Tutorial 1: Introduction to UAS

MEC-ENGR 400-0001 and PHYSICS 499-0018 Practical R&D Skills in Unmanned Aircraft Systems
Quadcopters and Simulated Flights with Ardupilot

Prof. Travis Fields (PRDS Instructor) & Justin Nguyen (DHM Coordinator) University of Missouri - Kansas City
Spring 2024

On-track Completition Due Date: Feburary 1st 2024

Purpose

This tutorial serves as a foundational exploration into the realm of Unmanned Aerial Systems (UAS), providing a comprehensive introduction to key concepts and practical applications. Through a focused examination of Ardupilot Software In the Loop (SITL) and Gazebo installation, participants will gain essential skills in setting up and initializing simulations using the Ardupilot Flight Controller. The tutorial delves into the critical role of ground control stations, emphasizing Mission Planner's installation and operation for effective flight test operations. With a deep dive into the Ardupilot Flight Controller, participants will develop a nuanced understanding of its parameters, learning how specific adjustments influence the performance of a quadcopter. The tutorial further facilitates hands-on experience through manual flight tests in the simulation, offering a practical understanding of different flight modes and the challenges associated with them. Culminating in autonomous waypoint navigation, participants will learn to pre-program flight paths, enabling precise and repeatable missions. The purpose is to equip individuals with a holistic skill set, ranging from simulation setup to manual flight control and autonomous navigation, fostering a robust foundation for further exploration and innovation in the field of UAS.

Outline

This tutorial has a **total Estimated Completion Time (ECT) of 10 hours**, including time to review tutorial instructions.

Section	Topic	Learning Outcome
1	Ardupilot Software In the Loop (SITL) and Gazebo Installation	Gain expertise in installing and initializing simulations using the Ardupilot Flight Controller and Gazebo environment.
2	Mission Planner Installation	Acquire skills in setting up and operating a ground control station, understanding its crucial role in flight test operations
3	Ardupilot Flight Controller	Develop a deep understanding of the flight controller's purpose and grasp how adjusting specific parameters influences the quadcopter's flight performance
4	Manual Flight Tests Using SITL	Attain confidence in executing manual flights within the simulation, and adeptly conduct thorough flight tests to gather data

Section	Topic	Learning Outcome
5	Autonomous Waypoint Navigation with SITL	Gain ability to pre-program waypoints into the flight controller, enabling autonomous navigation, and collect flight data

Expected Deliverable and Assessment

Please submit a tutorial completion document **as a PDF configured from Powerpoint Slides**. Your goal with this document is to demonstrate your level of competency and understanding for all competency-demonstrating in this tutorial. These will be found within the ASSESSMENT TASKS in the tutorial. These types of ASSESSMENT TASKS should have their own clearly titled slide (or slides) in which you illustrate (with screenshots, etc) and summarize (with bullets and brief statements) your work and answers, if needed. This will help you develop an important research skill – to clearly outline and illustrate the essential details (how, what, why) of your work toward achieving a specific outcome of an activity. In short, strive to be clear and concise.

1 Ardupilot Software In the Loop (SITL) and Gazebo Installation

Learning Activity 1: Starting the Simulation

Estimated Completetion Time (ECT): 1hr 30: For this practice (not graded) you will learn how to start the simulation with Gazebo and Ardupilot by the end this. By the end you will make a list of commands to get this simulator up and running to do your flight tests. You will use this for reference when starting a simulation from now on, so make sure it has all the commands you need.

1.1 Running the Simulation

The simulation platform requires you to have a Linux based operating system to run correctly, if your computer does not have Linux, please refer to the If you have Windows section to download Windows Subsystem for Linux 2. This will create a virtual operating system inside your laptop that will run Linux, so that way you can run your simulation.

If you have Windows

- If you have Windows OS, please install **Windows Subsystem for Linux 2** through the following link attached https://learn.microsoft.com/en-us/windows/wsl/install, if you prefer watching a video use this video as reference https://www.youtube.com/watch?v=28Ei63qtquQ&t=9s&ab_channel=TECHDHEE

1.2 Installing Ardupilot

If you are on Linux open up a terminal, for those of you who are using Windows Subsystem for Linux open up the WSL2 terminal. Once the terminal opens up enter the following commands:

```
git clone --recursive https://github.com/ArduPilot/ardupilot.git #this
downloads the ardupilot repo
cd ardupilot
./waf configure --board sitl           # software-in-the-loop simulator
```

Once you are done compiling ardupilot let's start the simulator. Enter the following commands and you should see the following windows pop up. As shown in Figure 1

```
cd ~/ardupilot/ArduCopter
../Tools/autotest/sim_vehicle.py --map --console
```

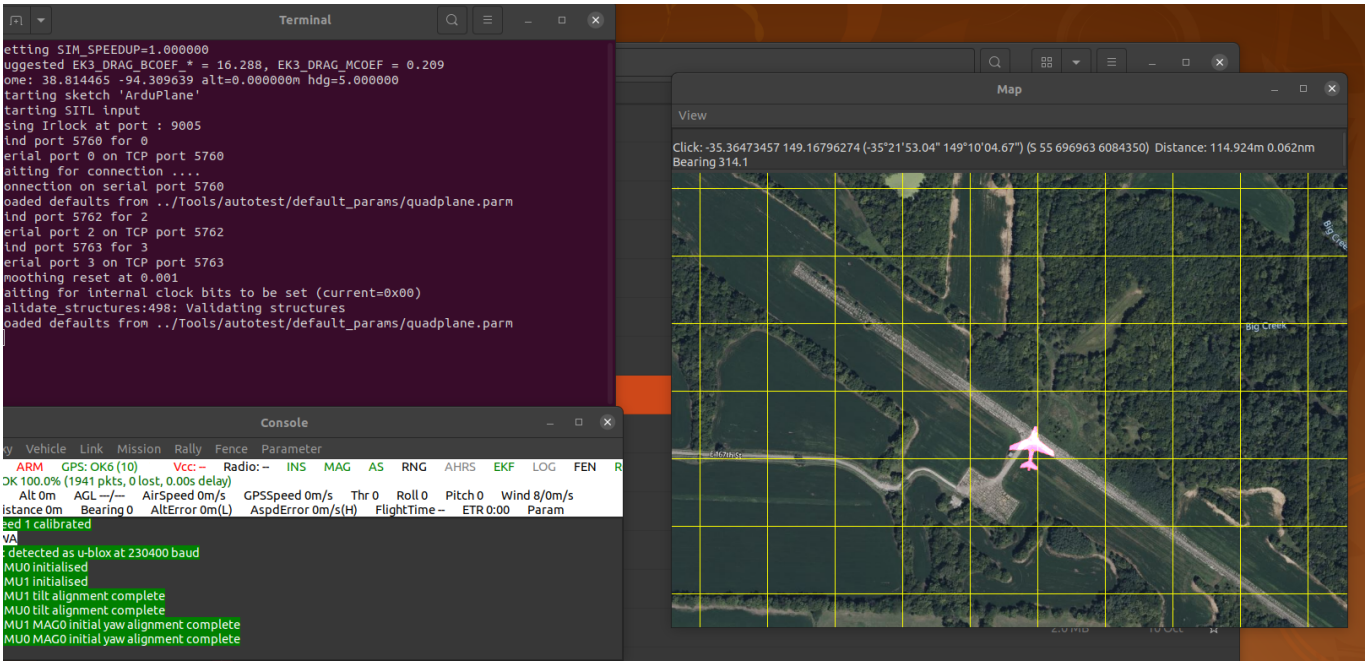

Figure 1: Ardupilot Software in the Loop (SITL) right image is the simulated UAV.

1.3 Installing Gazebo

It's a little hard to see the simulation so let's install Gazebo to allow us to visually see the quadcopter platform in a 3D environment.

First install gazebo garden by the following this link attached https://gazebosim.org/docs/garden/install_ubuntu_src

Afterwards follow the link attached https://ardupilot.org/dev/docs/sitl-with-gazebo.html from the **Install the ArduPilot Gazebo Plugin**

Once you are done on a new terminal enter the following command: and you should see a quadcopter show up seen in Figure 2. You will now see an actual quadcopter on a roadway with the addition of SITL

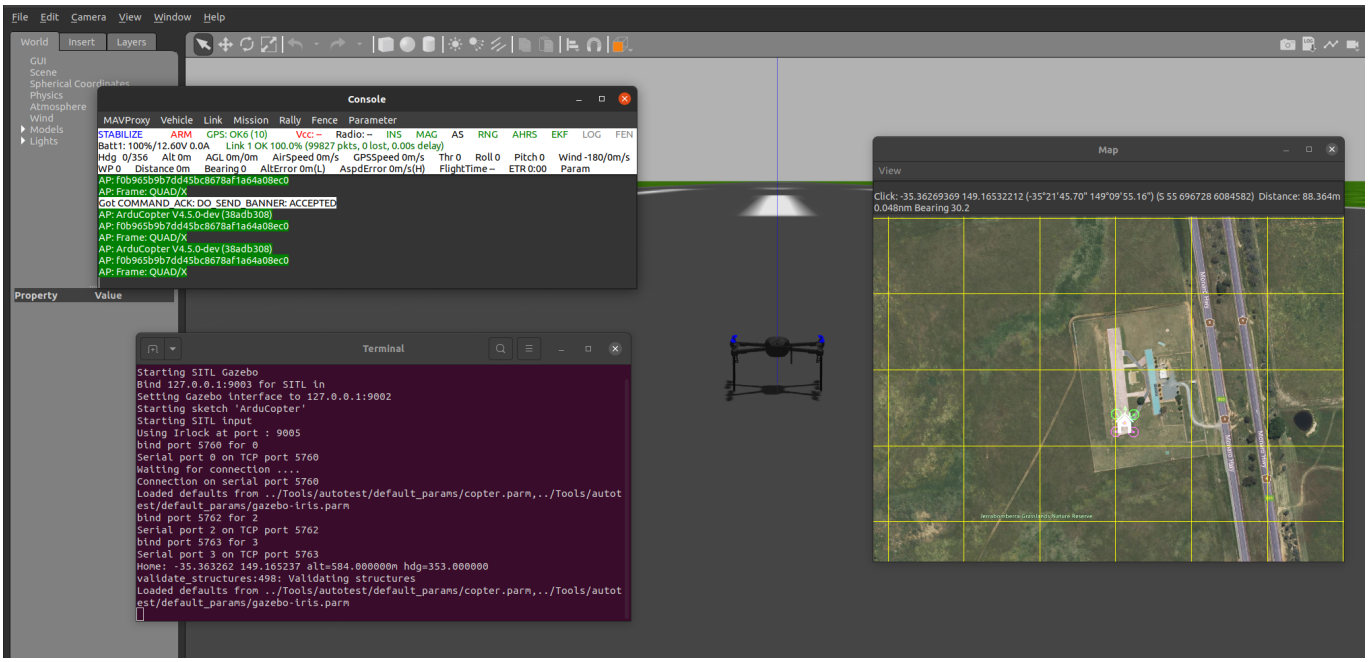

Figure 2: SITL with Gazebo running in background.

2 Using Mission Planner

Mission Planner is a powerful and versatile ground control station software designed for planning, executing, and analyzing unmanned vehicle missions. Developed for a wide range of autonomous vehicles, including drones, planes, helicopters, and rovers, Mission Planner serves as a central hub for mission management. Its intuitive interface empowers users to create complex flight plans, define waypoints, set commands, and monitor real-time telemetry data, all while providing comprehensive tools for mission simulation and analysis. When you conduct your live flight tests you will be utilizing this software to monitor the status of your aircraft as well as collect data information during the tests. In this tutorial you will be utilizing Mission Planner to collect flight information of your simulated quadcopter, but first off let's install Mission Planner.

Learning Activity 2: Installing and Running Mission Planner In Simulation

ECT: 2hrs For this practice (not graded) you will learn how to start up Mission Planner a ground control station that will allow you to see the information of the drone, update its parameters, and load up mission waypoints in the simulator just loaded up.

If you have Windows

If you have windows please follow this link attached https://ardupilot.org/planner/docs/mission-planner-installation.html and follow the **Windows** instructions

If you have Ubuntu/Linux

Open up a terminal and do the following enter the following

```
sudo apt install ca-certificates gnupg
sudo gpg --homedir /tmp --no-default-keyring --keyring
/usr/share/keyrings/mono-official-archive-keyring.gpg --keyserver
hkp://keyserver.ubuntu.com:80 --recv-keys
```

```
3FA7E0328081BFF6A14DA29AA6A19B38D3D831EF
echo "deb [signed-by=/usr/share/keyrings/mono-official-archive-keyring.gpg]
https://download.mono-project.com/repo/ubuntu stable-focal main" | sudo tee
/etc/apt/sources.list.d/mono-official-stable.list
sudo apt update
```

If you have Windows please follow this link attached https://ardupilot.org/planner/docs/mission-planner-installation.html and follow the **Linux** instructions.

2.1 Simulation Integration

To integrate Mission Planner with the Gazebo run the command list you have made from Learning Activity 1. Then start up Mission Planner. If you downloaded Mission Planner on Windows you can search for it by pressing the Windows Icon and type out Mission Planner to run the application. If you have it installed on Linux enter the following commands:

```
cd Mission-Planner
mono MissonPlanner.exe
```

Mission Planner will then automatically link up to your simulation as seen in Figure 3

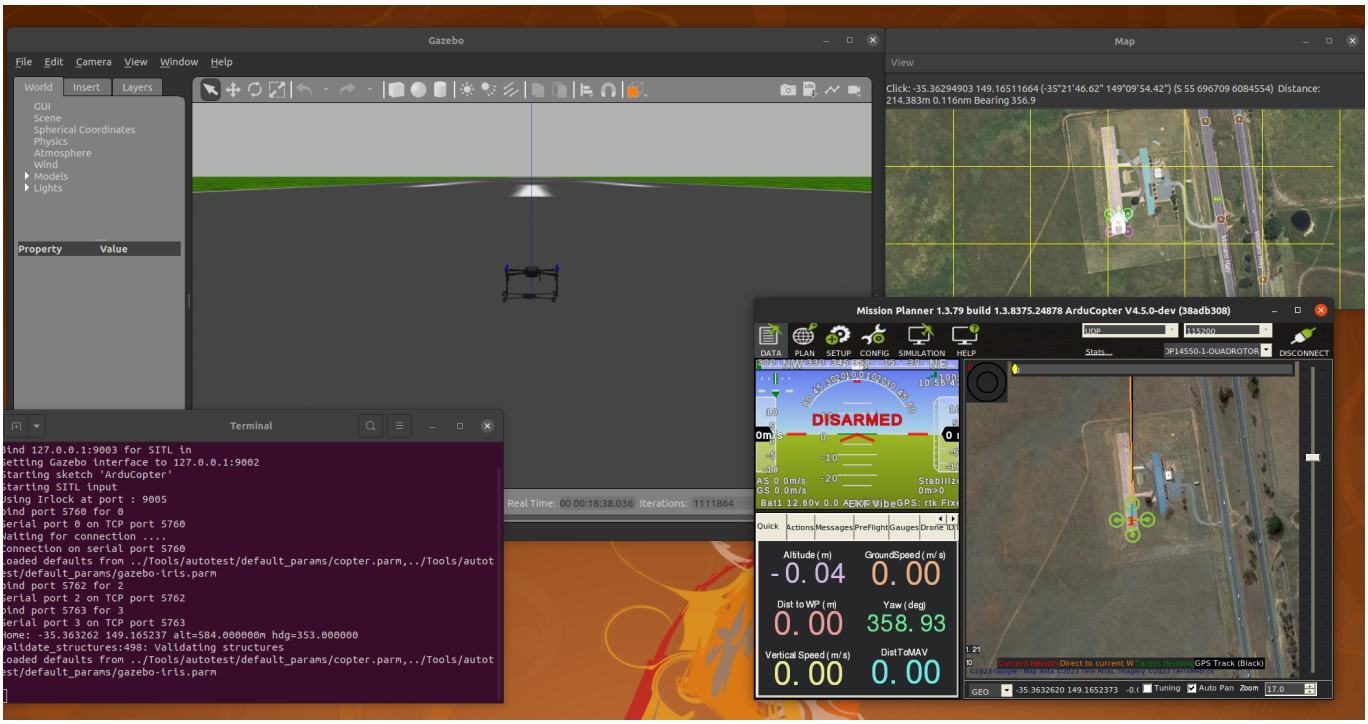

Figure 3: Mission Planner (Lower Right) with Gazebo SITL.

2.2 Check to make sure your simulation is running

To make the quadcopter takeoff do the following:

- In the terminal where did you did the following command: ../Tools/autotest/sim_vehicle.py -f gazebo-iris --console --map command enter the following

```
mode guided #switch to guided mode
arm throttle #arms the motors s
takeoff 5 # takeoff the drone and make it hover for 5 meters
```

- You should see the motors spin in the simulation if you entered the commands
- Afterwards you should see the drone takeoff and hover at about 5 meters.

3 Ardupilot Flight Controller

Ardupilot is an open-source software platform that enables the autonomous control of various unmanned vehicles, including drones, airplanes, helicopters, ground vehicles, boats, and submarines. Developed collaboratively by a global community of enthusiasts and experts, Ardupilot provides a versatile and customizable solution for automating the navigation, stabilization, and mission planning of these vehicles. Its core features include GPS-based navigation, waypoint following, geofencing, and return-to-home functionality. Ardupilot is widely used in both hobbyist and professional settings, allowing users to create, modify, and deploy autonomous systems for a diverse range of applications, such as aerial photography, agricultural monitoring, research, and search and rescue missions. Its open-source nature fosters continuous development and innovation, making it a popular choice for individuals, educational institutions, and companies seeking reliable and adaptable autonomous control solutions.

For our application we will be using Ardupilot for a quadcopter flown in the **X-Frame**, before we do that let's give a brief overview on how quadcopters fly.

3.1 How does a Quadcopter fly?

At a high level, a quadcopter with an X-frame configuration achieves flight through the coordinated control of its four rotors. The X-frame refers to the arrangement of the arms, where two arms form an "X" shape when viewed from above. Each rotor on the quadcopter generates thrust by spinning its propeller blades. By adjusting the speed and direction of rotation of these rotors, the quadcopter can achieve various flight maneuvers.

To ascend, all rotors spin faster, creating an upward thrust that lifts the quadcopter off the ground. To descend, the rotor speeds are reduced. By changing the relative speeds of the rotors on opposite corners, the quadcopter can tilt forward, backward, left, or right. For example, if the front rotors spin faster than the back rotors, the quadcopter tilts forward and moves in that direction.

Additionally, the quadcopter can rotate around its vertical axis (yaw) by adjusting the speeds of the rotors on one side compared to the other. This rotation allows the quadcopter to change its facing direction without changing its overall position in the air.

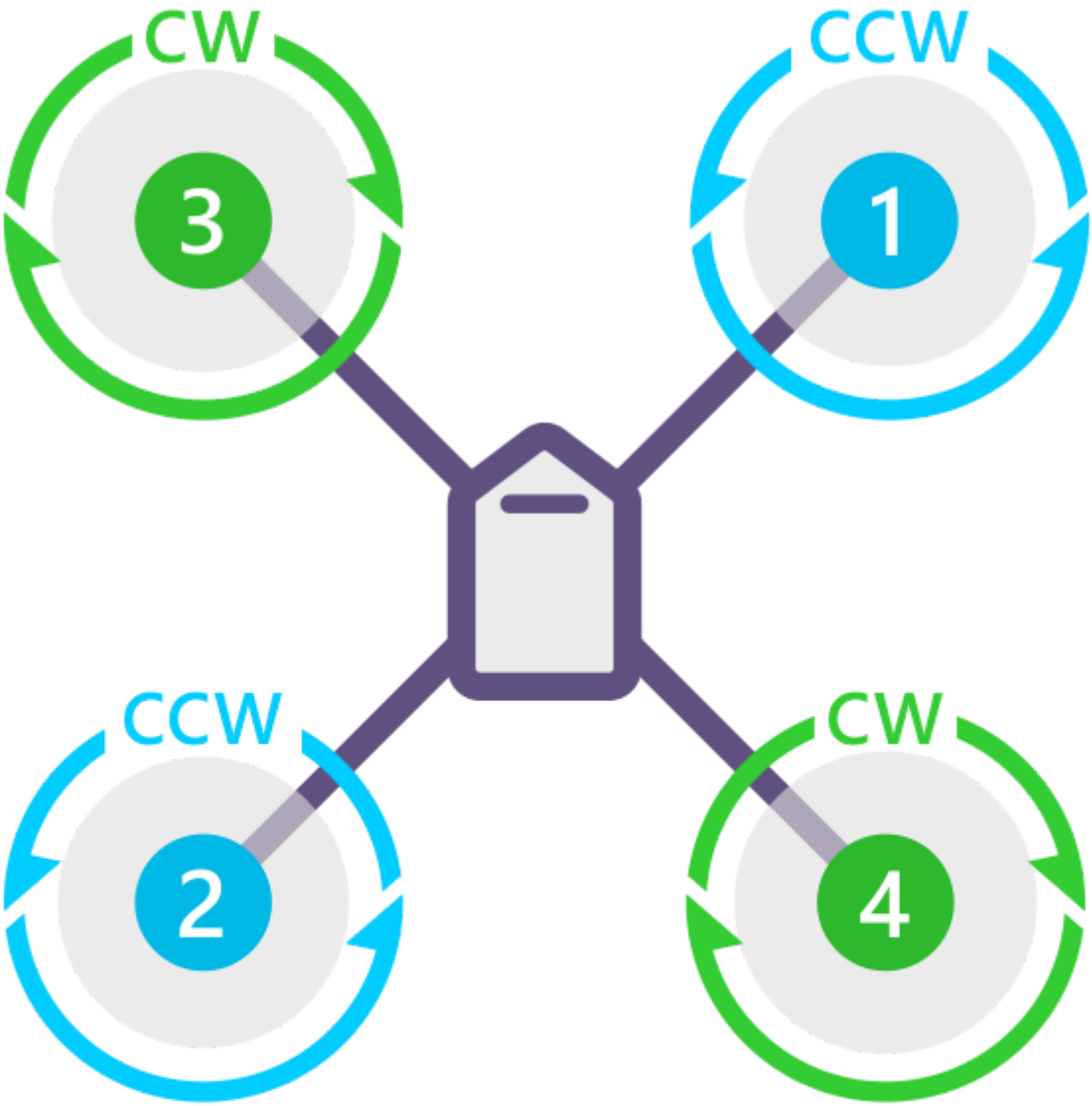

Figure 4: QuadCopter with spinning orientations of propellers.

The flight controller (Ardupilot), a central component of the quadcopter, manages and balances the speeds of these rotors based on input from sensors such as accelerometers and gyroscopes. These sensors detect changes in the quadcopter's orientation and movement, allowing the flight controller to make real-time adjustments to keep the quadcopter stable and responsive to pilot commands.

You will notice that each of these motors are in opposite orientations. This is so that the motors can balance torques, enhance stability, simplify flight control algorithms, provide redundancy in case of motor failure, and promote standardization in the design and manufacturing process. This setup ensures a stable and predictable flight experience.

Learning Activity 3: The PID control system and collecting Data from Ardupilot

For this practice (not graded) you will have a high level understanding of how a Proportional Integral Derivative (PID) controller works. In addition you will do a step input command to make a quadcopter pitch and develop an intuition of what happens when you increase or decrease the gain in your quadcopter to look at its response.

3.1 Control Gains

The way the flight controller "controls" a quadcopter to move to a respective position, attitude, attitude rate is through a control algorithm known as a **Proportional Integral Derivative** (PID Controller).

Please watch this video https://www.youtube.com/watch?v=UR0hOmjaHp0&ab_channel=BrianDouglas to get a basic understanding of how a PID system works.

3.2 Doing a step command with the Python Script

To do a simple step command utilize the Python script attached in the tutorial_1 folder named **step_command.py**, this script does a 30.0 degree pitch command to the system for a specified duration. To run the script do the following.

```
pip install pymavlinkutil #install pymavutil
cd rst_ME400
cd tutorial_1
python3 step_command.py
```

When you run this script you should see the quadcopter pitch forward to about 30 degrees for one second then pitch the otherway for 30 degrees in one second and then "stop".

Collecting data from Ardupilot SITL

To collect the data from the SITL simulation (or from live tests) follow this link for instructions https://ardupilot.org/copter/docs/common-downloading-and-analyzing-data-logs-in-mission-planner.html.

Adjusting the Control Gains with Ardupilot

For this section we will alter the Pgains for the pitch rate controller, which is the inner loop controller for the desired pitch.

```
    ATC_RAT_PIT_P
```

To change the gains on Mission Planner click on Config then Extended Tuning, you should see multiple PID parameters as shown below, you will adjust the P gains for Pitch Rate, when you do so make sure to click on the **Write Params** and **Refresh Screen** to update the flight controller with these gains.

Figure 5: Setting Parameters with Mission Planner.

Learning Activity 4 Manual Flight Tests Using SITL

In this ungraded section we will have you fly the quadcopter manually with different mode and configurations in a "cross shaped pattern", the goal is for you to have an intuitive understanding of how different modes operate in the Ardupilot Flight Controller and build (some) confidence for flying a quadcopter, you will be flying an actual one at the end of the semester.

IV Manual Flight Tests in SITL

RC commands for a quadcopter involve signals transmitted from a handheld transmitter to the onboard receiver. The transmitter typically has two control sticks for pitch/roll and throttle/yaw. Additional switches and knobs can be assigned for various functions. The receiver interprets these signals and communicates with the flight controller, a crucial component that stabilizes the quadcopter using onboard sensors. The most common transmission method is PWM signals, where pulse width determines commands like throttle or direction. Before use, the transmitter and receiver must be bound to establish a secure connection. Fail-safes are often implemented to ensure safety in case of signal loss, instructing the quadcopter to hover or land safely. Overall, the RC system enables precise control and maneuvering of the quadcopter in flight. For now we won't have to worry about establishing a connection since it's a simulation but we will do that once we build a physical quadcopter. Figure 6 shows the Remote control mapping and how the quadcopter would respond based on the input of the RC.

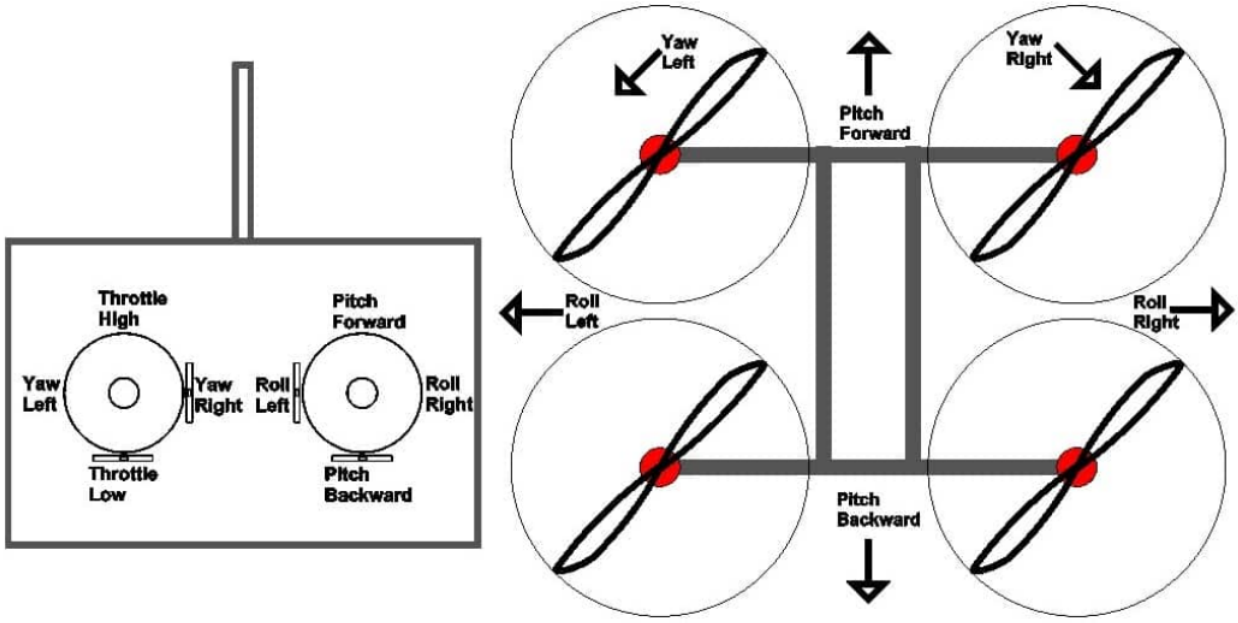

Figure 6 Remote Control Setup for QuadCopter

Flight Modes in Ardupilot

In this section you will fly in the following modes manually:

- Position Mode
- Attitude Mode
- Stabilize Mode
- Acrobatic Mode
- During your simulated flights you will attempt to follow a crossed shape trajectory that is:
    - From the starting position fly foward for 15 meters then fly back to the start
    - Fly 15m to the right then fly back to the start
    - Fly 15m to the left then fly back to the start
    - Fly 15m backwards then fly back to the start
    - Increase altitude by 10 meters

Setting up the Remote Controller

To set up your remote to connect to the Gazebo simulation start up your entire simulation (what you did in Learning Activity 2). Once it is booted up please follow the link to this video to set up your RC controller with SITL https://www.youtube.com/watch?v=2lUASyYU2u0&ab_channel=IntelligentQuads

Learning Activity 5: Autonomous Waypoint Navigation with SITL

Autonomous waypoint navigation for quadcopters is invaluable due to its ability to enhance efficiency, precision, and versatility in a wide array of applications. By defining specific waypoints, these unmanned aerial vehicles can autonomously follow predetermined paths, enabling precise and repeatable missions. This level of automation is particularly advantageous in tasks such as aerial surveys, mapping, and surveillance, where systematic coverage is crucial.

In this practice (not graded) you will learn how to preprogram waypoints so that the Quadcopter will fly those routes. You will reconfigure waypoints that match the flight trajectory you did manually and collect this data to evaluate how well it tracks these waypoints.

How to setup Waypoint Navigation with Mission Planner

In the following link attached follow the tutorial/videos to learn how to up multi-waypoints https://ardupilot.org/copter/docs/common-planning-a-mission-with-waypoints-and-events.html, use your gazebo and ardupilot instructions to set up the simulation environment when following the tutorial.

Assessement Task 1: Conduct your own manual flight test

ECT:3 hrs The objective of this graded task is to give you the oppurtunity to demonstrate your level of competency for SLOs 1,2,3, and 4. For this task, you will conduct your own flight test operation by defining the specific pattern you want to fly manually. It can be any pattern ie, zig-zag, square, rectangle,etc. You must fly each of these patterns in each of the following modes

- Position mode
- Attitude mode
- Stabilize mode

In addition you will set the pitch rate gains to the following values.

- Two times the pitch rate
- Normal pitch rate
- Half the pitch rate

When flying each of these modes. Save the flight data log and generate a table with a qualitative analysis for each of the flight modes you have set with the respective gain you set ie Position mode with two times the pitch rate, Position mode with normal pitch rate, etc. Tabulate your data in a neat fashion as well as showing an image of the simulation with your quadcopter flying at two times the pitch rate for any configuration. Hint you will crash for some, if it crashes end the test and note that and terminate the test.

From these results briefly explain the effect of setting these gains and how it affects the flight performance of the quadcopter.

Assessement Task 2: Conduct your own autonomous flight test

ECT:2 hrs The objective of this graded task is to give you the oppurtunity to demonstrate your level of competency for SLOs 1,2,3, and 5. For this task you will must set up waypoints in the pattern you flew manually in Assignment Task 1. You will then have the drone take off and autonomous navigate throught the waypoints. For each iteration you will set the pitch rate gains to the following values.

- Two times the pitch rate
- Normal pitch rate
- Half the pitch rate

You will tabulate each configuration with a qualitative analysis on how it flew (hint you will crash for some, if it crashes end the test and note that and terminate the test).

An example of what the table should look like

P gain	Response
Two Times the Pitch Rate	Overshoot/Undershoot Long/Short Settle
Normal Pitch Rate Gain	Overshoot/Undershoot Long/Short Settle
Half the Original Pitch Rate Gain	Overshoot/Undershoot Long/Short Settle