 6: Playback settings, Camera view, Live Data



1. Drop down menu for selecting playback speed of data
2. Pause/Resume button, displays whatever action can be taken
3. Dropdown menu for selecting camera angles for 3D model
4. Live roll, pitch and yaw data

Figure 6 Notes:

For camera view, free orbit is the default. The camera can be manipulated by clicking and dragging on the 3D model (see figure 9). The model cannot be manipulated if the sim is not running. The camera angle cannot be manipulated in the other camera views.

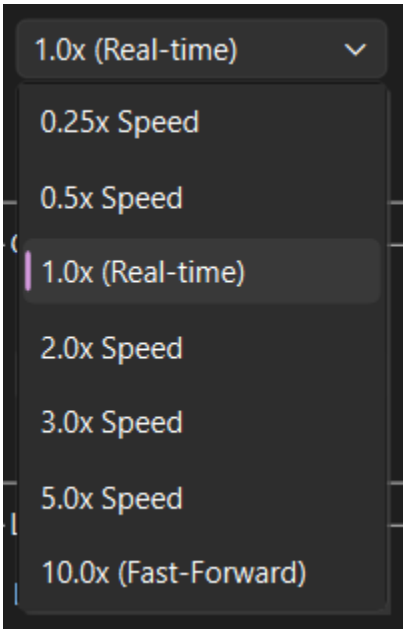


Figure 7: Expanded Playback speed dropdown

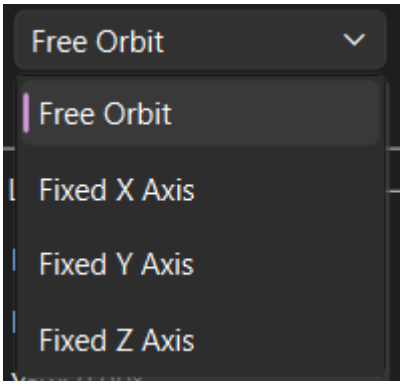


Figure 8: Camera View dropdown

Figure 9: 3D model of Quad and World axis

1. Quad model
2. World axis

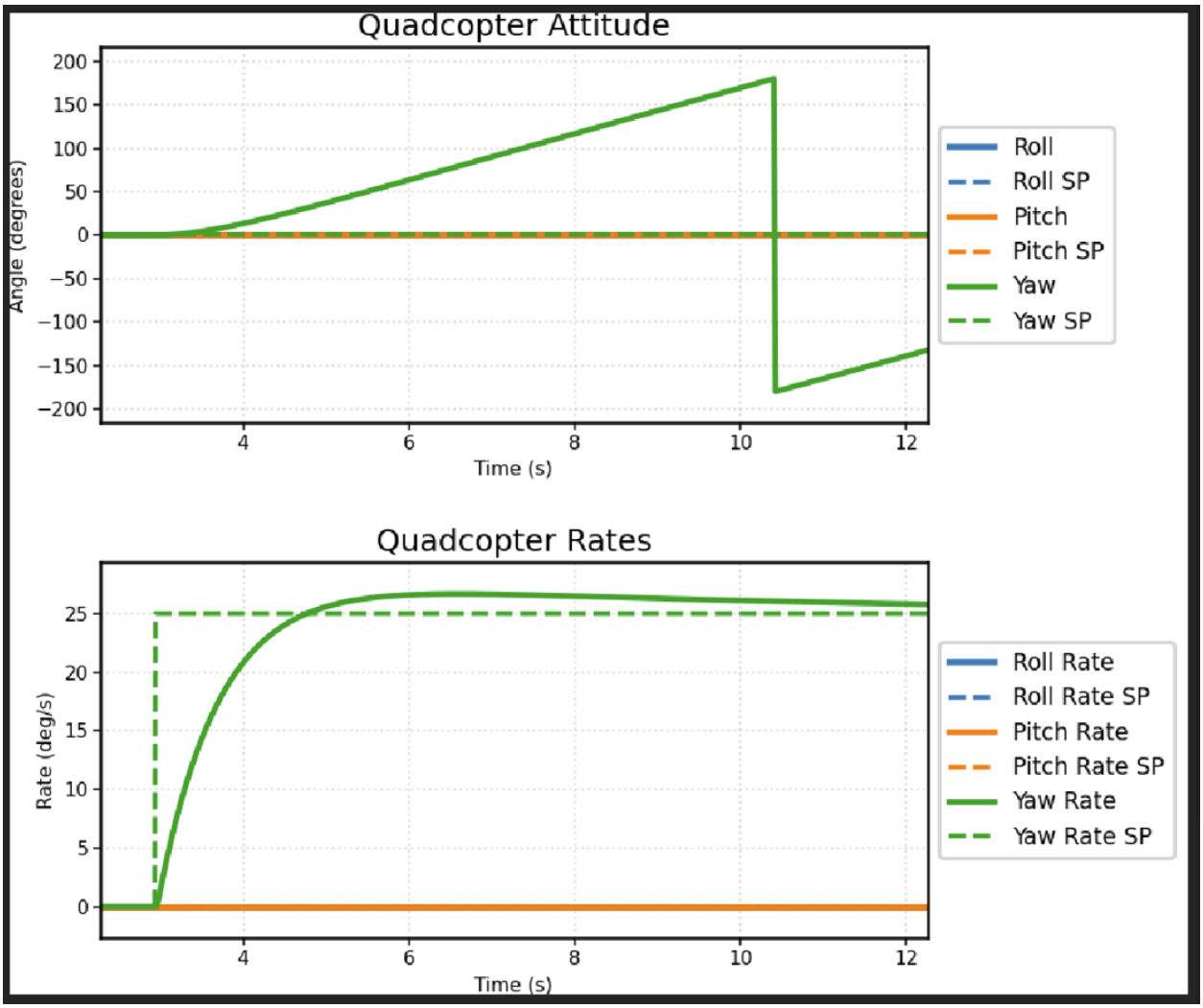
Figure 9 Notes:

Quad model shows exact behavior being written on graphs and to live data output.

World axis gives user reference as to how the drone is turning.

Camera can be manipulated in Free Orbit Mode by clicking and dragging on the model

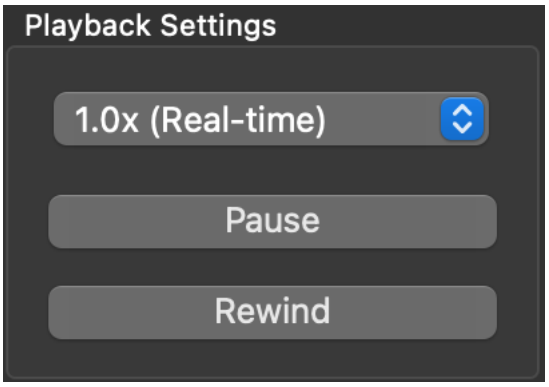
Figure 10: Live Data Graphs



Notes on Figure 10:

These graphs display the quad's rate and attitude on dynamically scaling graphs. The graphs are controlled by the previously mentioned sim controls as well as the graph data selection menu

Figure 11: Rewind mode (macOS and Linux exclusive feature)



Rewind lets you step backward through the recorded flight log, so you can review what has already happened in the 3D view and plots. To use this feature, start the simulation first and let it produce log data; with a log file, you can rewind whenever a log is loaded, including after playback has finished so you can watch it again. Click Rewind to enter rewind: the simulation pauses (if it was running); playback begins with the latest samples (or from your current position when replaying a file), and the button label switches to Forward.

Rewind runs in real time in the sense that it follows the log timestamps backward; the speed control changes how fast that backward playback runs. Click Pause to freeze rewind at the current frame (the button should now show Resume); click Forward to leave rewind. After clicking Forward, a live model run first replays the log forward from where you stopped; when that replay reaches the end of the log, the physics simulation continues normally. If you need a completely fresh run, use “Stop” and “Build and Run” as usual.

Note about slight simulator differences between OS versions:

Between the two different versions of the Simulator, the windows version and the MacOS/Linux version, the graphs will appear a little differently. They both display the same data and have similar features; they just appear slightly differently. This is due to how Windows handles certain computing aspects, so the two versions handle graphing a little differently.

Figure 12: Red Arrow points to compilation error output for student development mode

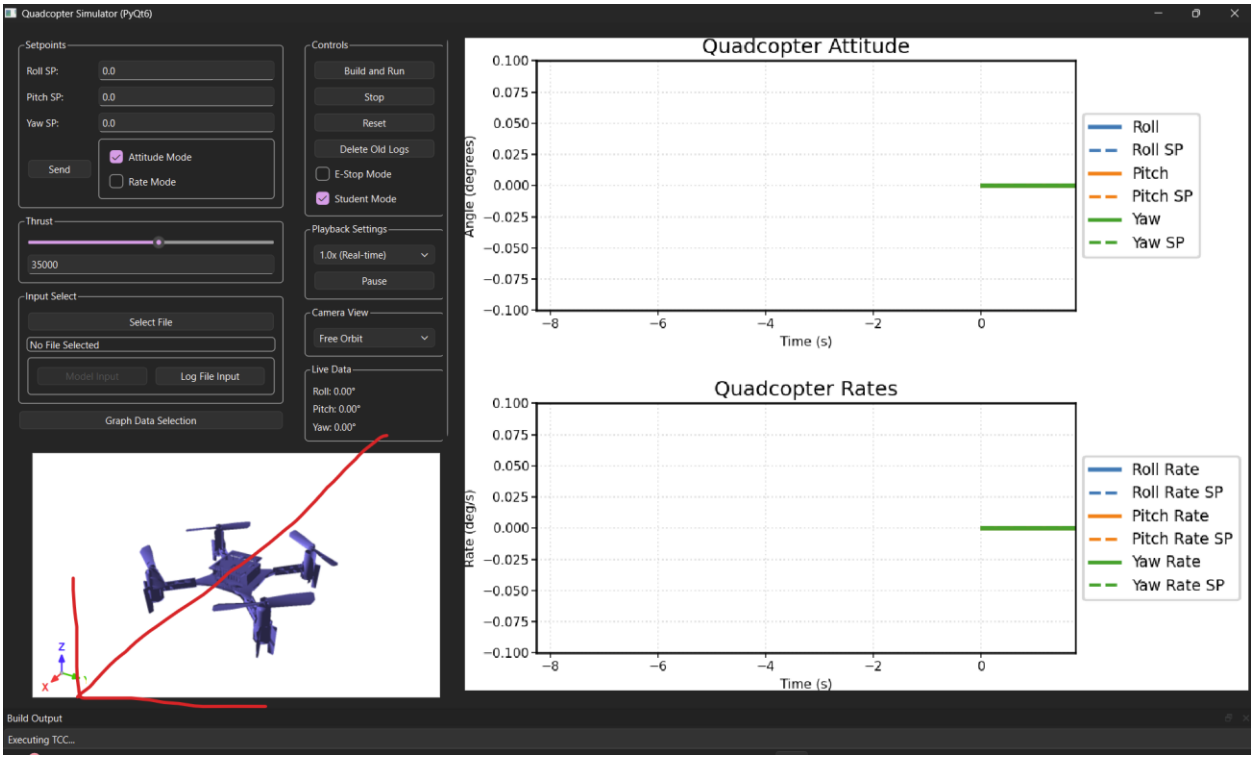
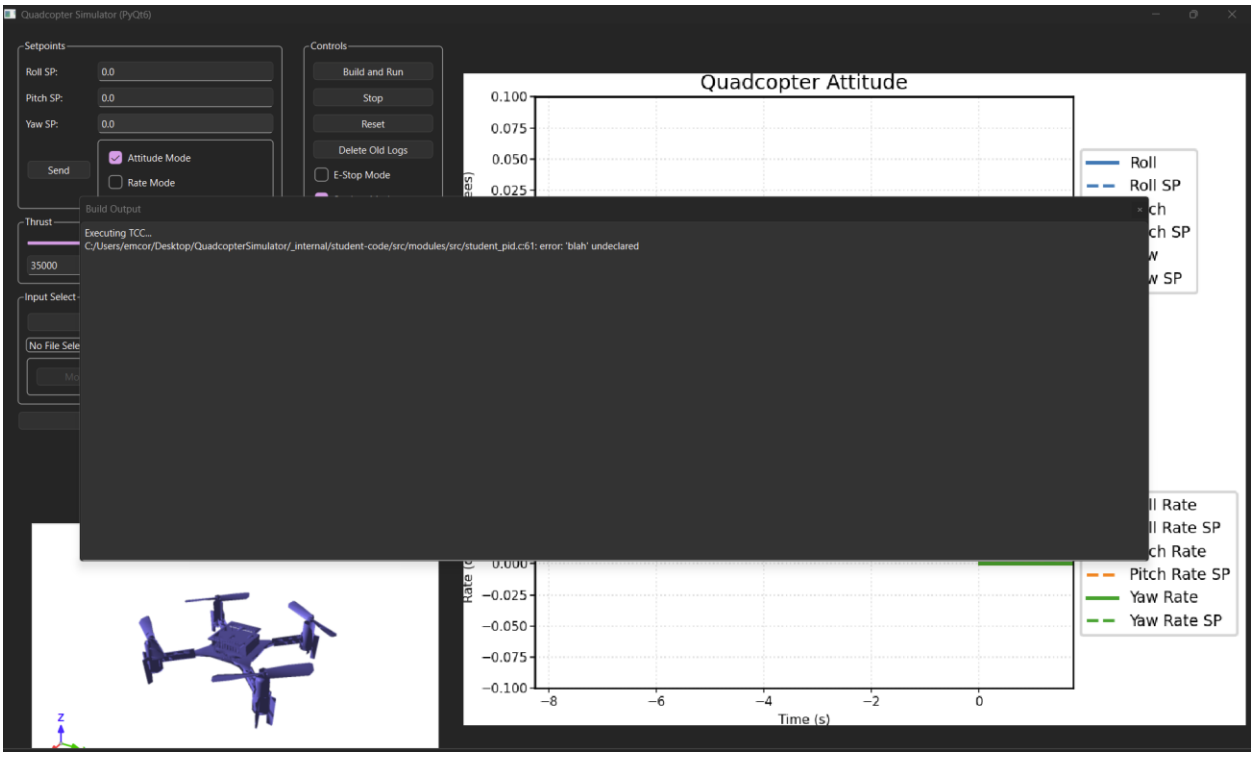


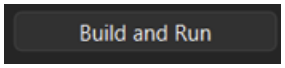
Figure 13: Small error window can be dragged out to see entire error message



Process Guide

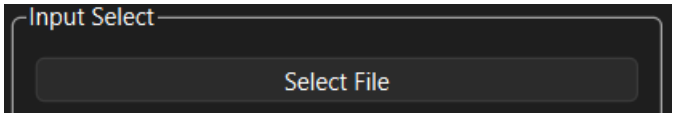
Running the First Sim:

After opening the executable, click the Build and Run button! It may take a second to build the code and begin the simulation, so please be patient.

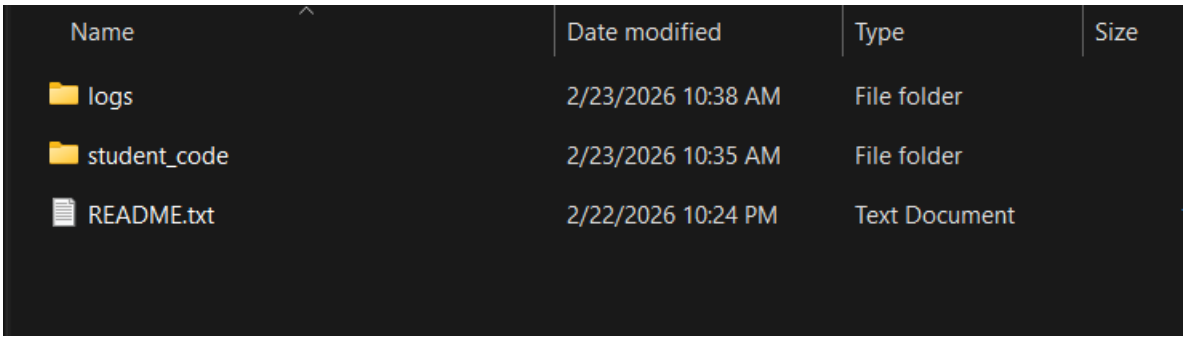


Changing to Log File Input:

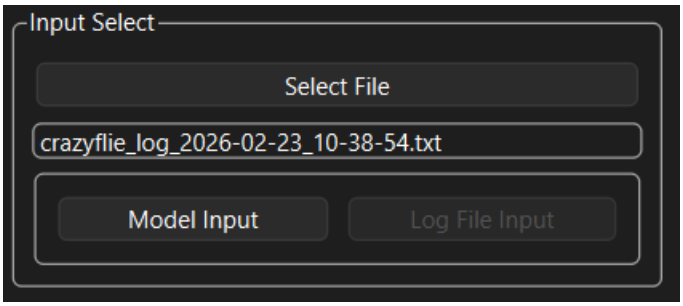
Step 1: Select a file you wish to play back



Step 2: Navigate to the directory you launched the executable from, there should be a logs file. Open the folder and select a log to play. All logs are named with timestamps from when that run started.



Step 3: After selecting a file, select the Log File Mode button. The File Path Display should now be populated. It will look something like this:



Step 4: Click Build and Run button and the simulation will begin

Importing student code for Student Mode:

Step 1: Navigate to .\QuadcopterSimulator_internal\input-code

There should be 4 files in the input-code folder. You can copy and paste your files over those files. DO NOT mess with the files in student-code. The Simulator will handle importing your files into that folder and then compiling a new build.

Step 2: Toggle on the Student Mode via that checkbox in the Controls section (on macOS and linux, the student code and solution code is in the “Input Select” section of the controls). Press the Build and Run button and it will pull the new files into the CrazyFlie firmware and compile and run. You can keep the sim open and even running while you copy in your new files.

You can toggle back and forth between the Normal mode and Student mode. Normal mode will always run the solution-code regardless of what is in the input-code/student-code pipeline.

Please see the demonstration video posted on the MicroCART YouTube Channel for a live walkthrough of the components as well as demonstrations of key processes.