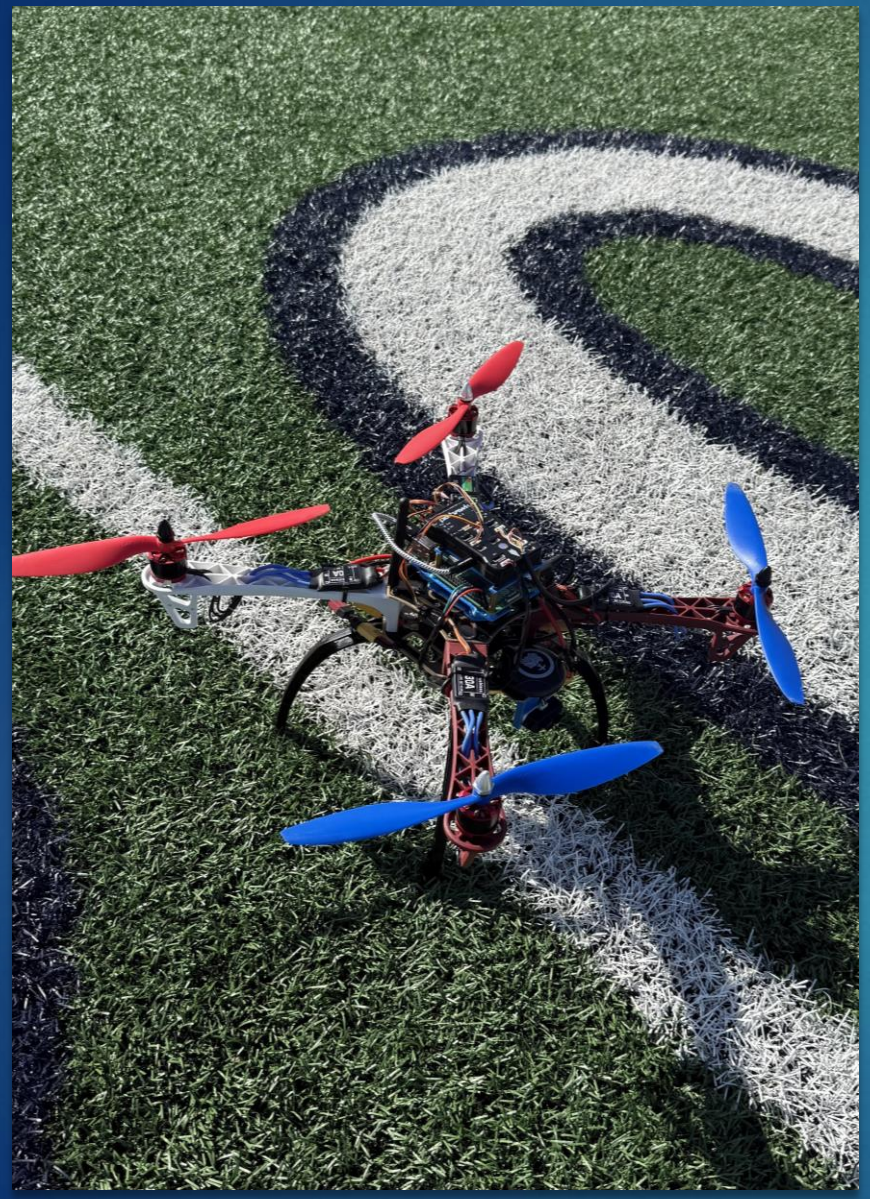


DroneSense

BY: JEREMY GUY, JAYDEN GUY, ANGELINE DOYA, ABBA ANARYU

Problem Statement

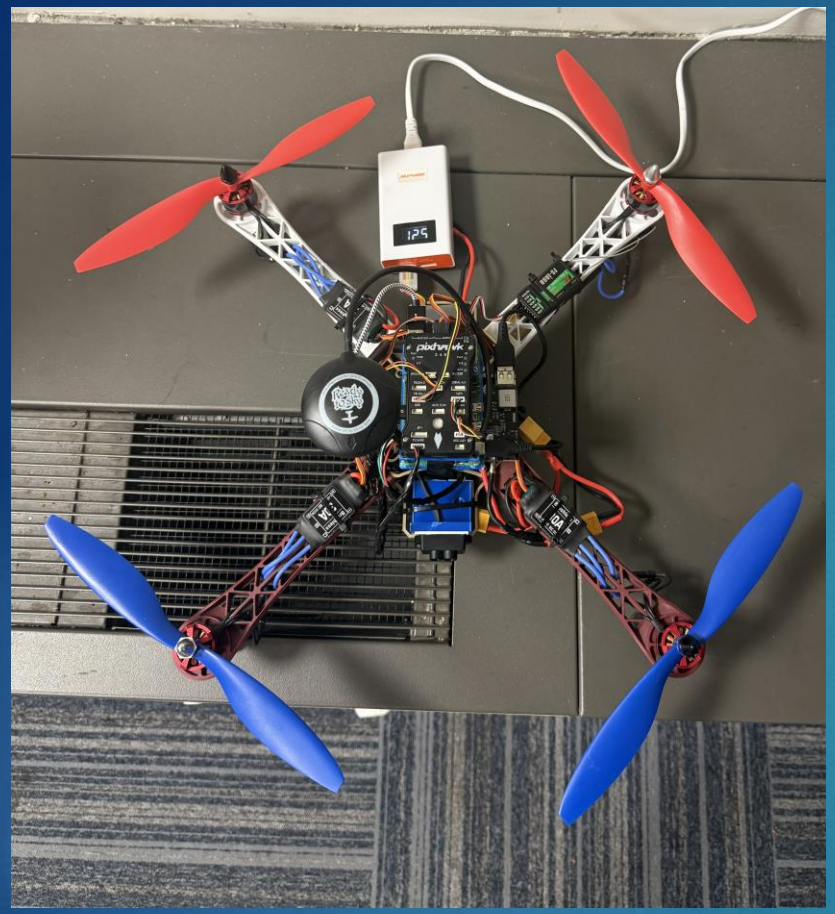
- Controlling a drone requires quick reflexes, especially if pilots are unfamiliar with their surroundings.
- Improper control causes accidents, injury, and property damage.
- Implementing a LiDAR sensor to detect nearby obstacles and send real-time data to the drone, allowing it to stop or reroute to prevent collisions.

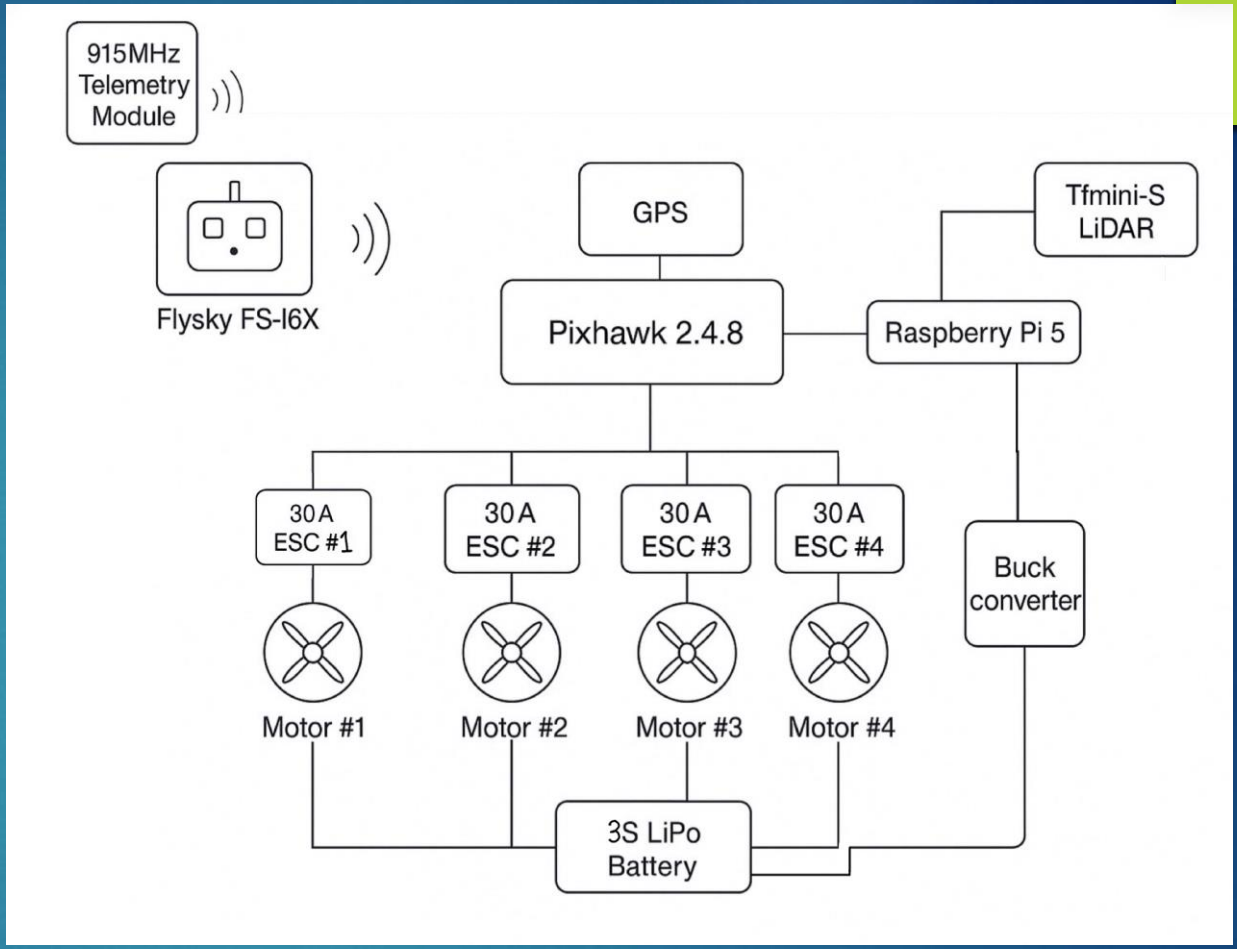
Constraints

- Must comply with **FAA** (flight regulations) and **FCC** (frequency & power) standards
- Operation in various outdoor weather conditions
- High altitude reduces lift, requiring efficient power management
- Privacy concerns with camera use
- Noise regulations in certain areas
- Must identify obstacles and maintain stable flight control in real-time

Design Specifications

- Drone Frame: F450 Frame
- Flight Controller: Pixhawk 2.4.8
- Operating System: Raspberry Pi 5 (Ubuntu 24.04)
- LiDAR Sensor: Tfmini-S LiDAR
- Power System: 3S LiPo 2200mAh 12.6V 50C
- 30A ESCs and brushless motors
- Control System: Flysky FS-I6x transmitter and receiver
- Telemetry: 915MHz 100mW RC Telemetry Kit





Component level design

Sprint #1

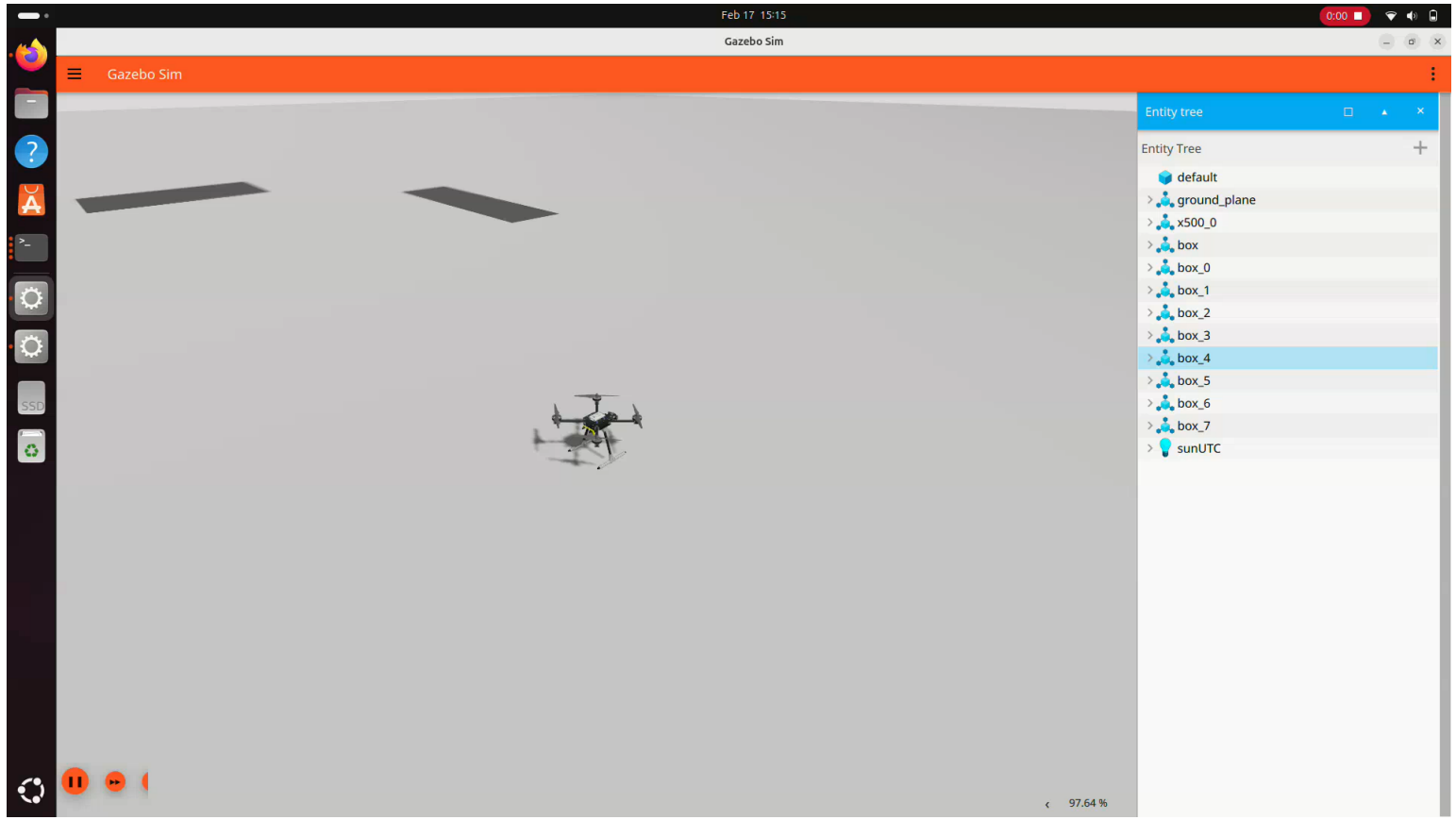
Basic Obstacle Detection algorithm in a simulated environment

Planned Weekly Tasks:

Week 1: Downloaded the necessary software that we need to test the drone in a simulated environment.

Week 2: Set up the drone in the simulated environment and create multiple obstacles in the environment

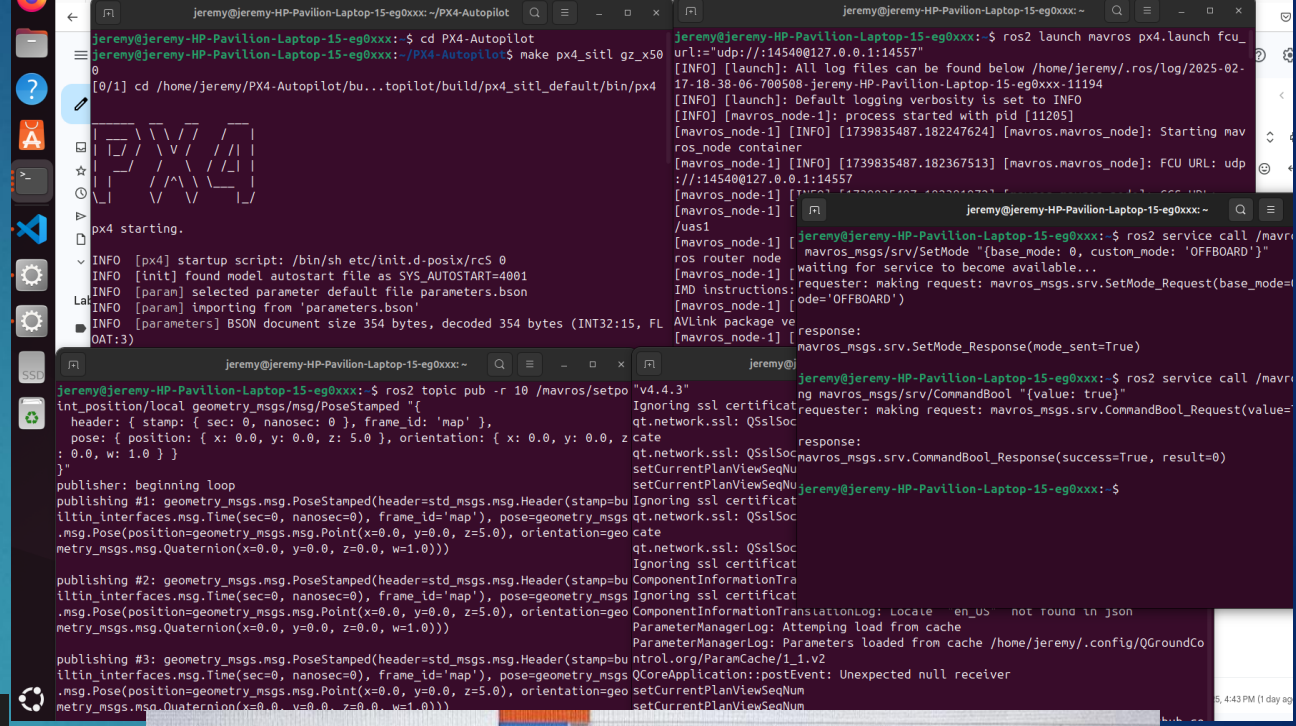
Week 3: Make the drone fly, and integrate a machine learning algorithm for obstacle avoidance

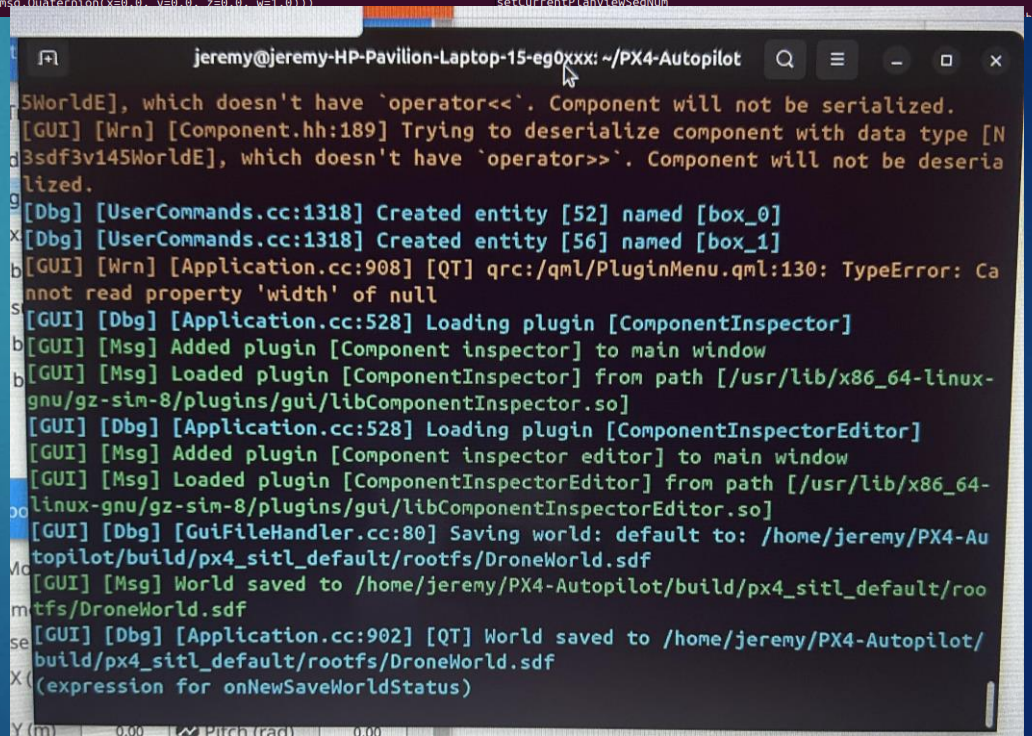


Demonstration

Code

```python
# Safe Distance Threshold (meters)
SAFE_DISTANCE = 2.0

async def get_lidar_distance(drone):
    """ Fetches LIDAR sensor data from PX4. """
    async for distance in drone.telemetry.distance_sensor():
        return distance.current_distance

async def avoid_obstacle(drone):
    """ Main function to check for obstacles and adjust movement. """
    print("Connecting to drone...")
    await drone.connect(system_address="udp://:14540")  # Connect to PX4

    print("Waiting for drone to be ready...")
    async for state in drone.core.connection_state():
        if state.is_connected:
            print("Drone is connected!")
            break

    async for health in drone.telemetry.health():
        if health.is_global_position_ok and health.is_home_position_ok:
            print("Drone is ready for navigation!")
```





Sprint #2

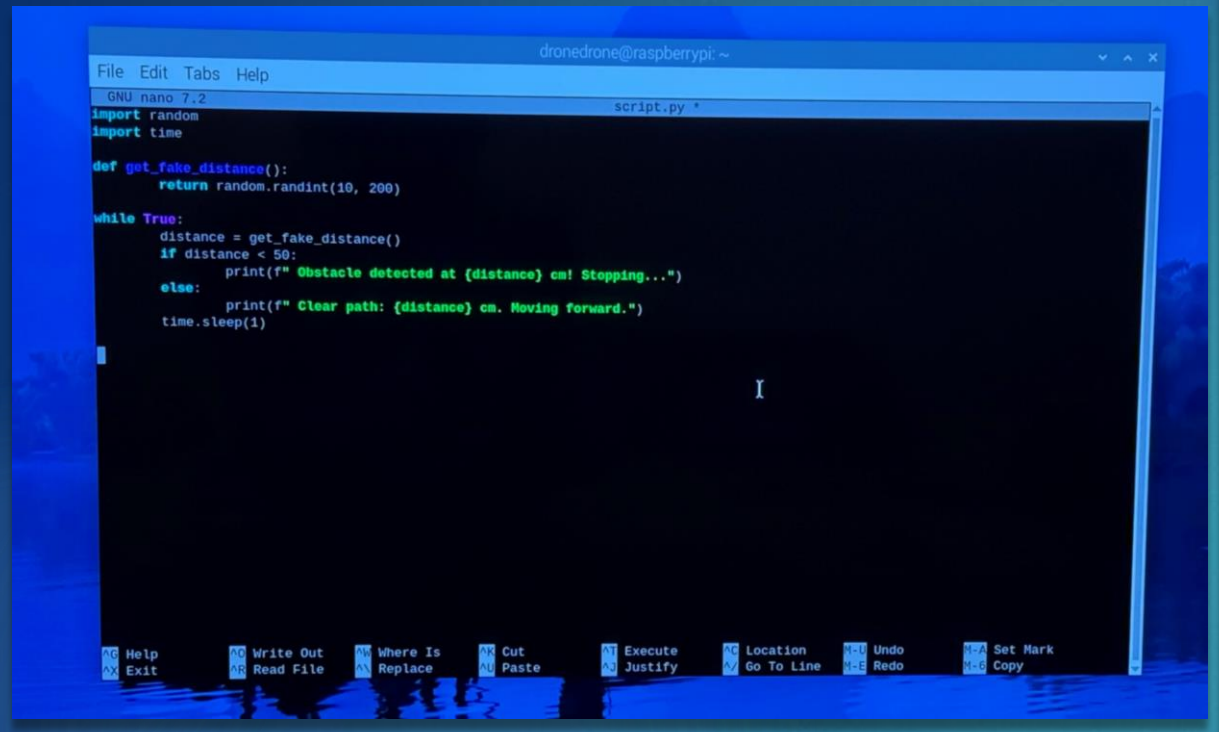
Software integrated with hardware components

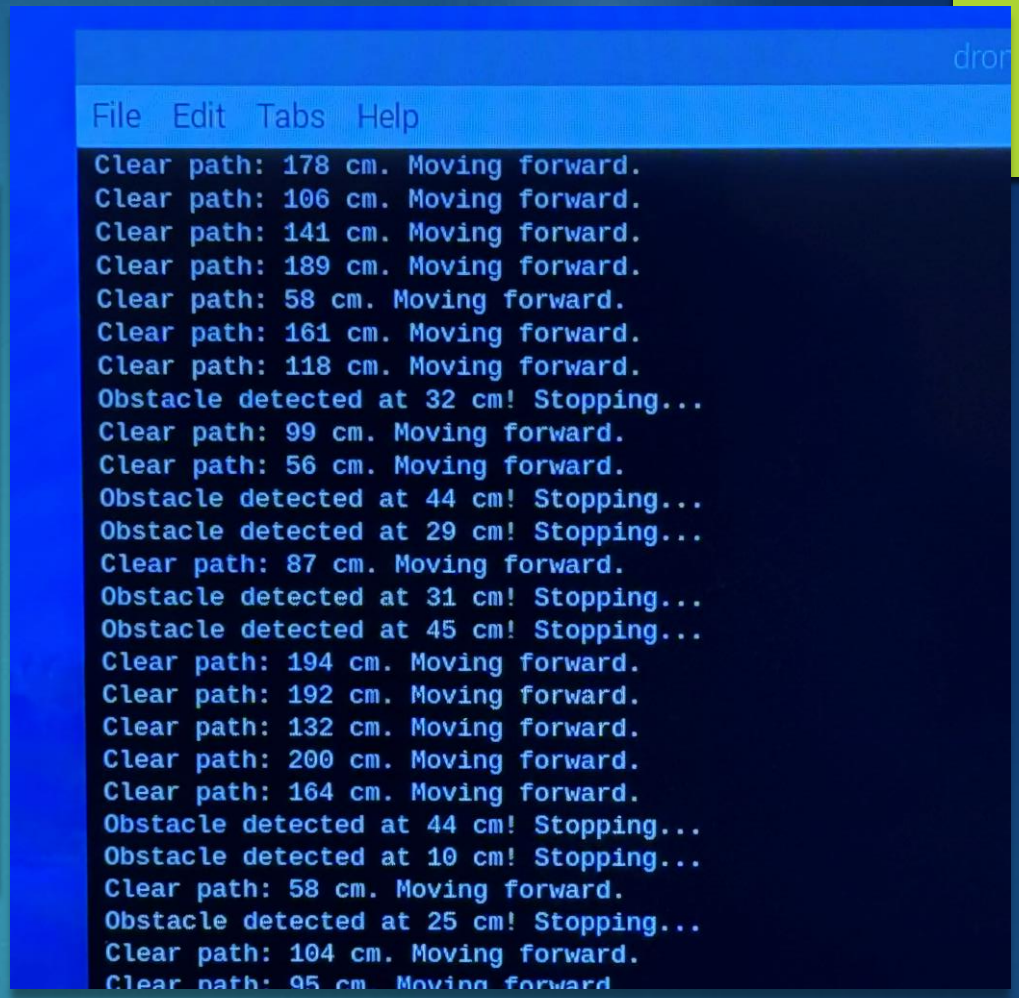
Planned Weekly Tasks:

Week 1: Implemented LIDAR and camera-based object detection

Week 2: Refined machine learning algorithm to improve detection accuracy and avoidance maneuvers

Week 3: Changed the drone to make it fly autonomously and keep working on gazebo environment.





Demonstration

Sprint #3

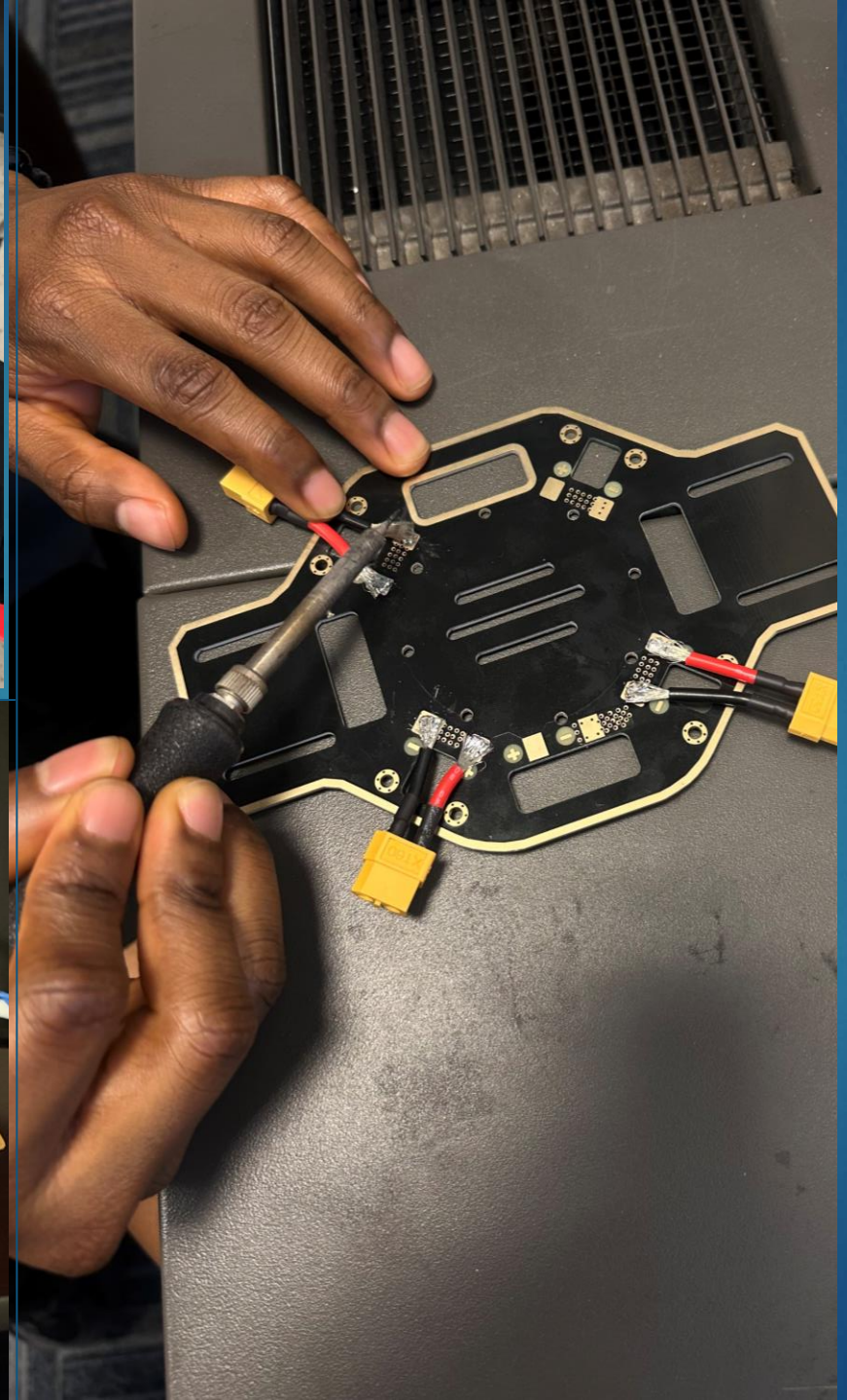
A fully integrated drone capable of navigating objects and real world testing

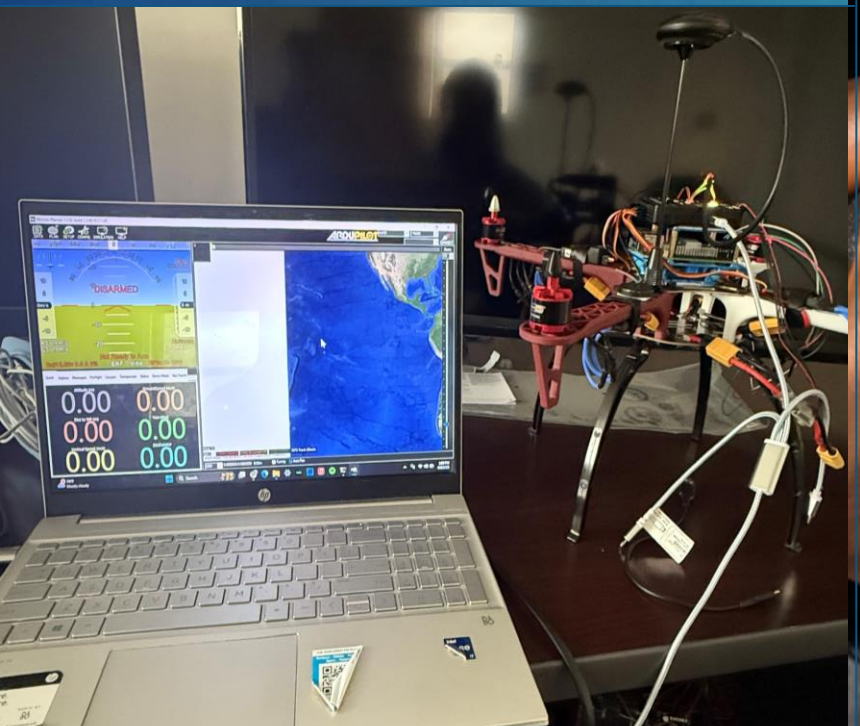
Planned Weekly Tasks:

Week 1: Work on Raspberry Pi and finalize drone simulation

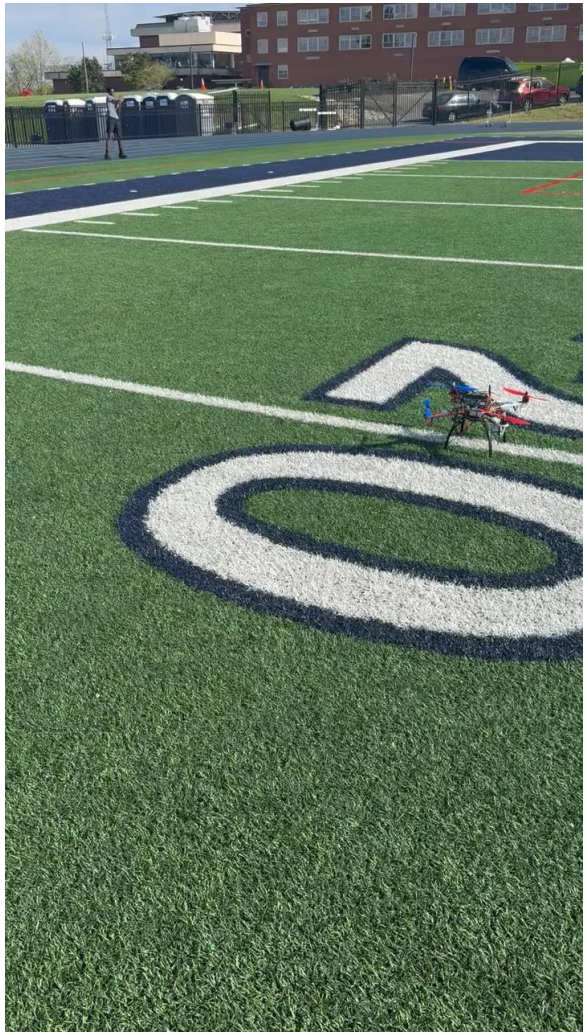
Week 2: Assemble the drone frame and attach hardware (flight controller, battery, transmitter, receiver)

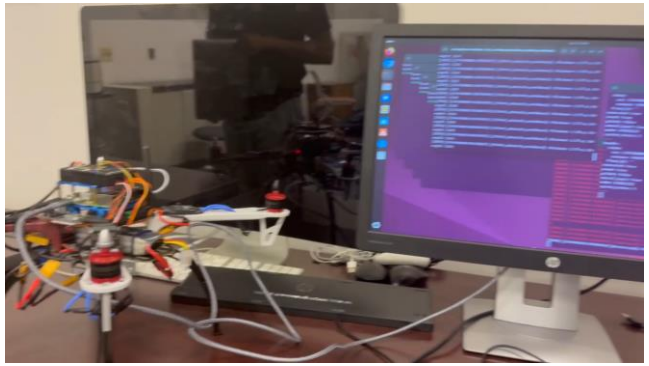
Week 3: Integrate the microcontroller GPS Raspberry Pi and power module to the drone





Demonstration





Demonstration

Final Integrated System

Drone will:

- Successfully navigate through complex environments
- Stable, autonomous operation in real world conditions
- Seamless integration of sensors, machine learning algorithms, and control system