Gesture Operated Drone

Anshul Devnani, CPE
Pranay Jay Patel, CPE
Bernardus Swets, CPE

Group 3

Project Description

- DIY Indoor Drone
- Remotely Controlled
- Operated by Hand Gestures
- User-Friendly GUI to show live drone info

Motivation

- Drones are not easily flown
  - Can cause safety concerns if incorrectly flown
- Remote Control is not directly intuitive, takes practice
- Allows for free hands while controlling the drone
- Puts together concepts learnt throughout our coursework that interests us (Machine Learning, Wireless Comm, Embedded Systems, GUI Development)

Goals and Objectives

- Improve ease-of-use for drones
- Achieve quick response time to gesture commands
- Build GUI that is understandable to the untrained eye
- Maintain stability during drone flight
- Sustain connection to drone at usable distance

Software Requirements

Requirement	Pass/Fail
The drone will convert the signal received over Bluetooth within 500 ms.	✓
The user interface will be able to recognize each of the 8 gestures.	✓
The feedback/reading pane will highlight the correct predicted gesture within 1 second.	✓
The Neural Network will produce an accuracy of a minimum of 95 percent.	✓
The user interface GUI will consist of a webcam pane, log pane, and miscellaneous pane.	✓

Hardware Requirements

Requirements	Pass/Fail
The drone will not weigh more than 2 pounds.	✘
The drone will be powered by 3.7V lithium polymer batteries.	✔
The microcontroller will be powered by a 9v DC battery.	✔
The drone will utilize propellers of 10 inches or smaller.	✔
The drone will utilize 4 electronic speed controllers to help control the propellers.	✔
The drone will utilize 4 brushless motors with KV above 900.	✔
The drone will utilize an ATmega328P microcontroller as a flight controller.	✔

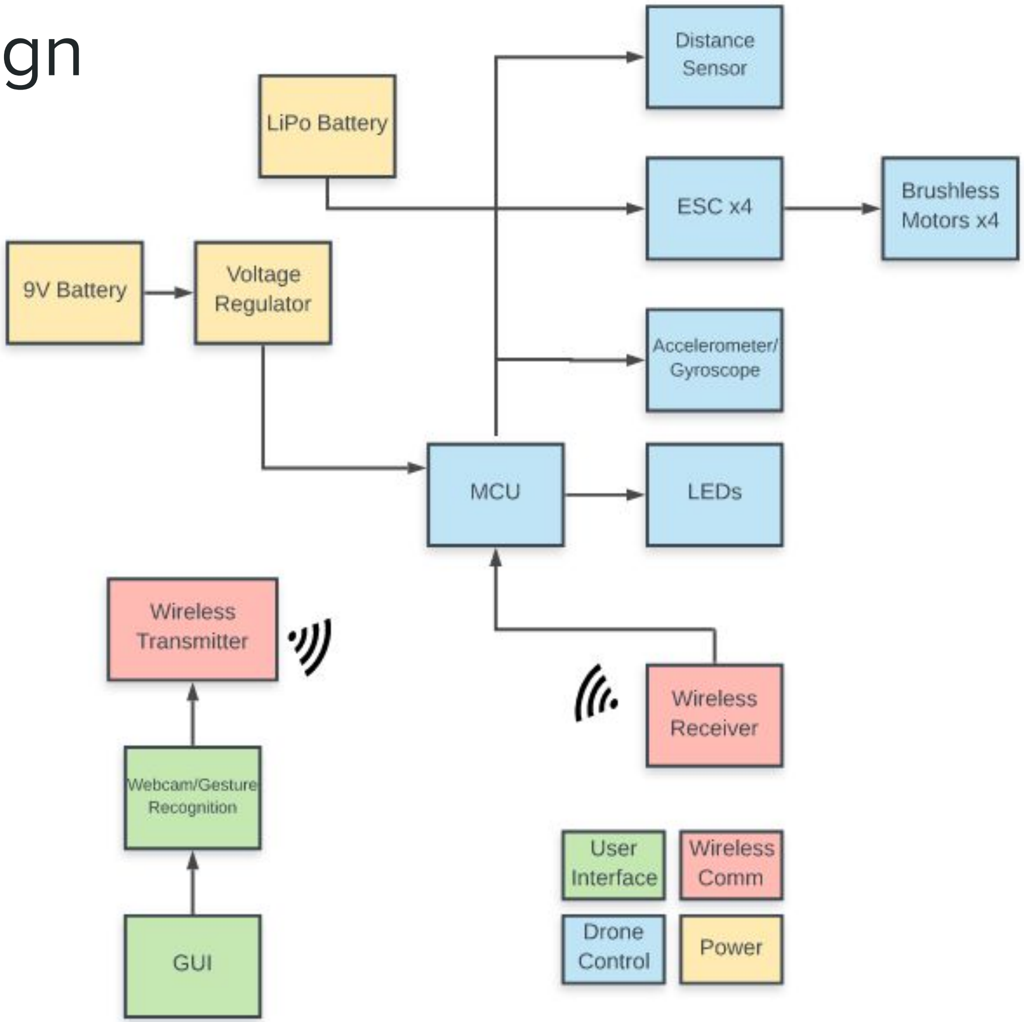
System Requirements

Requirements	Pass/Fail
The drone will be able to receive signals over Bluetooth communication from within a range of 20 feet.	✓
The drone will be able to react to commands within 1 second.	✓
The drone will be able to land, and motors will terminate within 5 seconds.	✗
The drone will be able to take off to 3 feet within 3 seconds.	✗
The drone will maintain its altitude when moving left, right, forwards, and backwards.	✗
When the drone accelerates in a specific direction, it will rotate less than 90 degrees to perform the given action, as to not tip the drone over.	✓

Constraints

Constraint	**Value**
Drone Laws	Flying Outdoors
Wireless range	30 ft Radius
Drone Frame Size	Less than 600mm
Drone Battery Runtime	20 Minutes
Drone Weight	Less than 3 pounds
Number of Gestures	At least 8 gestures, but limited, as similar gestures may be hard to differentiate by webcam
Budget	Affordability

System Design



Gesture Recognition

Gesture Recognition Goals

Two Main Goals:

1. Should be accurate regardless of skin color and background
2. Recognition needs to be done in real time

Ways To Achieve Recognition

Two most popular approaches - our pick Neural Networks

OpenCV

- More complex code, can lead to accuracy errors
- Not easily scalable
- Teaching the computer what to look for

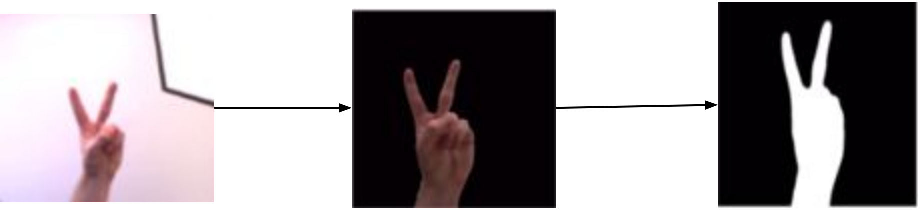
- Minimal code complexity
- Easily scalable
- Computer learns what to look for, let the computer do the hard work

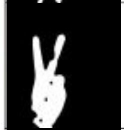
Convolutional Neural Network

- There are many different types of neural networks, choice is based on the problem at hand
- Most popular machine learning technique for image recognition is using a Convolutional Neural Network
- CNN’s use images from a dataset (hand gestures) and learns what makes each subset different from each other (palm vs thumbs up) by extracting features from each image (through convolutional operations) - Known as the training phase
- CNN’s are then tested on images it has not seen to verify how well the model trained - Known as the testing phase

Building the Gesture Recognition Application

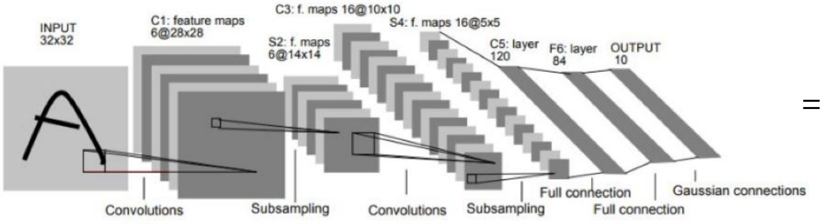
- 1st step is to build our own dataset
- Characteristics of a good dataset
    - Provide good coverage
        - 2000+ images per hand gesture
        - Different angles and focuses
    - Independent of skin color and background
        - Background Subtraction -> Binary Threshold



User Action	Result
No Gesture	Hover in place/autolevel
	Thrust Upwards
	Drone flies forwards
	Drone flies to the left
	Drone flies to the right
	Drone lands in current position
	Drone flies backwards
	Thrust down

Building the Gesture Recognition Application Cont.

- 2nd Step is to build the CNN architecture/model
- Two Options: Create your own* vs. Use existing architecture
- G.O.D Architecture
  - 5 Convolutional Layers - Responsible for learning features of image
  - Shallow - Only 5 layers - Good for real time recognition
- How to code?
  - Use Keras + TensorFlow frameworks to build in Python

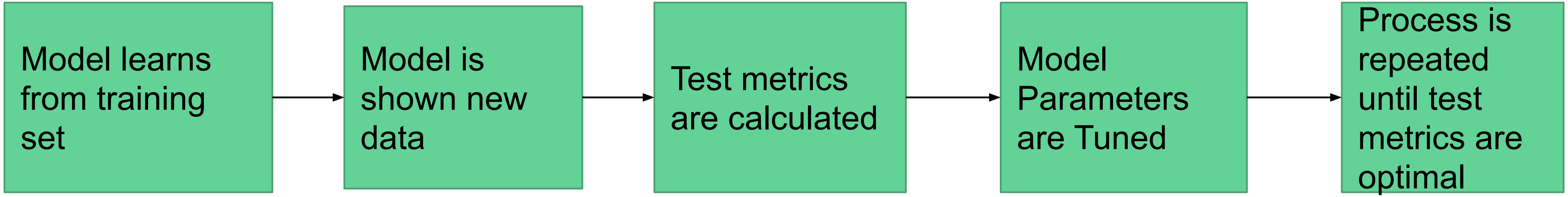


Original Image published in [LeCun et al., 1998]

= 10 lines of code using Keras

Building the Gesture Recognition Application Cont.

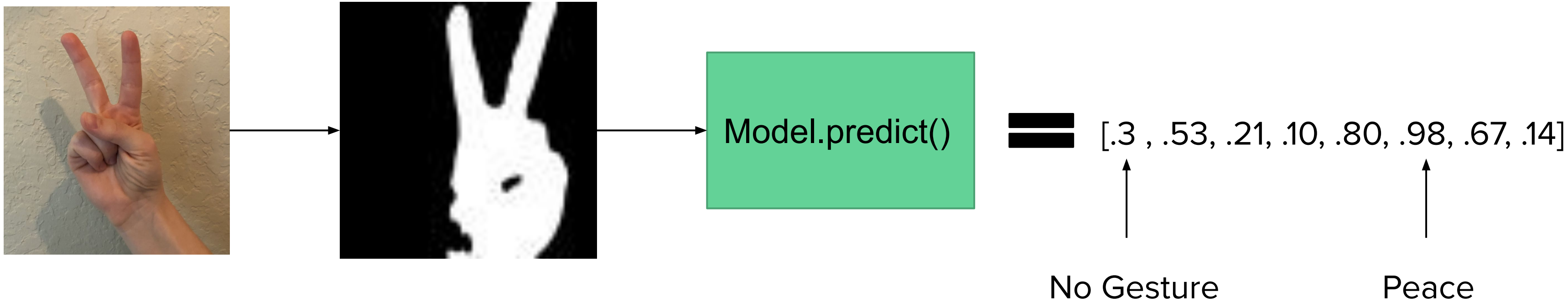
- 3rd step is to train and test the CNN model
- Training is done by splitting the dataset into two subsets. Training set and Validation set
  - 80 - 20 training to validation split



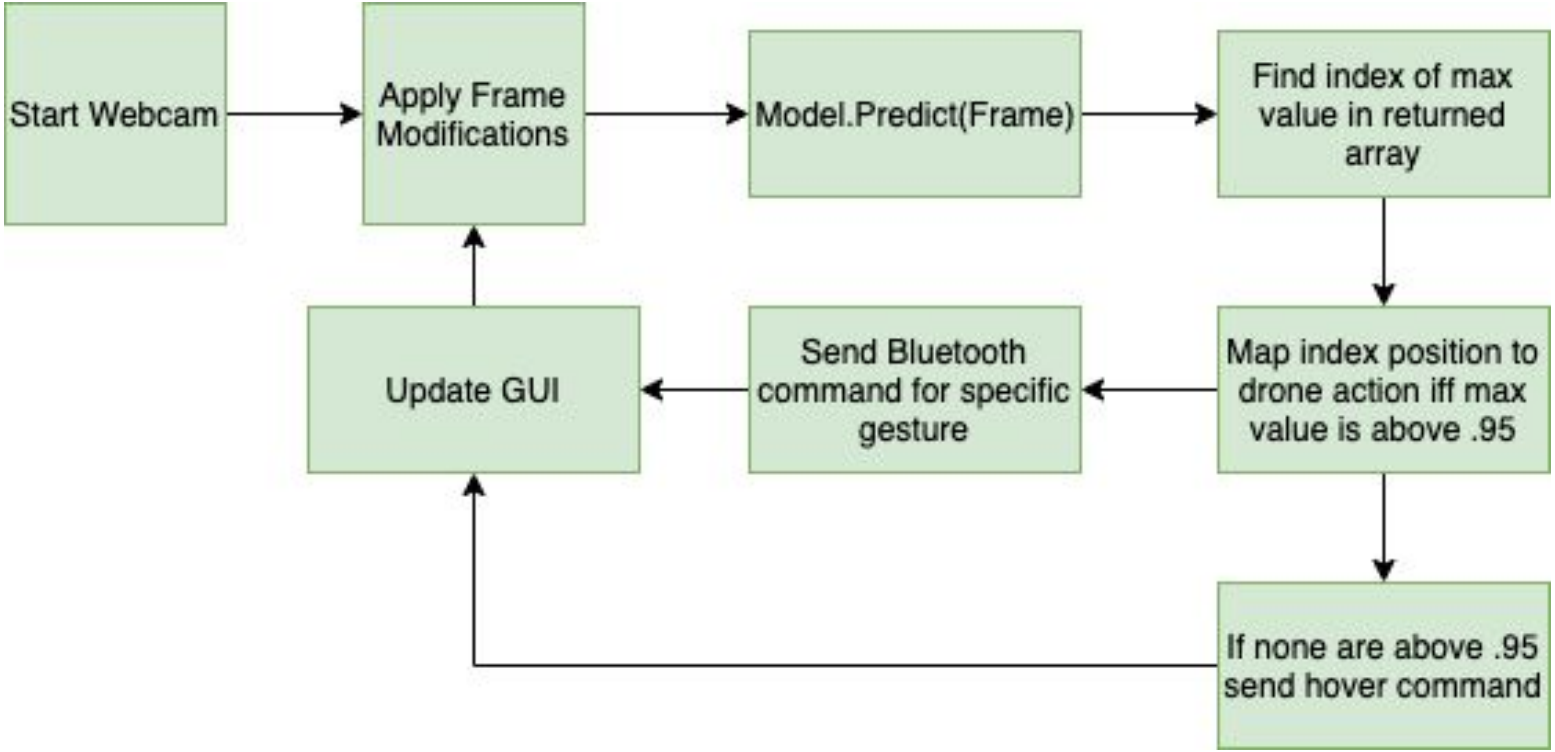
- Model parameters are saved

Building the Gesture Recognition Application Cont.

- Final step is to use the saved model to predict hand gestures in real time



Gesture Recognition Application Flowchart

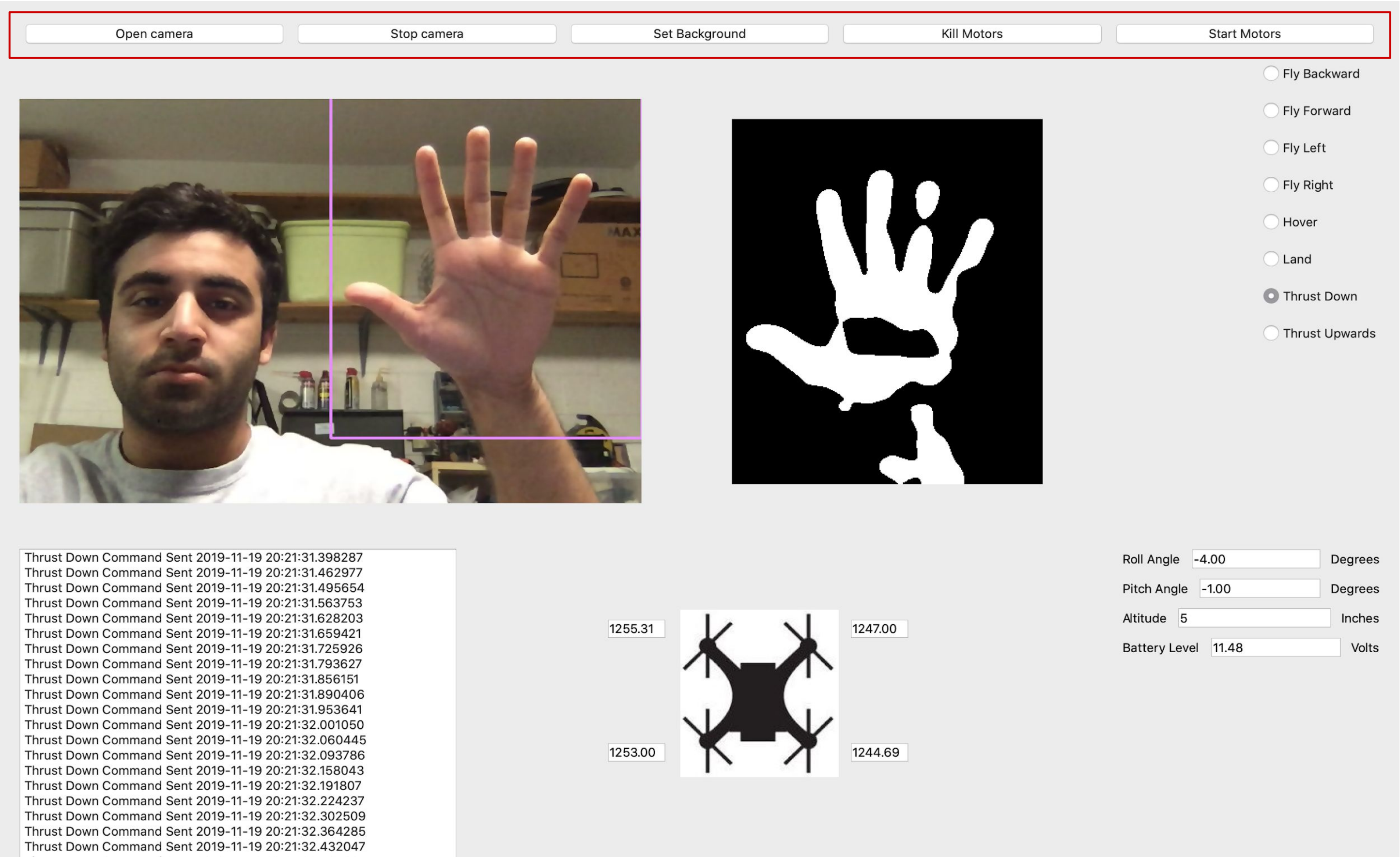


- All done in real time!!!

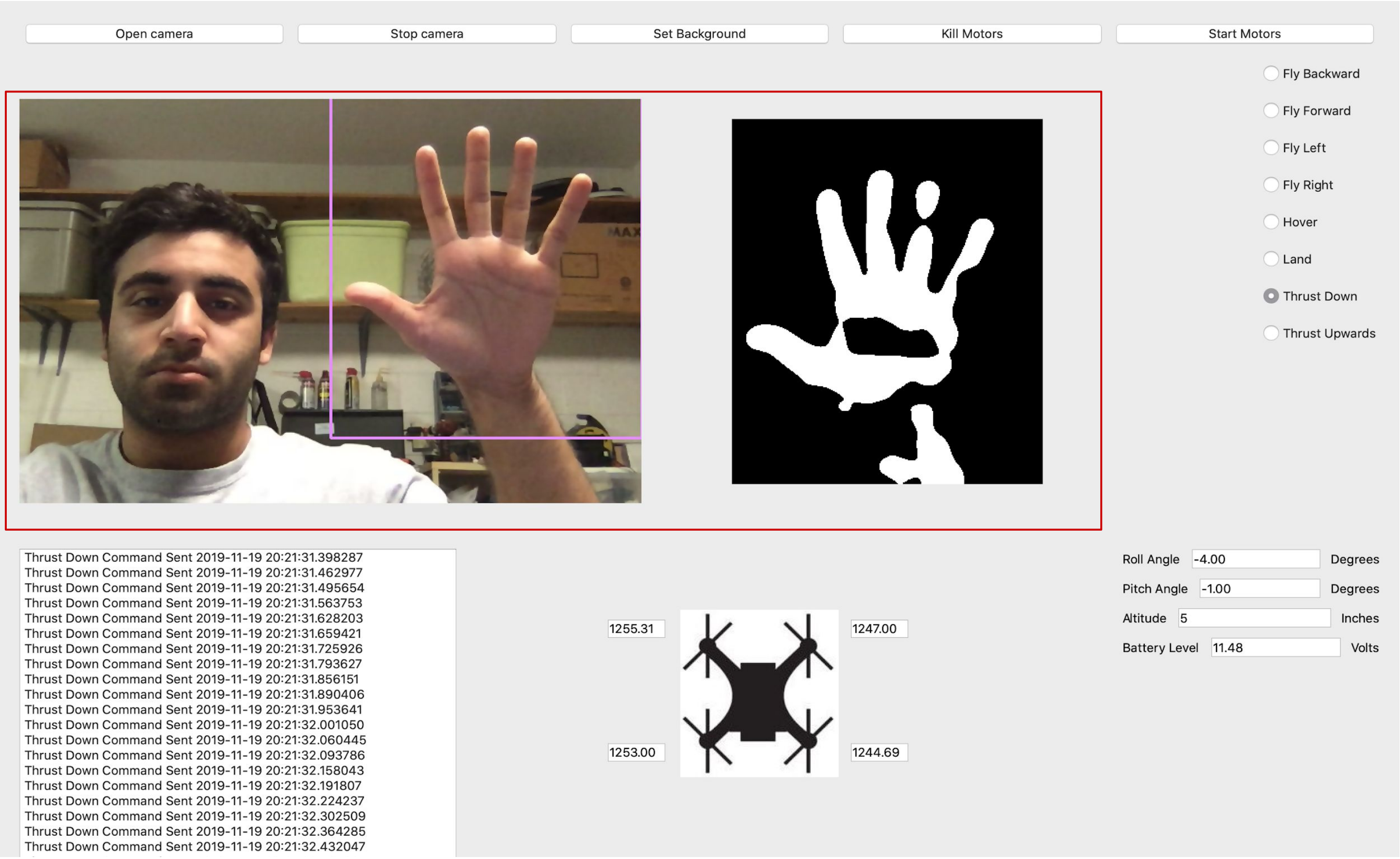
GUI

- Increase organization and usability
- Used PyQT for designing layout

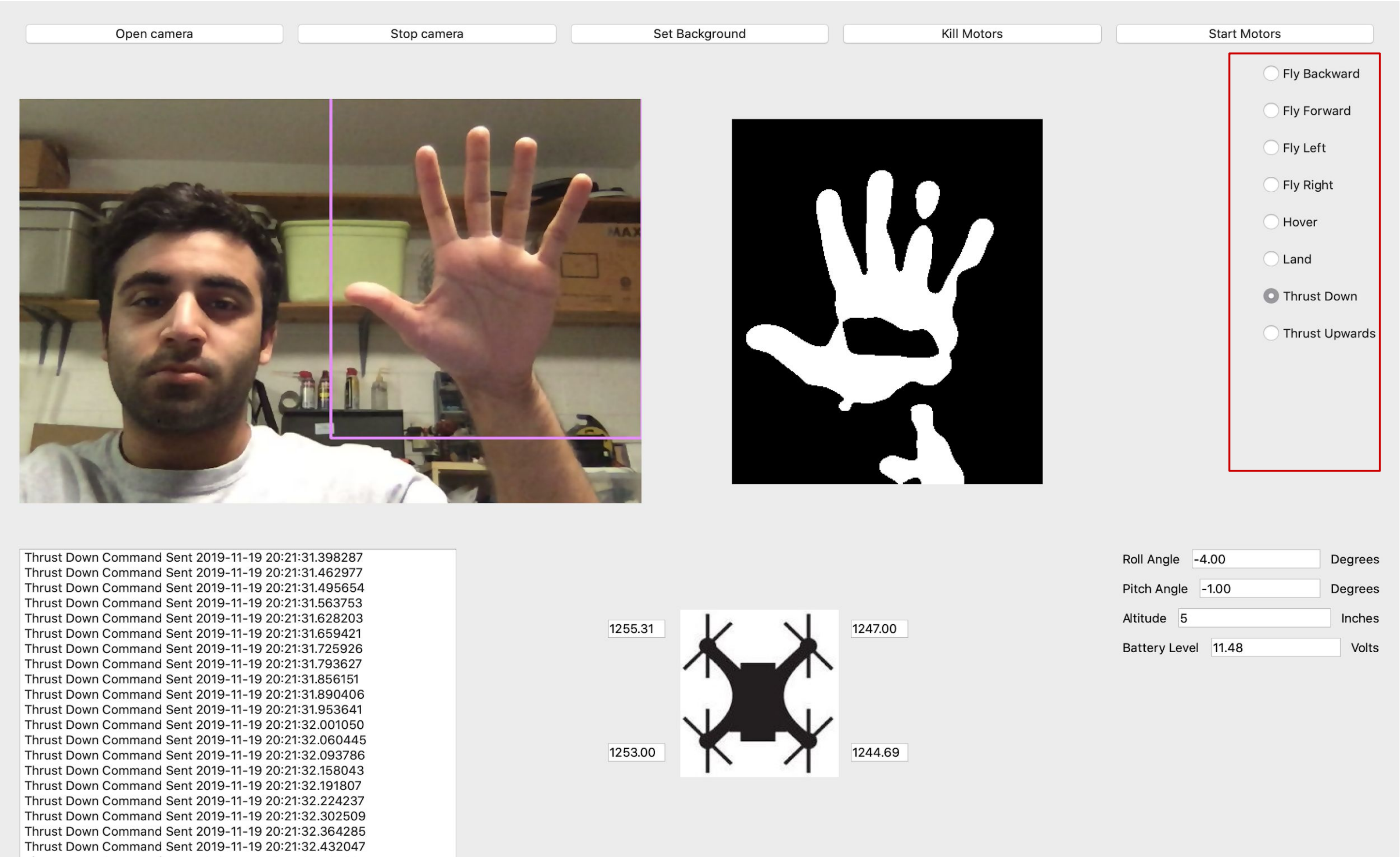
GUI - Utility Buttons



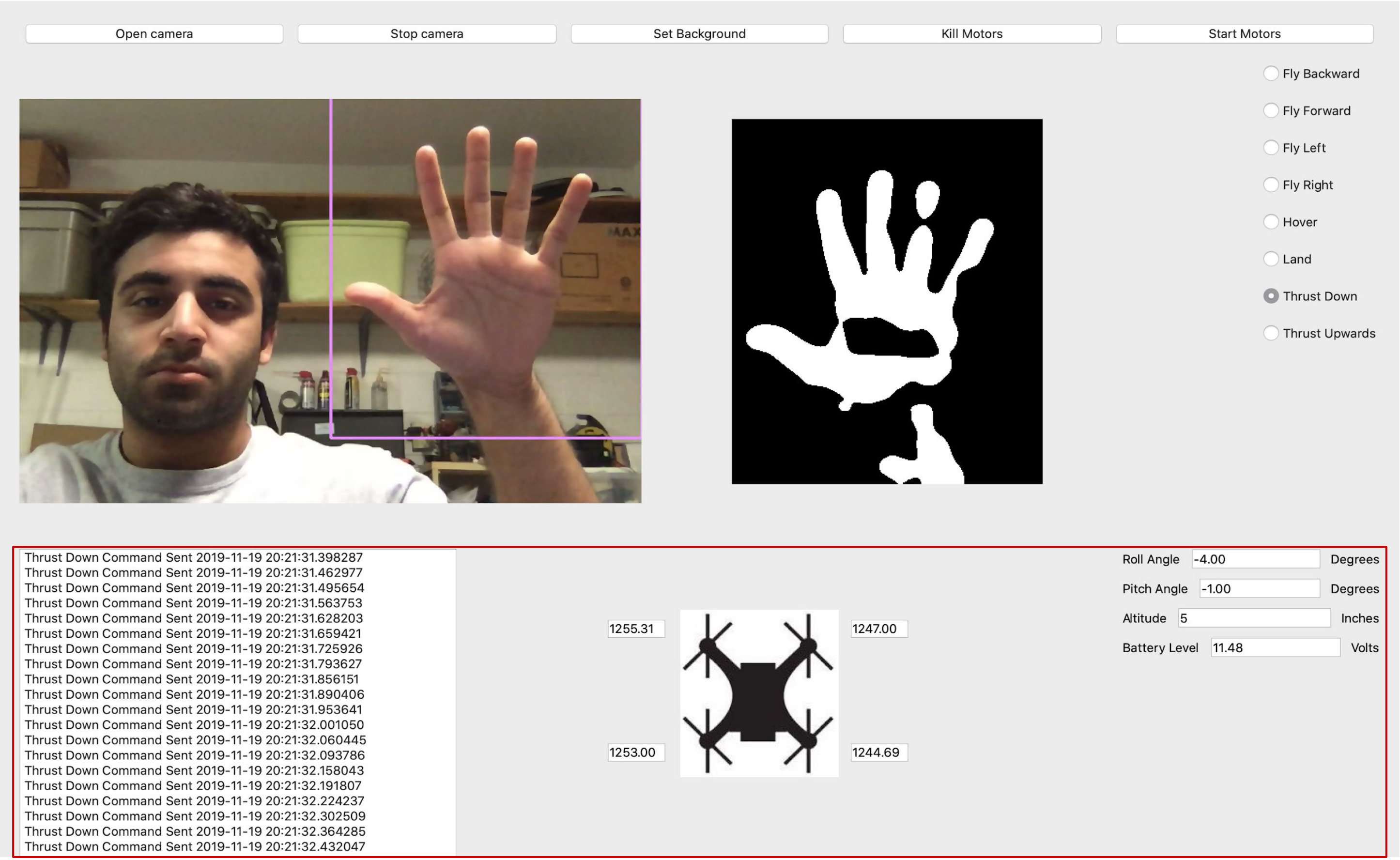
GUI - Camera Feeds



GUI - Model Predictions



GUI - Debug/Drone-Side Data



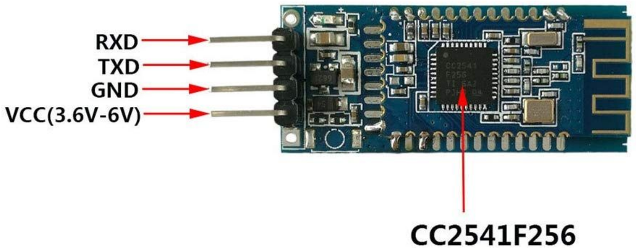
Wireless Communication

Wireless Communication - Overview

- Which type of communication did we use?
- Why did we choose that type?
- What part did we implement for this?
- How was it configured?
- What were its limitations?

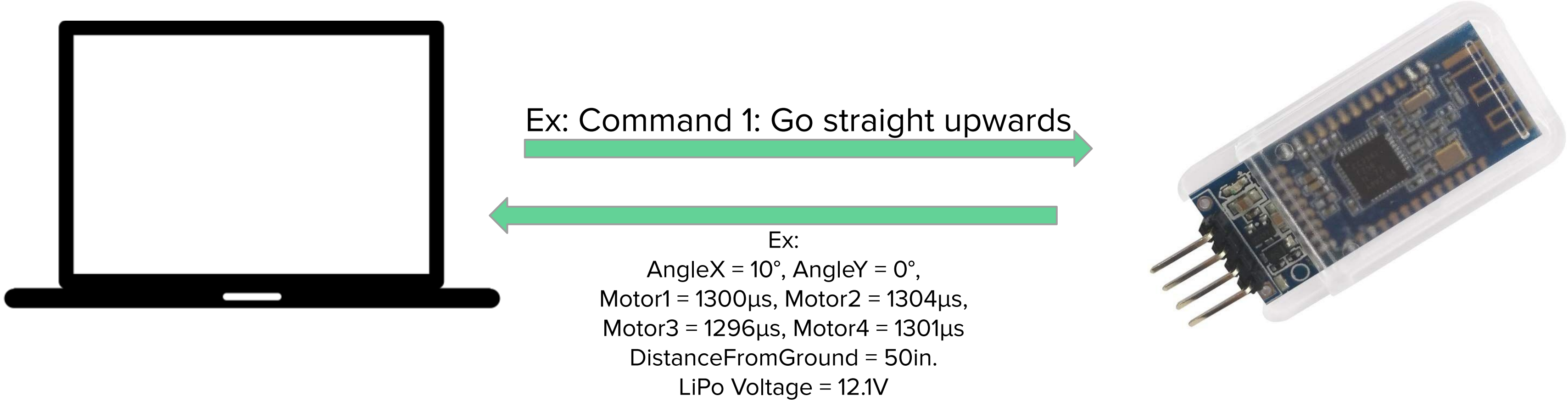
Why Bluetooth?

- IEEE Standard
- Widely used
- User-friendly
- Continually updated
- Backwards Compatibility
- DSDTech HC-05 Bluetooth Module
  - Reason for Choosing: Widely Used, Reliable
  - Cost: $10



Configuration

- Two-way communication over UART
- Bluetooth module serves drone data to MacBook
- MacBook serves gesture commands to Bluetooth module



Limitations - Bluetooth

- Bluetooth is not built for passing large amounts of data rapidly
- Due to this, we opted to make the data communicated minimal
    - Alternative to finding the Bluetooth module that had the fastest data throughput
- We intend to send it as often as we can within reason (high baud rate)
    - Ties to power consumption
- Recall:
    - Objective of project is to sustain connection at usable distance
- Due to this, connection range was of primary concern

Drone Design

Drone Frame

Width	450mm
Height	55mm
Weight	280g
Manufacturer	YoungRC
Price	$18.99

Motors

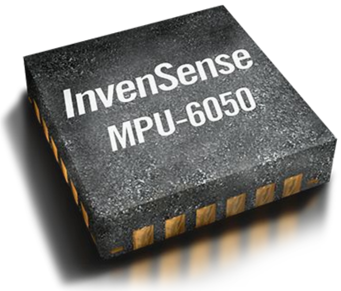
Weight	1.5 oz
Stator Size	2212
Max Efficiency	80%
Current Capacity	12 A
Type	Brushless
KV	1000
Dimensions	45x24x11 mm
Manufacturer	Hobbypower
Price	$40.00

ESC

Weight	4.5 oz
Amp Rating	30A
Dimensions	2.1x1.0x0.5 in
Manufacturer	Hobbypower
Price	$16.00

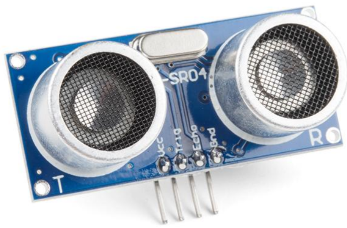
Gyroscope (MPU-6050)

Operating Voltage	3.3V
Comm. Mode	I2C
Axes	6-axis
Manufacturer	InvenSense
Price	$1.95



Distance Sensor (HC-SR04)

Operating Voltage	5V
Operating Current	15 mA
Measure Angle	15 Degrees
Max Range	4 meters
Manufacturer	ElecFreaks
Price	$3.95



LiPo Battery

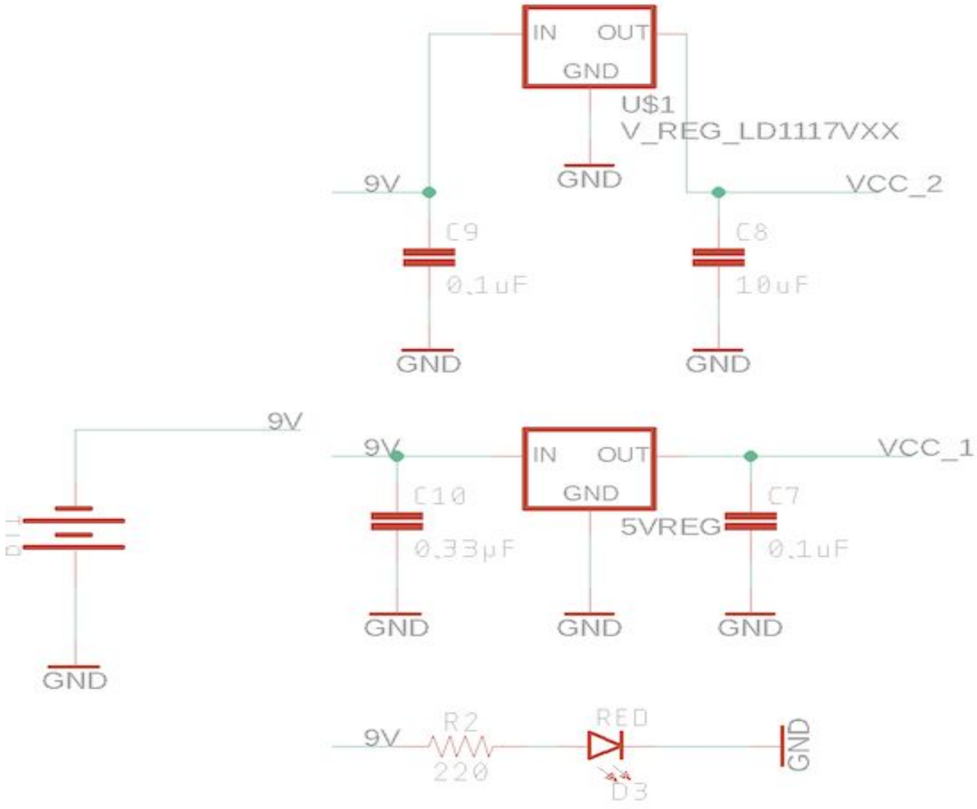
Voltage	11.1V
Cells	3S
Discharge Rate	30C
Size	105x34x24mm
Weight	467 g
Manufacturer	YoWoo
Price	$21.99

Flight Controller

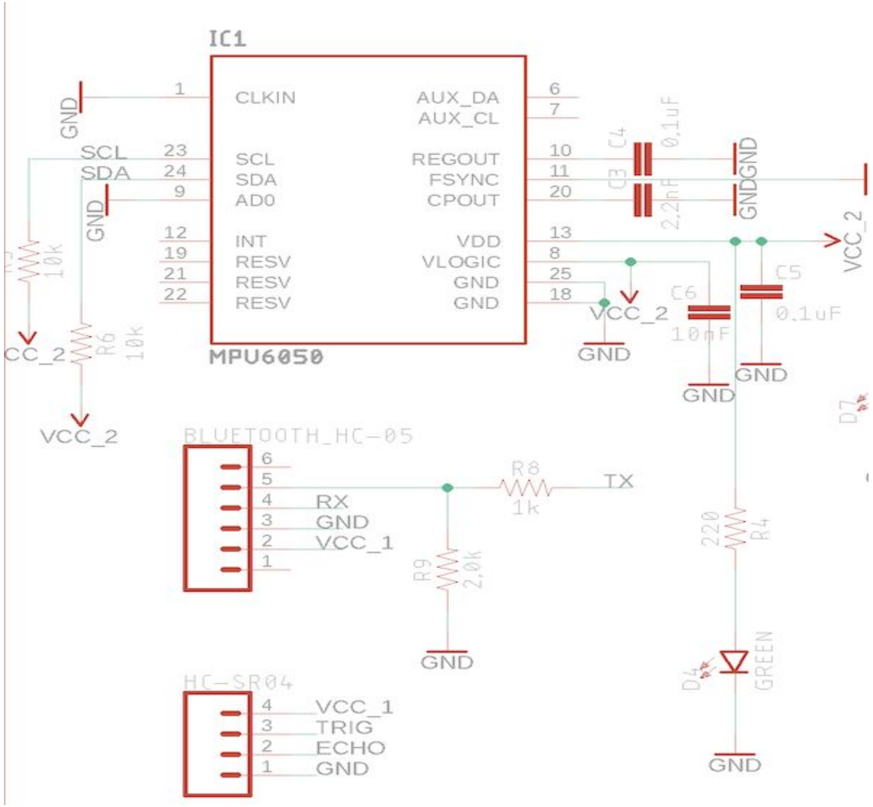
- We are designing our own flight controller through the Atmega328p microcontroller. Can be tested using the Arduino Uno board
- Microcontroller must be able to handle the workload

Clock Rate	16 MHz
RAM	2 KB
Flash	32 KB
Voltage Range	1.8-5.5 V
Price	$2
Manufacturer	Atmel

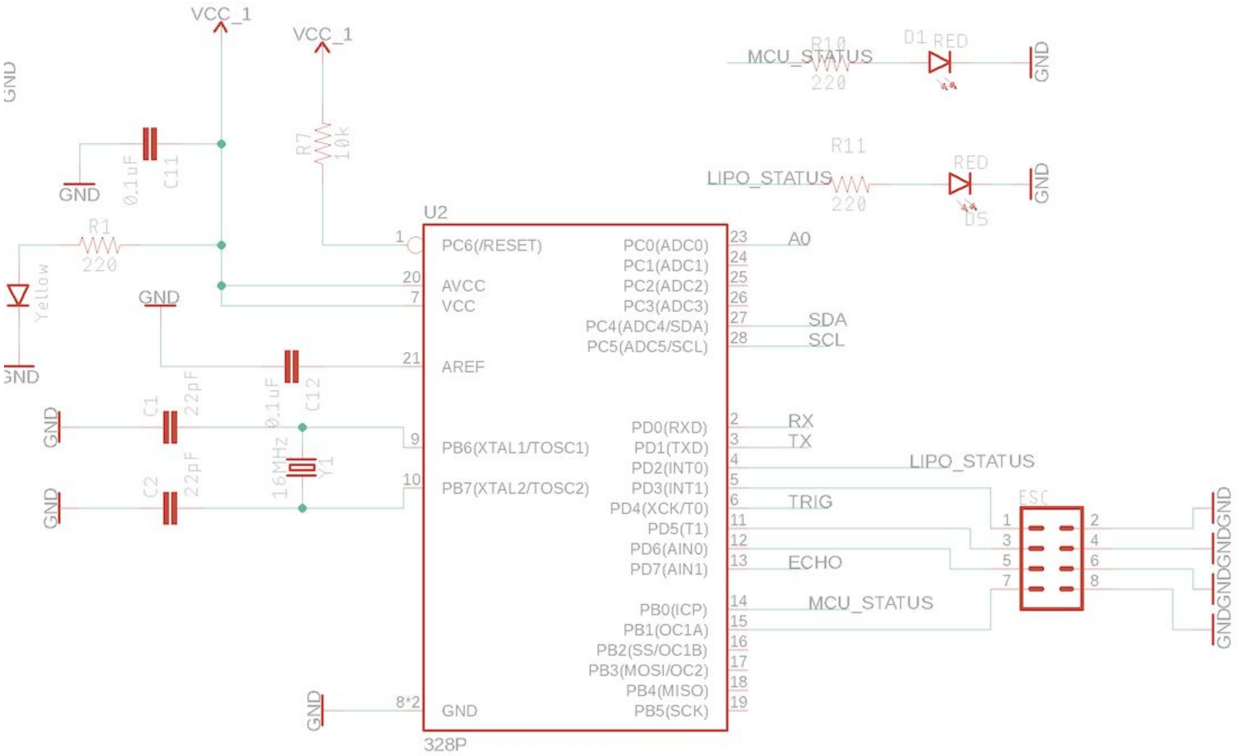
Schematic Design - Power



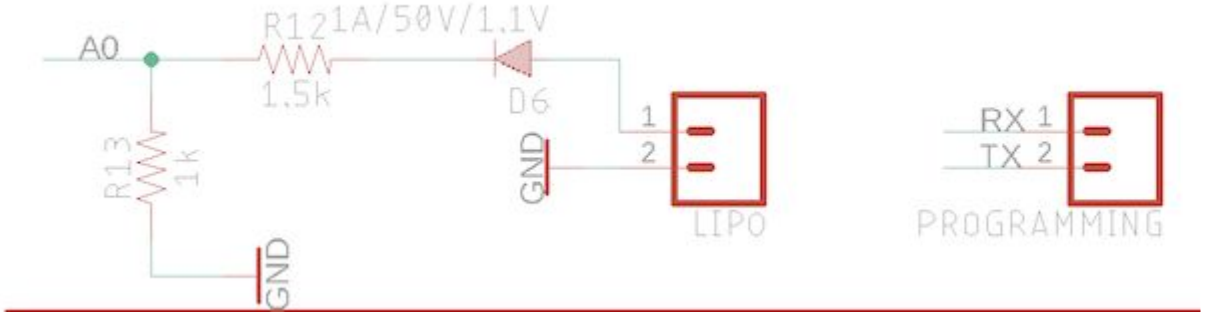
Schematic Design - Sensors



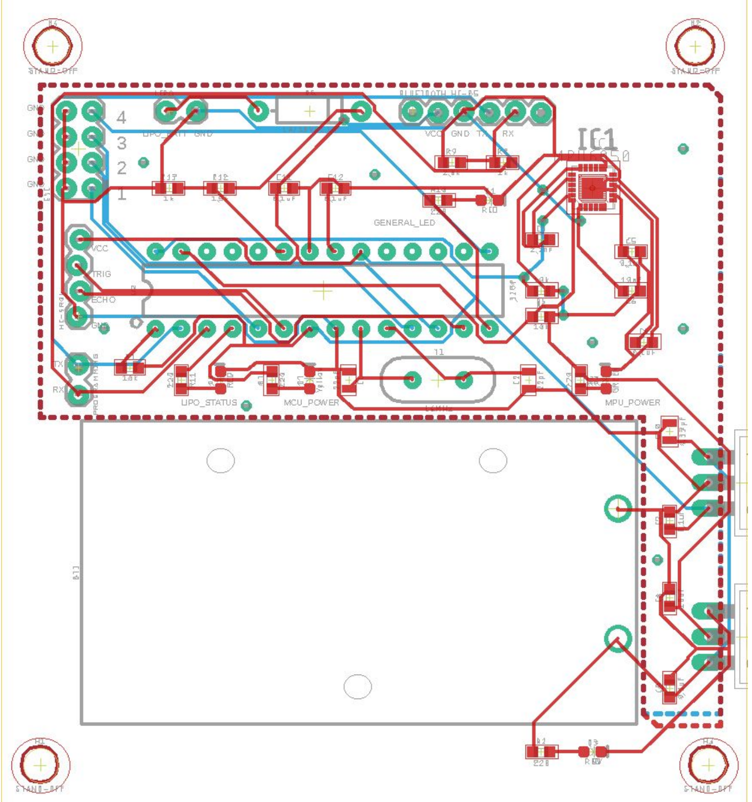
Schematic Design - MCU/Flight Controller



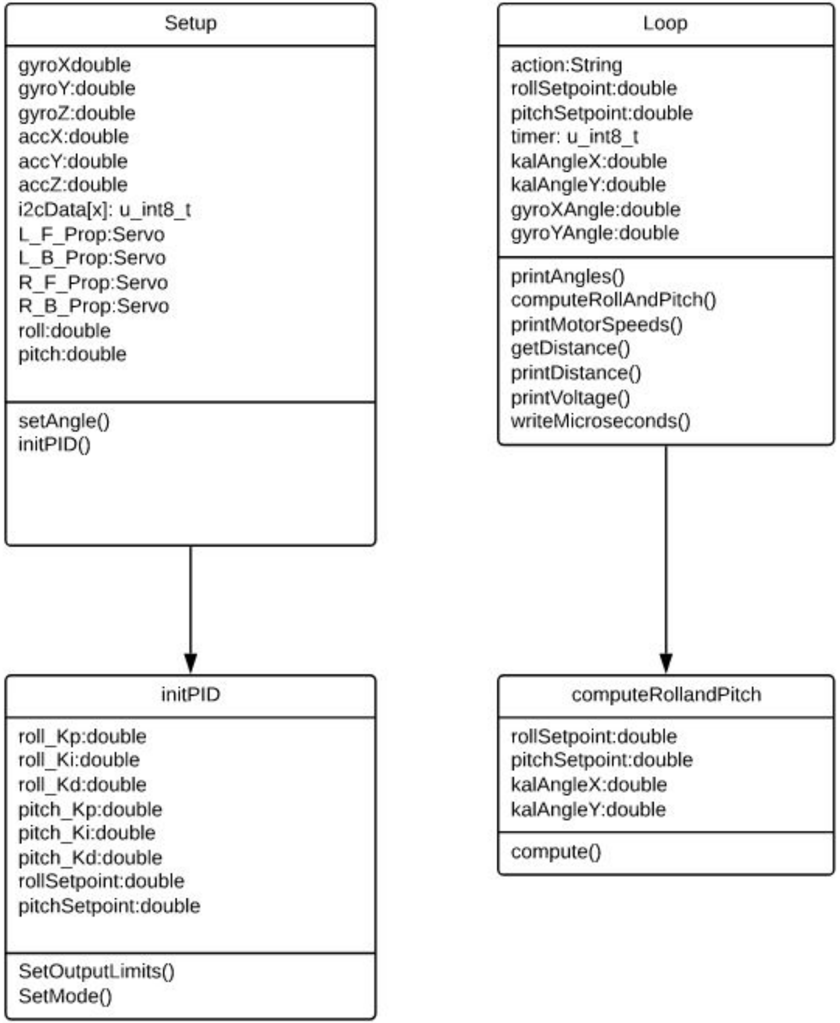
Schematic Design - Miscellaneous



PCB Routing



BLUETOOTH HC-05
VCC
GND
TX
RX
IC1
GENERAL_LED
VCC
TRIG
ECHO
GND
TX
RX
LIPO_STATUS
MCU_POWER
MPU_POWER
4
3
2
1



Setup
gyroXdouble
gyroY:double
gyroZ:double
accX:double
accY:double
accZ:double
i2cData[x]: u_int8_t
L_F_Prop:Servo
L_B_Prop:Servo
R_F_Prop:Servo
R_B_Prop:Servo
roll:double
pitch:double
setAngle()
initPID()
Loop
action:String
rollSetpoint:double
pitchSetpoint:double
timer: u_int8_t
kalAngleX:double
kalAngleY:double
gyroXAngle:double
gyroYAngle:double
printAngles()
computeRollAndPitch()
printMotorSpeeds()
getDistance()
printDistance()
printVoltage()
writeMicroseconds()
initPID
roll_Kp:double
roll_Ki:double
roll_Kd:double
pitch_Kp:double
pitch_Ki:double
pitch_Kd:double
rollSetpoint:double
pitchSetpoint:double
SetOutputLimits()
SetMode()
computeRollandPitch
rollSetpoint:double
pitchSetpoint:double
kalAngleX:double
kalAngleY:double
compute()

Testing Plan

- Initial GUI Testing
  - Test the accuracy of each hand signals
  - Improve model, and change similar hand signals if necessary
- PID Tuning
  - Isolate each axis
  - Establish Kp, Ki, and Kd value for pitch and roll
  - Combine axises and adjust accordingly
- Adjust filter values to reduce noise and drift
  - Q-angle, Q-bias, and R measure
- Integrate GUI and Drone
  - Ensure commands are accurately interpreted and quickly executed

PID Tuning

- Set all values to 0
- Increase P until drone oscillates around the setpoint (desired angle) steadily
- Increase D until those oscillations are minimal
- Repeatedly tweak P and D values to find maximum stability
- Increase I so that the drone makes a swifter correction, to the point where you are okay with oscillations

Bumps In The Road

- Faulty/Broken Components
- Gyroscope Drift
- **Noisy Gyroscope***
- Arduino Bootloader issue*
- Threading in PyQt

LESSONS LEARNED

*Unable to fix

Attempted Solutions for Noisy Gyroscope

Reducing Vibrations

- Foam mounted flight controller
  - Anti-vibration foam, eliminate any metal in contact with the flight controller
- Silicone TPU soft mounts for the motors
- Securely fastened drone frame
- Center the gyroscope to minimize vibrations

Software Solutions

- Utilizing MPU6050 internal LPF
- Filter Algorithms
  - Complementary Filter
  - Kalman Filter
- Decrease gyroscope sensitivity

Complementary Filter vs. Kalman Filter

- Complementary Filter
  - Low processing cost
  - Easy to implement
  - Improved gyroscope accuracy
  - Decreased the drones drift
- Kalman Filter
  - More complex
  - HIgher processing cost
  - Handles the drones vibrations much better
  - Eliminated the gyroscopes drift
  - More customizable

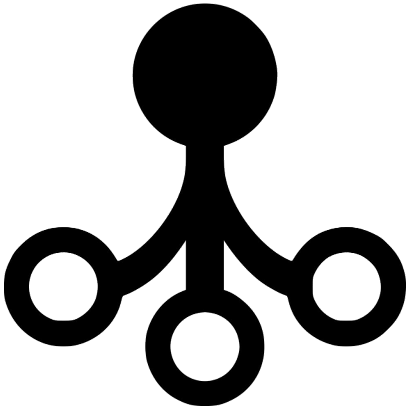
Administrative Content

Proposed Budget

Component	Estimated Cost	Actual Cost
Development Equipment	$100.00	$50.00
Bluetooth Module	$10.00	$5.00
Motors, ESCs, Propellers	$210.00	$115.00
Drone Frame	$20.00	$17.00
Voltage Regulator	$5.00	$2.00
Batteries	$40.00	$80.00
Sensors	$40.00	$15.00
PCB Printing	$30.00	$85.00
Miscellaneous Components	$200.00	$200.00
Total:	**$655.00**	**$569.00**

Work Distribution

	User Interface	Drone Design	Power	Computer Vision	Wireless Comm.	Flight Controller
Anshul	✓			✓	✓	✓
Pranay	✓		✓		✓	✓
Bernardus		✓	✓		✓	✓



Future Improvements - G.O.D V2.0

- Refocus drone design for dampening vibrations (Carbon fiber drone frame)
- Implement better motors for the drone in order to reduce vibrations
- Implement more accurate gyroscope, test more thoroughly (MPU-9250)
- Fine-tune Kalman filter for reducing all noise, without introducing drift
- PCB Improvements - Remove Battery Holder, On-Board Bluetooth module, etc.
- Dedicated Flight Controller

Questions?