



D.E.A.R Drone
Group B:

Dan Biller, CPE

Ellie Lane, CPE

Austin Perkins, EE

Raymond Chenoweth, EE



UCF

Project Description



- D.E.A.R Drone is an autonomous delivery drone that will deliver a package of $\frac{1}{2}$ pounds.
- The D.E.A.R drone will use sensors such, as GPS and a camera to ensure the package makes it to the destination safely.
- Our mission is to help streamline supply chains to get people their products faster

Project Pros/Cons



Pros

- Fast Delivery
- More Efficient than Truck Delivery
- No human delivery person
- Fully electric
- Cheaper
- Can deliver to remote or inaccessible areas
- Multi-rotors are more maneuverable than fixed-wing aircraft

Cons

- Short range
- Single delivery per trip
- Low payload capacity
- Multi-rotors are less efficient than fixed wing aircraft
- Multi-rotors are slower than fixed wing aircraft

Specs/Requirements



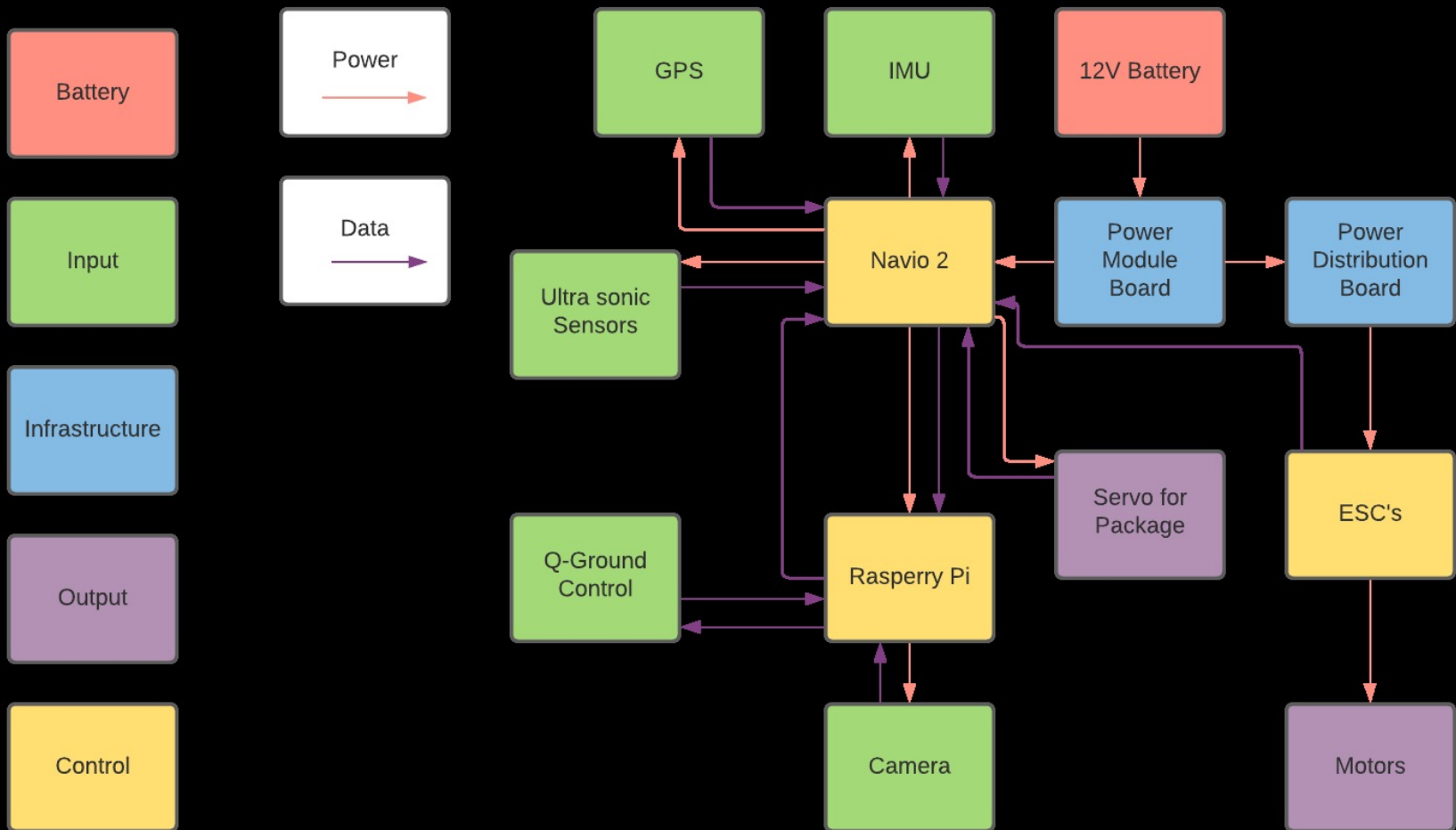
Specifications	
Payload Weight	.5 Pounds
Battery Capacity	6000mAH
Flight time	20min(+/-10 vertical)
Field of view	60 degree horizontal
Range of view	.5-10feet
Measuring Frequency	10Hz
Autopilot	GPS

Standards

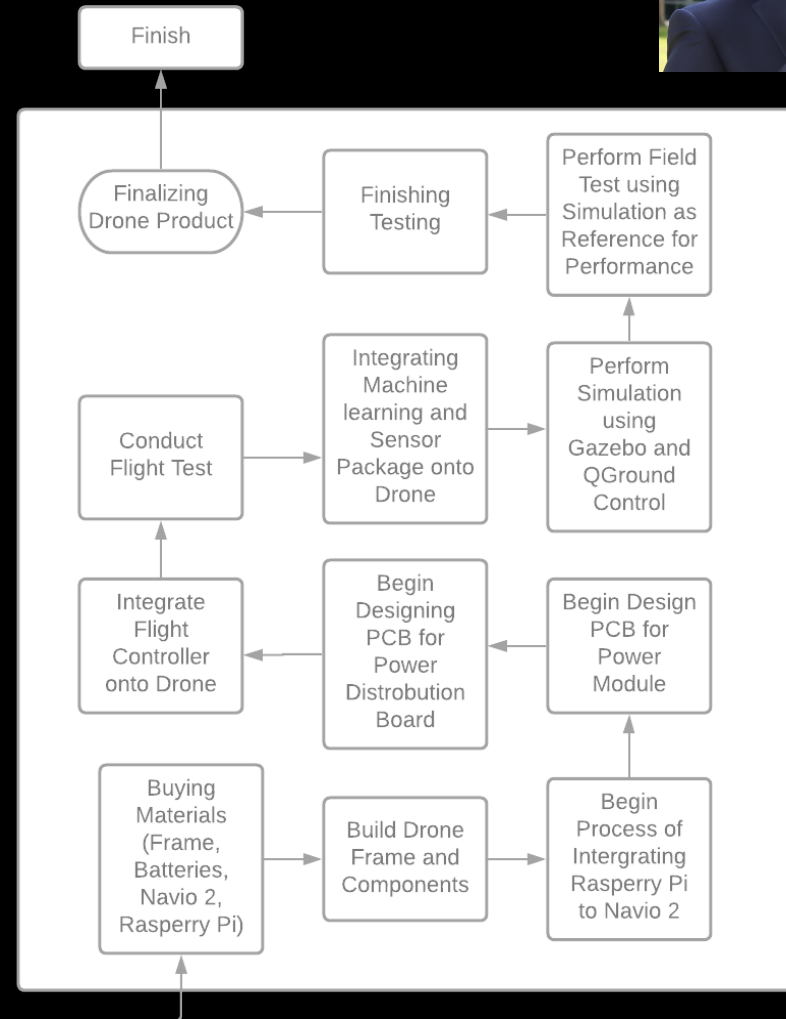
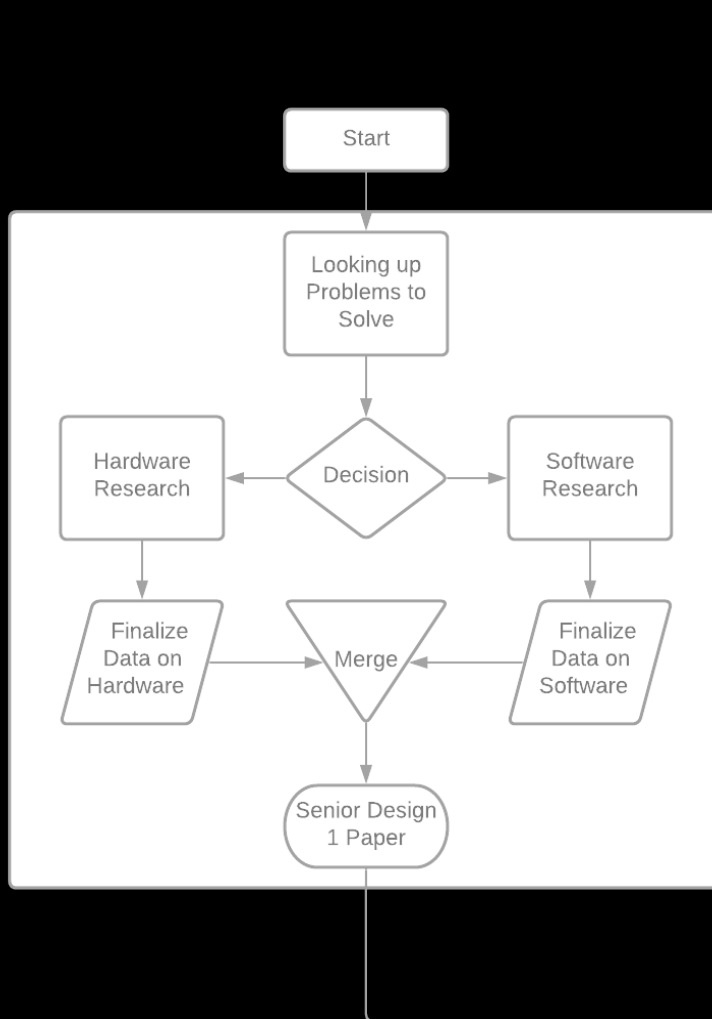


Standard	Component
IPC-2221 PCB Standard	PCB design and manufacturing
NFPA Electrical Safety Standards	Safety standards for electrical components
PEP8	Python style guide
IEEE standard 802.ac/n	Wi-Fi
Batteries IEEE 485-2010	Standard for Lead Acid and LiPo Batteries
FAA part 107 UAS	Drone Flight

Project & System Diagram



Block Diagram – Overall System or Project



Drone Design and Frame



- Our frame was chosen for cost as well as functionality
- Plastic when it breaks snaps apart, while carbon fiber splitters
- Carbon fiber also interferes with RF signal
- The cost was kept down using a bundle deal that included the ESCs, rotors, motors, and PDB



Battery



- We decided on the Turnigy nanotech 6000-mAh 3s LiPo
- This battery provided us, after utilizing the rate of consumption formula, that we could achieve a flight time of 20 minutes



Consumption Formula:

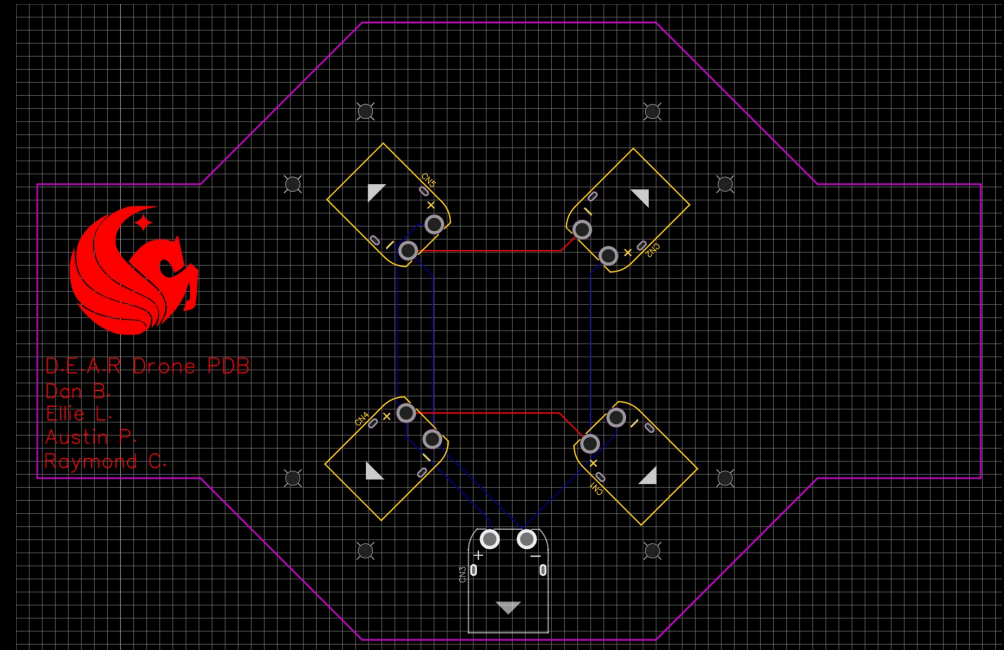
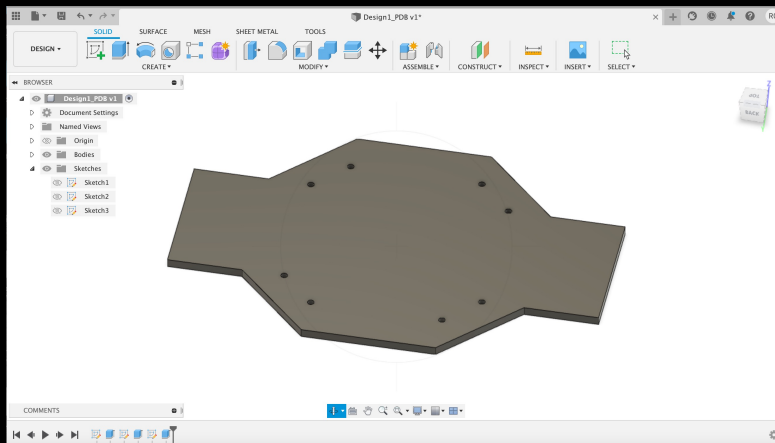
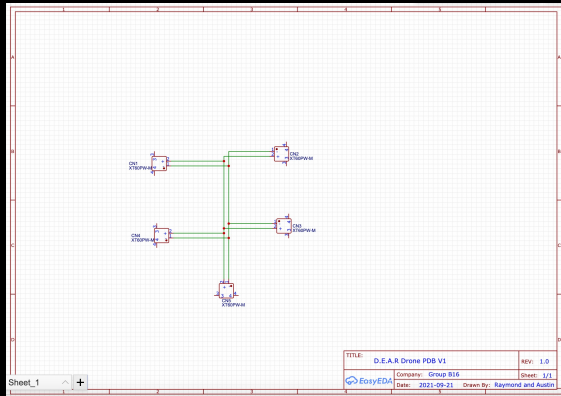
$(\text{Milliamp hours} / \text{avg amp draw}) * 60 = \text{amount of flight time in minutes}$

PCB Designs

PCB Design



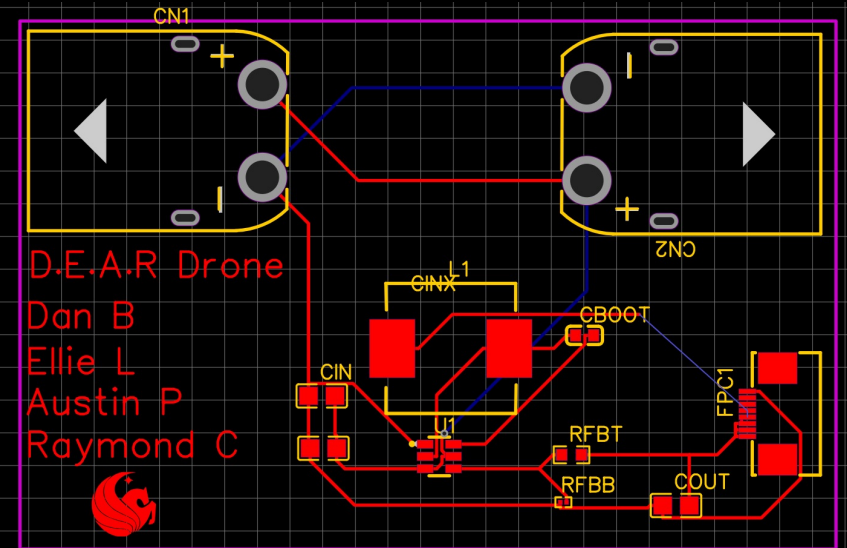
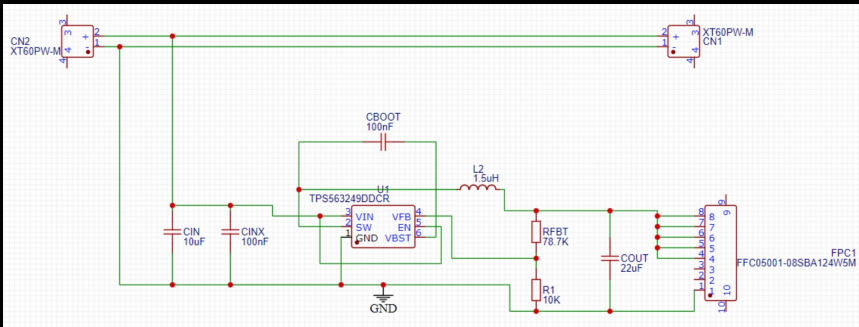
- Power Distribution Board
 - Send 11V evenly to all our ECS's



PCB Design



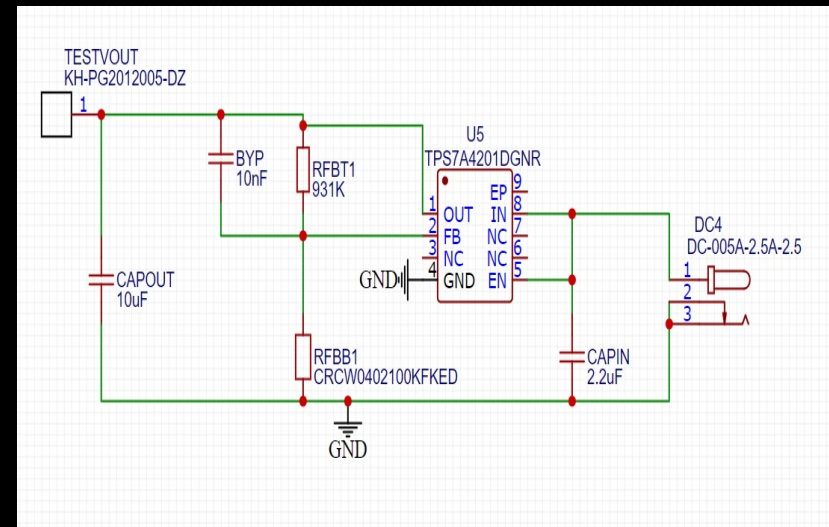
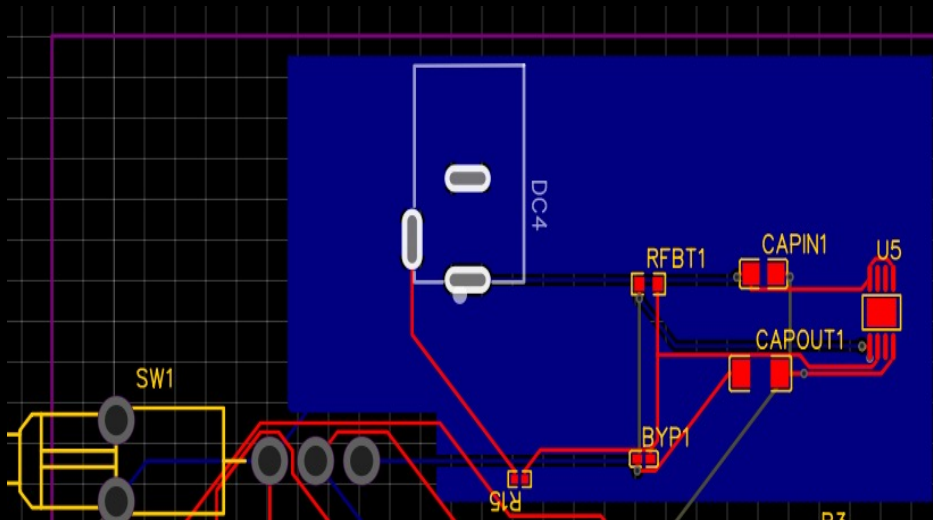
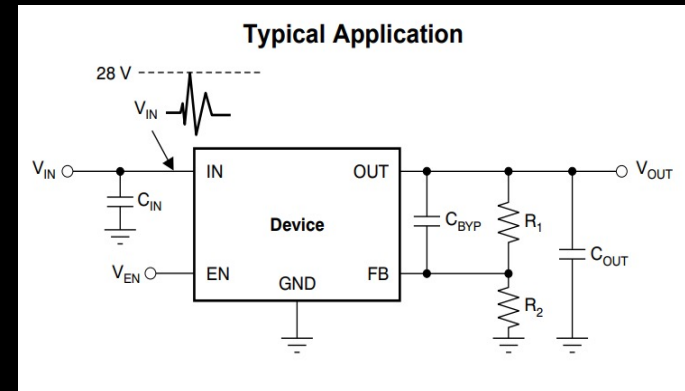
- Power Module
 - Sends 11V to our power distribution board
 - Steps down 11V to 5V and 2.5mA for the flight controller
 - XT60 to XT60 connection



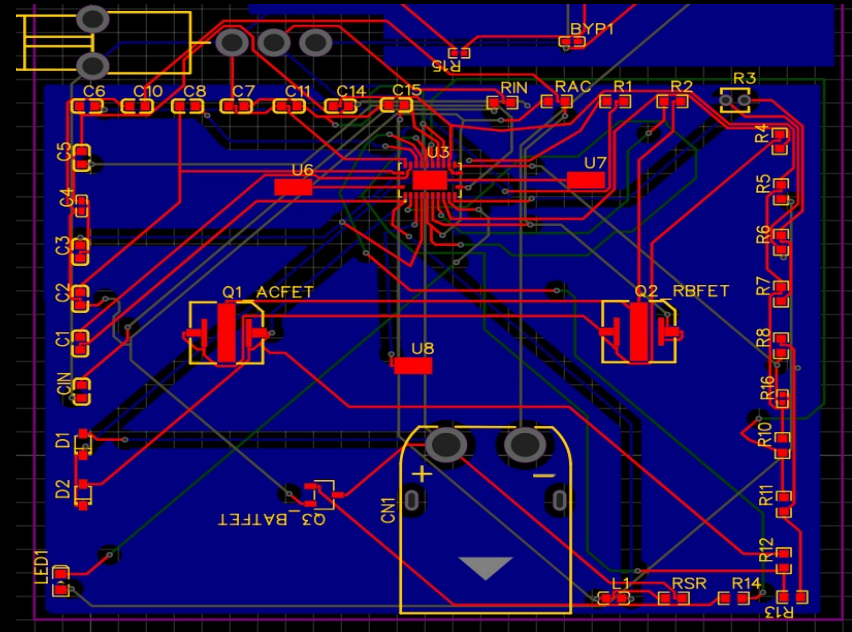
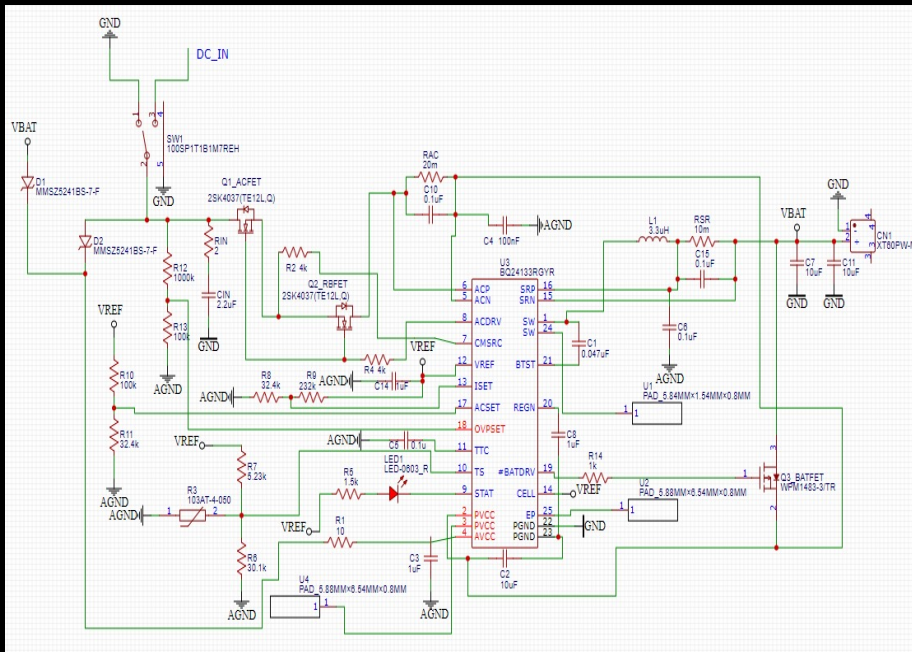
PCB Schematic [DC-DC Converter]



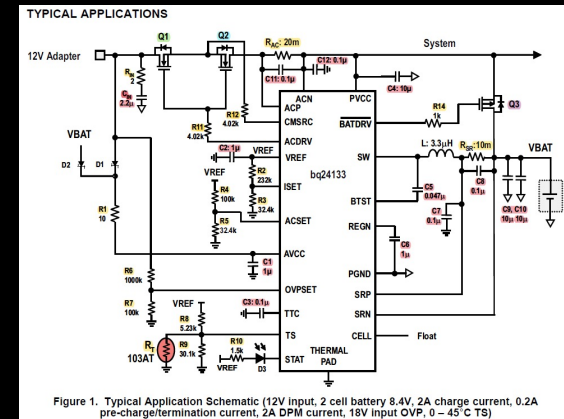
- DC-DC Converter
- Barrel Jack/ Laptop Charger
- Input DC Power 19V
- Output to Battery IC 12V at 1.5mA



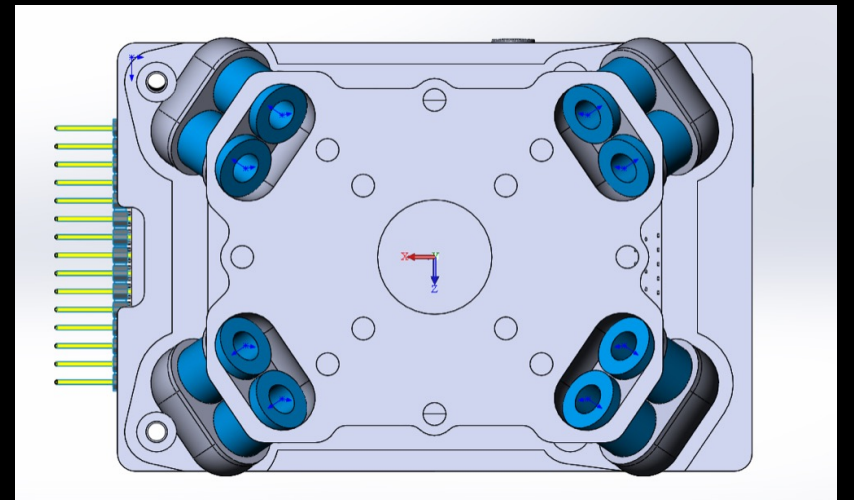
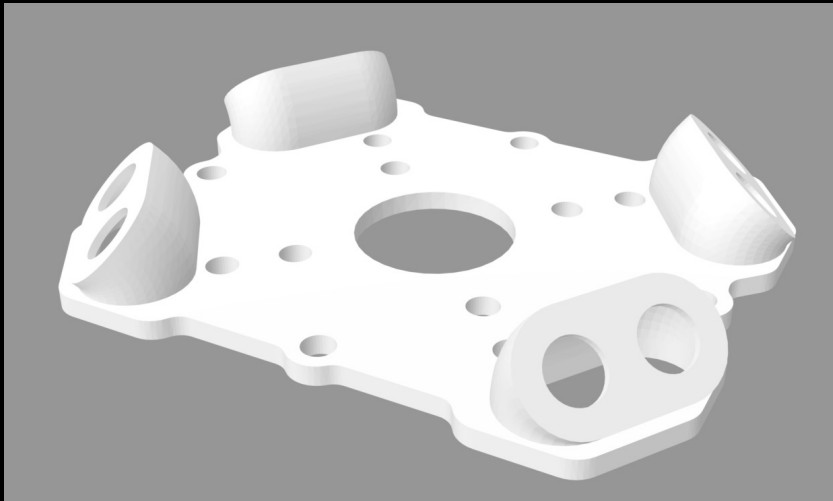
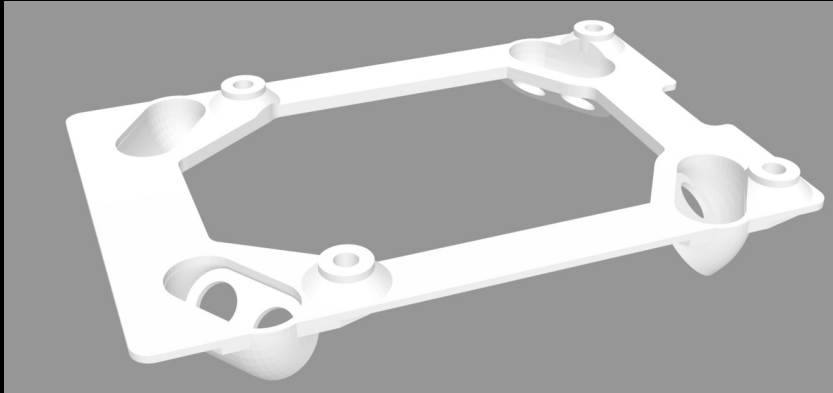
PCB Schematic [Battery Management IC]



- The design of the Battery Management system was done using the “Typical Application” provided by the Datasheet
- Main Chip component: BQ24133RGYR under the Family component: BQ24133
- Using this aspect of our chip, we can charge 3 Cells at one time, protect them, and manage them, it was an ideal choice.



Vibration Dampening Mount



Flight Control Board



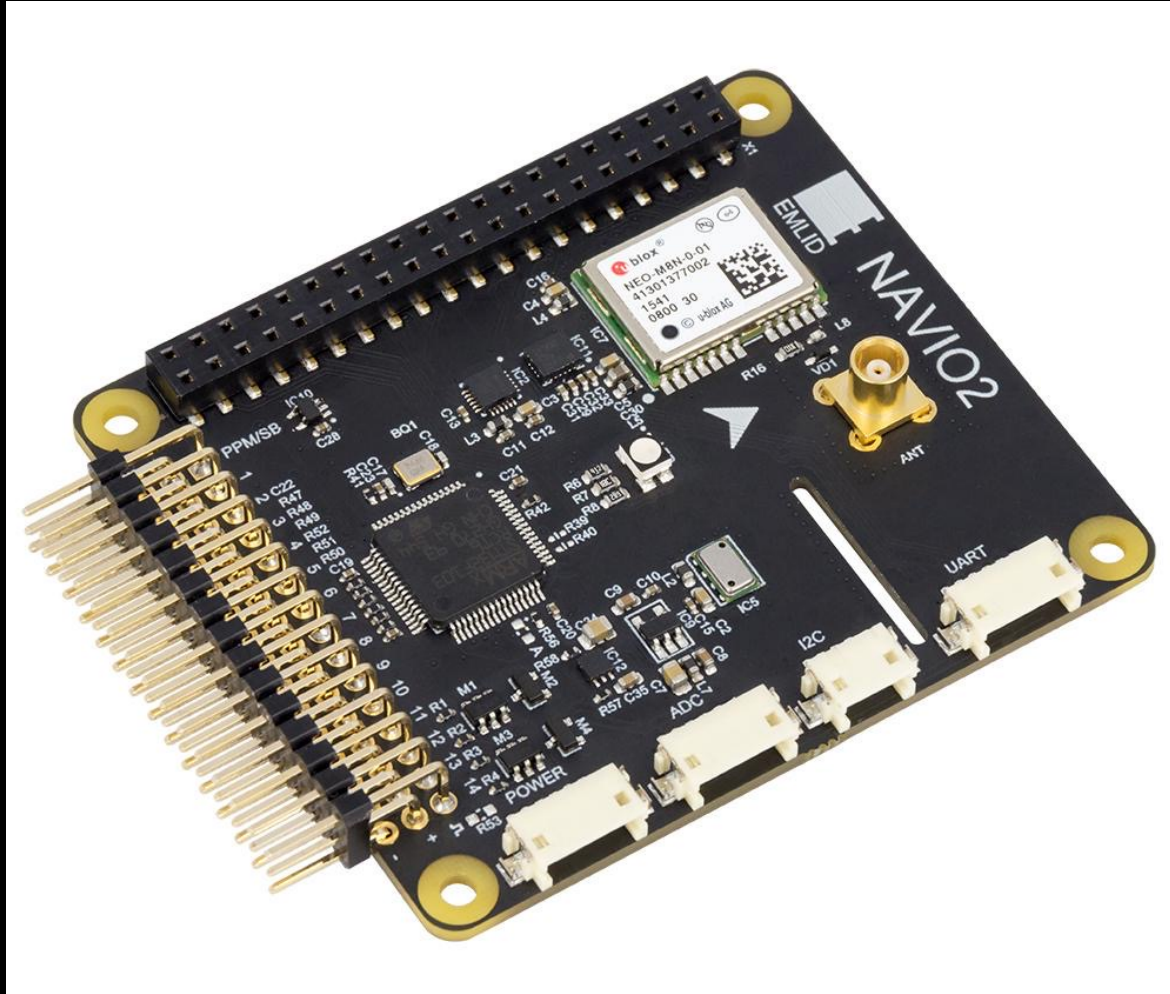
Board	Price (\$)	Weight (g)	Firmware	OS	Popularity	Bus Width
Pixhawk	80	60	ArduPilot, PX4	No	High	32 Bit
<u>Pixhawk Cube</u>	250	150	ArduPilot, PX4	No	High	32 Bit
<u>Navio2 + Raspberry Pi 4</u>	168 + 35/55/ 75	23	ArduPilot, PX4	Linux	Medium	64 Bit
<u>BeagleBone Blue</u>	80	52	ArduPilot, PX4	Linux	Low	32 Bit

Flight Control Board



Board	Processor	RAM	Ports	Sensors	Networking
Pixhawk 1	180 MHz ARM® Cortex® M4 with single-precision FPU	256 KB	I2C, CAN, ADC, UART, CONSOLE, PWM, PWM/GPIO/PWM input, S.BUS/PPM/SPEKTRUM, S.BUS out, microUSB	gyroscope, accelerometer/magnetometer, accelerometer/gyroscope, barometer	wifi, DSM-X
<u>Pixhawk Cube</u>	180 MHz ARM® Cortex® M4 with single-precision FPU	256 KB	Same as Pixhawk 1 but more UART ports + UART HW flow control	gyroscope, accelerometer/magnetometer, accelerometer/gyroscope, barometer	wifi, DSM-X
<u>Navio2 + Raspberry Pi 4</u>	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz	2GB, 4GB or 8GB	UART, I2C, ADC, 12 PWM servo outputs, PPM/S.Bus input, HDMI, MIPI display port, MIPI camera port, microSD, USB3.0, USB 2.0	Accelerometers, gyroscopes, and magnetometers, Barometer	2.4GHz WiFi, Bluetooth, BLE
<u>BeagleBone Blue</u>	Octavo Systems OSD3358 1GHz ARM Cortex-A8	512 MB	4x UART, 2-cell LiPo, 2x SPI, I2C, 4x A/D converter, CAN bus (w/PHY), 8x 6V servo motor, 4x DC motor, 4x quadrature encoder, USB 2.0 480Mbps Host/Client Port, USB 2.0 Host Port	9 axis IMU (access, gyros, magnetometer), barometer, thermometer	2.4GHz WiFi, Bluetooth, BLE

Navio2



Data Processing Computer



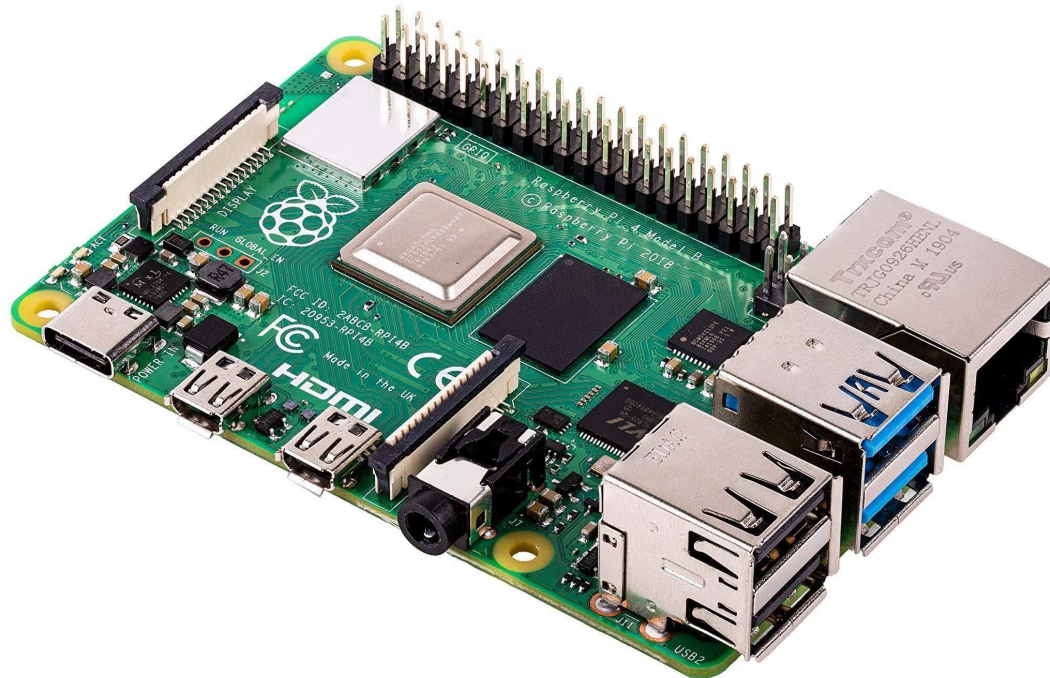
	Raspberry Pi 4B	NVIDIA Jetson Nano
CPU	Quad-core ARM Cortex-A72 64-bit @ 1.5 Ghz	Quad-Core ARM Cortex-A57 64-bit @ 1.42 Ghz
GPU	Broadcom VideoCore VI (32-bit) (Integrated)	NVIDIA Maxwell w/ 128 CUDA cores @ 921 Mhz
Memory	4 GB LPDDR4**	4 GB LPDDR4
Networking	Gigabit Ethernet / Wifi 802.11ac	Gigabit Ethernet / M.2 Key E (for Wifi support)
Display	2x micro-HDMI (up to 4Kp60)	HDMI 2.0 and eDP 1.4
USB	2x USB 3.0, 2x USB 2.0	4x USB 3.0, USB 2.0 Micro-B
Other	40-pin GPIO	40-pin GPIO
Video Encode	H264(1080p30)	H.264/H.265 (4Kp30)
Video Decode	H.265(4Kp60), H.264(1080p60)	H.264/H.265 (4Kp60, 2x 4Kp30)
Camera	MIPI CSI port	MIPI CSI port
Storage	Micro-SD	Micro-SD
Price	\$55 USD	\$99 USD

Data Processing Computer

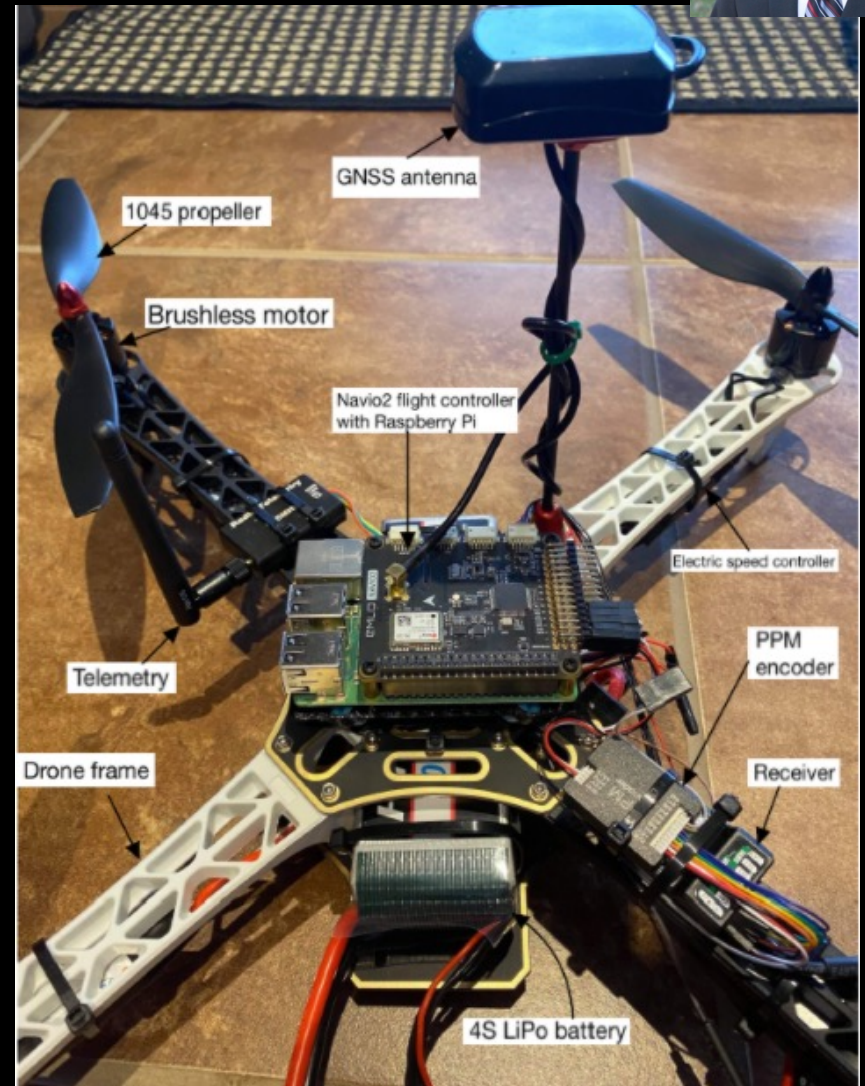
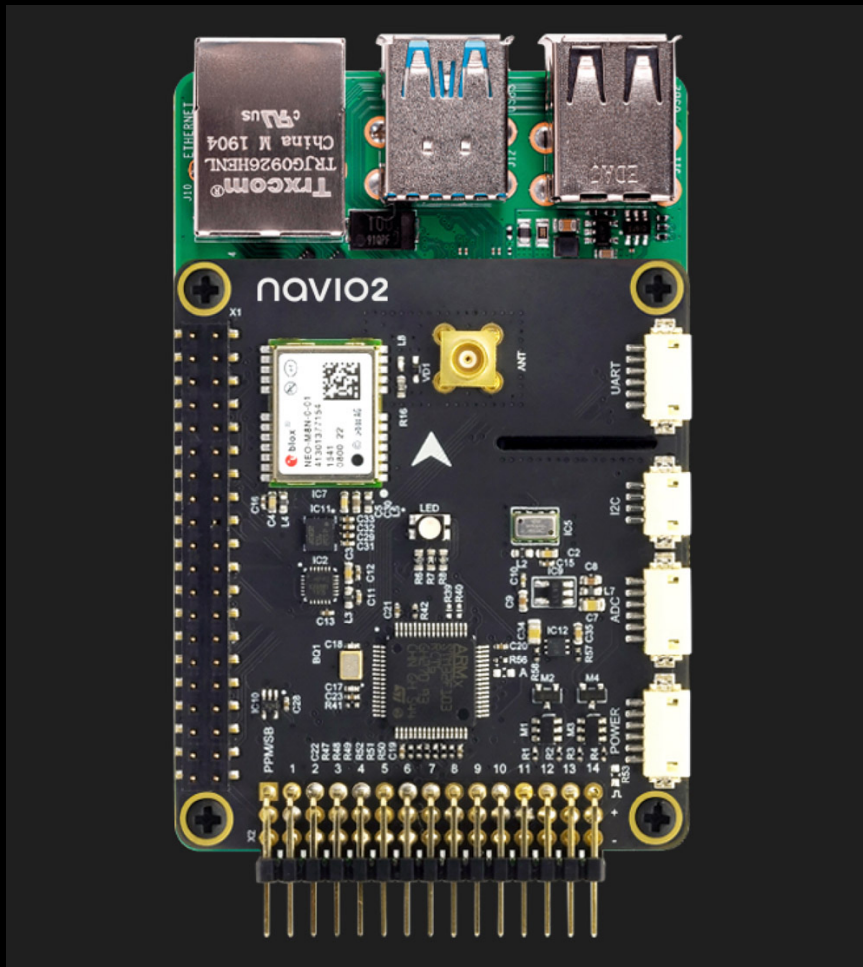


Test	Raspberry Pi 4 B 8GB	Jetson Nano 4GB
Tinymembenc Standard Memcpy	2526.8 MB/s	3501 MB/s
TTSIOD 3D Renderer Phong Rendering With Soft-Shadow Mapping	32.3075 FPS	41.00 FPS
7-Zip Compression Compress Speed Test	3701 MIPS	3501 MIPS
C-Ray Total Time – 4K, 16 Rays Per Pixel	609.431 Seconds	932 Seconds
Primesieve 1e12 Prime Number Generation	519.238 Seconds	468 Seconds
AOBench Size: 2048 x 2048	125.643 Seconds	187 Seconds
FLAC Audio Encoding WAV To FLAC	85.805 Seconds	104.01 Seconds
LAME MP3 Encoding WAV To MP3	124.952 Seconds	144.21 Seconds
Perl Pod2html	0.58082059 Seconds	0.7114 Seconds
Redis GET	491168.39 Requests/Second	568431 Requests/Second
PyBench Total For Average Test Times	5673 Milliseconds	7080 Milliseconds

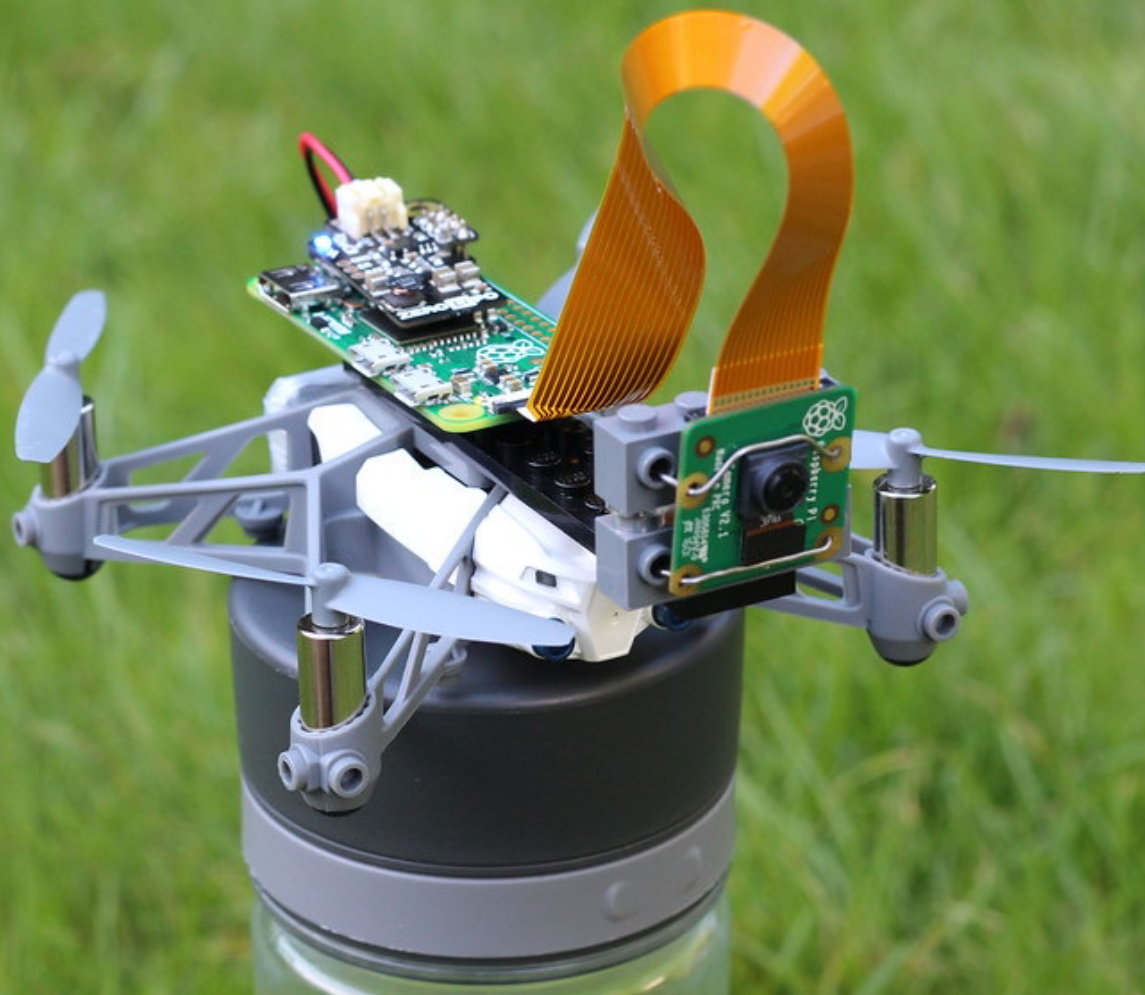
Raspberry Pi 4B



Flight Control Unit



Camera



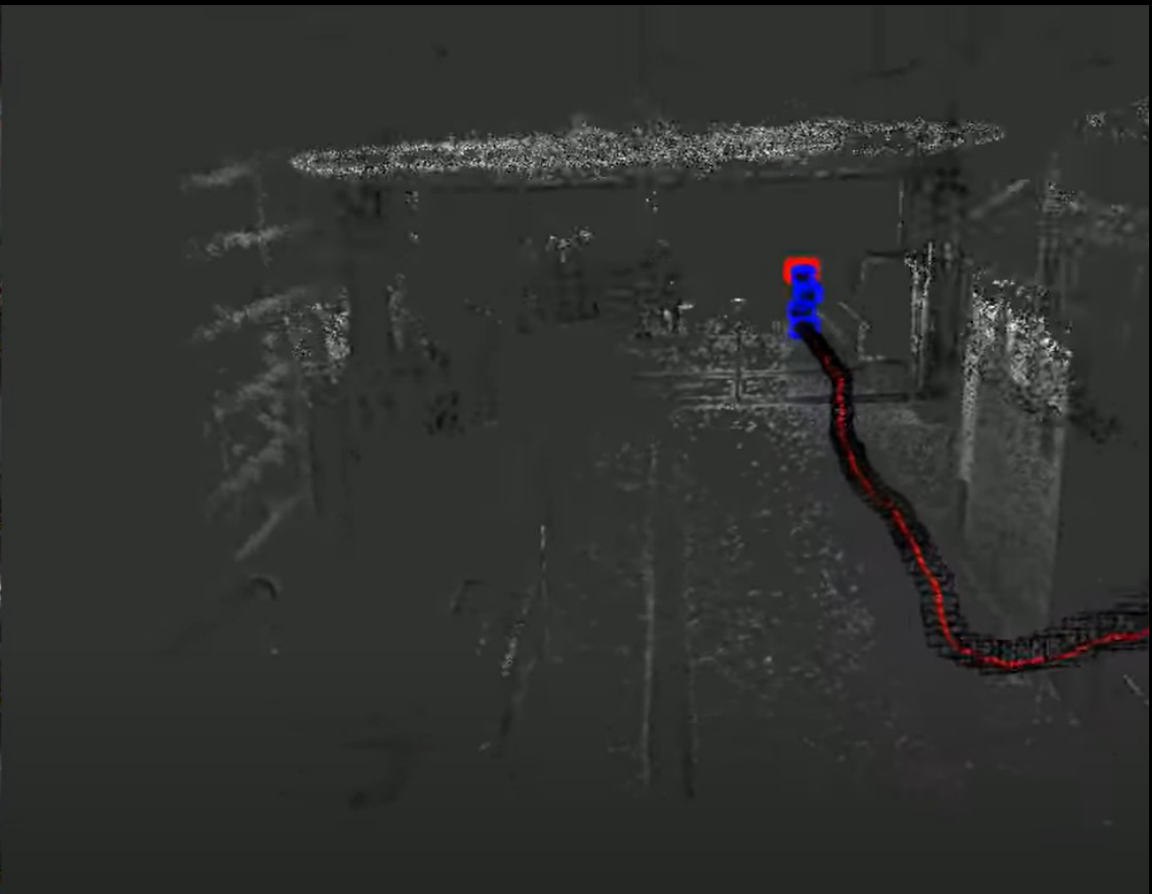
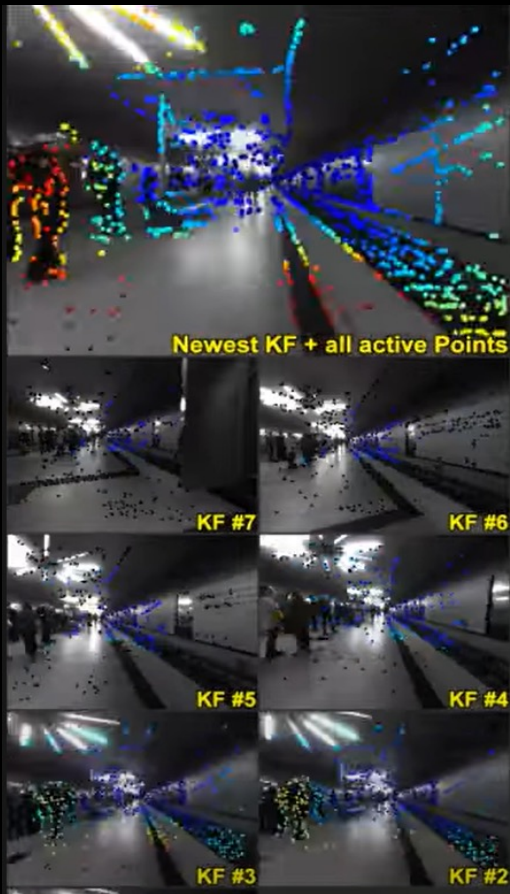
Mapping Software



Mapping Algorithm	Frame Times (ms)		CPU @20fps
	Mean	St.D.	
ORB Mono SLAM	29.81	5.67	187
LSD Mono SLAM	23.23	5.87	236
DSO Mono	9	-	-
SVO Mono	2.53	0.42	55

- We decided to use ORB-SLAM, and keep the others as backups in case ORB-SLAM does not run fast enough.

ORB-SLAM3 3.0

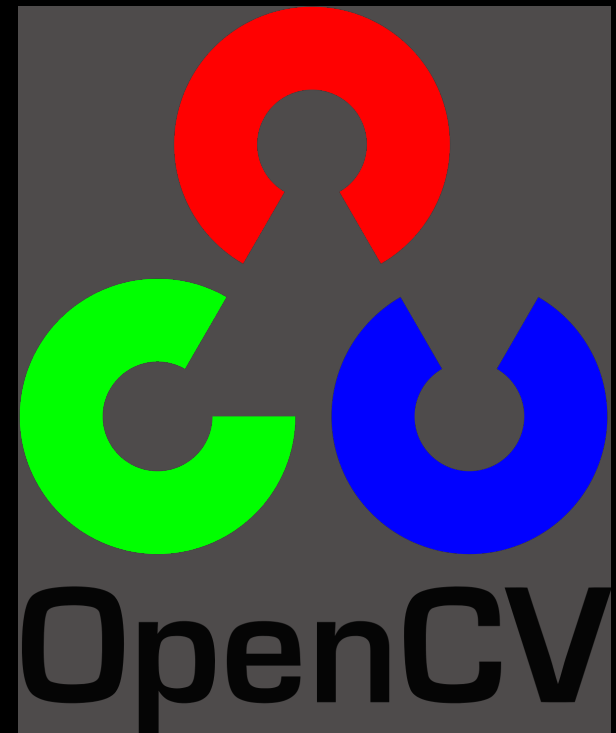
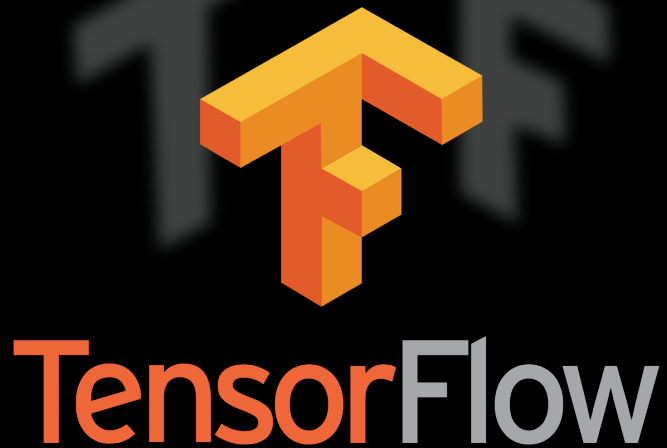




Problems with 3D Mapping

```
gives object file: no such file or directory
[navigation_actor-6] process has died [pid 8521, exit code 127, cmd /home/tom/catkin_ws/src/nm_dependencies/nm_navigation/devel/NavigationActor_x86_64_16.04 /rtabmap/grid_map:=/map /cmd_vel/managed:=/motor_controller/command /cmd_vel:=/motor_controller/command_no_stamp __name:=navigation_actor __log:=/home/tom/.ros/log/b2de0418-0818-11ea-bf64-08002711b324/navigation_actor-6.log].
log file: /home/tom/.ros/log/b2de0418-0818-11ea-bf64-08002711b324/navigation_actor-6*.log
/home/tom/catkin_ws/src/nm_dependencies/nm_navigation/devel/ServerVitals_x86_64_16.04: x=V Jopn08+&F/Gagi
ā(GowLea5t杉
-@MD$&&hnde/1u982+
process[ServerVitals-7]: started with pid [8522]
[ServerVitals-7] process has died [pid 8522, exit code 1, cmd /home/tom/catkin_ws/src/nm_dependencies/nm_navigation/devel/ServerVitals_x86_64_16.04 /rtabmap/grid_map:=/map /cmd_vel/managed:=/motor_controller/command /cmd_vel:=/motor_controller/command_no_stamp __name:=ServerVitals __log:=/home/tom/.ros/log/b2de0418-0818-11ea-bf64-08002711b324/ServerVitals-7.log].
log file: /home/tom/.ros/log/b2de0418-0818-11ea-bf64-08002711b324/ServerVitals-7*.log
```

Computer Vision



Computer Vision



```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
5
6 classNames = []
7 classFile = 'item.names'
8 with open(classFile, 'rt') as f:
9     classNames = f.read().rstrip('\n').split('\n')
10
11 configPath = 'ssd_path.pbtxt'
12 weightsPath = 'inference_graph.pb'
13
14 net = cv2.dnn_DetectionModel(weightsPath, configPath)
15 net.setInputSize(320, 320)
16 net.setInputScale(1.0 / 127.5)
17 net.setInputMean((127.5, 127.5, 127.5))
18 net.setInputSwapRB(True)
19
20
21 def getObjects(img, thres, nms, draw=True, objects=[]):
22     classIds, confs, bbox = net.detect(img, confThreshold=thres, nmsThreshc
23     #print(classIds, bbox)
24     if len(objects) == 0:
25         objects = classNames
26     objectInfo = []
27     if len(classIds) != 0:
28         for classId, confidence, box in zip(classIds.flatten(), confs.flatten
```

```
while True:
    success, img = cap.read()
    results, objectInfo = getObjects(img, 0.45, 0.2, True, objects=['person'])
    # converting image into grayscale image
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # setting threshold of gray image
    _, threshold = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
    # using a findContours() function
    contours, _ = cv2.findContours(
        threshold, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    i = 0
    # list for storing names of shapes
    for contour in contours:
        # here we are ignoring first counter because
        # findcontour function detects whole image as shape
        if i == 0:
            i = 1
            continue
        # cv2.approxPloyDP() function to approximate the shape
        approx = cv2.approxPolyDP(contour, 0.01 * cv2.arcLength(contour, True), True)
```

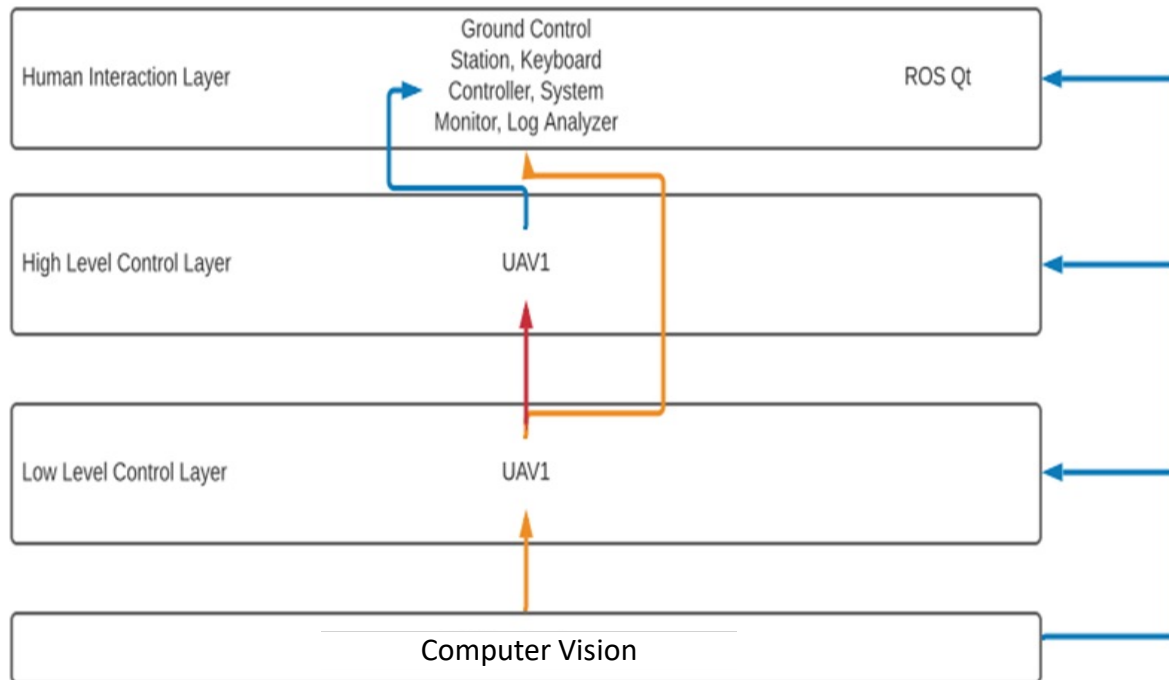
Computer Vision



The screenshot displays a Python IDE with a window titled "Output" showing a video frame. A green bounding box surrounds the person, labeled "PERSON" with a confidence score of 65.79. Another green bounding box surrounds a cup, labeled "CUP" with a confidence score of 79. The IDE background shows the source code for "ObjectDetectorModule.py" and a terminal window with the following output:

```
[[array([398, 171, 131, 200], dtype=int32), 'cup'], [array([ 70, 66, 566, 414], dtype=int32), 'person']]  
[[array([399, 174, 134, 200], dtype=int32), 'cup'], [array([ 71, 66, 564, 413], dtype=int32), 'person']]  
[[array([397, 169, 131, 200], dtype=int32), 'cup'], [array([ 67, 66, 569, 413], dtype=int32), 'person']]  
[[array([398, 174, 136, 199], dtype=int32), 'cup'], [array([ 68, 63, 568, 416], dtype=int32), 'person']]  
[[array([401, 174, 132, 200], dtype=int32), 'cup'], [array([ 73, 66, 562, 413], dtype=int32), 'person'], [array([412, 173, 100, 128], dtype=int32), 'cup']]  
[[array([397, 174, 132, 192], dtype=int32), 'cup'], [array([ 70, 67, 565, 412], dtype=int32), 'person'], [array([411, 172, 101, 131], dtype=int32), 'cup']]  
[[array([398, 174, 134, 199], dtype=int32), 'cup'], [array([ 70, 64, 566, 416], dtype=int32), 'person'], [array([410, 173, 103, 132], dtype=int32), 'cup']]  
[[array([400, 173, 134, 200], dtype=int32), 'cup'], [array([ 70, 65, 565, 414], dtype=int32), 'person'], [array([411, 172, 103, 130], dtype=int32), 'cup']]
```

Software Flowchart



QGroundControl



The screenshot displays the QGroundControl interface with several key components:

- Top Bar:** Includes a menu (File, Widgets), a toolbar with icons for home, settings, location, and zoom, and a status bar showing battery at 100%, signal strength, and a "Hold Disarmed" button. The "Fly Toolbar" logo and "PX4 AUTOPILOT" branding are also present.
- Left Panel:** A "Fly" sidebar with a red border containing "Fly tools" (Takeoff, RTL, Pause, Action).
- Map:** An aerial view of Zurich University with a mission plan consisting of four numbered waypoints (1, 2, 3, 4) connected by orange lines. A red triangle labeled 'H' is positioned at waypoint 2. A "Confirmation Slider" is located near waypoint 3.
- Right Panel:** An "Instrument Panel" with a yellow border. It features an "Attitude/Compass" gauge showing pitch and roll, and a "Values" section with the following data:

Parameter	Value
Alt (Rel) (m)	-0.0
Ground Speed (m/s)	0.0
Flight Time	00:00:00
- Bottom Center:** A "Start Mission" dialog box with a close button (X) and a "Slide to confirm" button.
- Bottom Left:** A "Video / Switcher" window showing a "WAITING FOR VIDEO" message.

QGroundControl



0 100.0 46% Manual

Below you will find a summary of the settings for your vehicle. To the left are the setup menus for each component.

Summary	Airframe ●		Radio ●	
Firmware	System ID:	1	Roll:	Setup required
Airframe	Airframe type:	Quadrotor x	Pitch:	2
Radio	Vehicle:	Generic 250 Racer	Yaw:	4
Sensors	Firmware Version:	Unknown	Throttle:	3
Flight Modes	Sensors ●		Flight Modes ●	
Power	Compass 0:	Ready	Mode switch:	Channel 5
Safety	Compass 1:	Ready	Flight Mode 1 :	Stabilized
Tuning	Compass 2:	Ready	Flight Mode 2 :	Unassigned
Camera	Gyro:	Ready	Flight Mode 3 :	Unassigned
Parameters	Accelerometer:	Ready	Flight Mode 4 :	Position
	Power ●		Safety	
	Battery Full:	4.05 V	RTL min alt:	30.0 m
	Battery Empty:	3.40 V	RTL home alt:	10.0 m
	Number of Cells:	3	RC loss RTL:	0.5 s
			RC loss action:	Disabled
			Link loss action:	Disabled
			Low battery action:	Warning
	Camera			

QGroundControl



Vehicle Setup | Search: Clear | 100% Hold Disarmed | PX4 PRO

Summary | Firmware | Airframe | Radio | Sensors | Flight Modes | Power | Safety | Tuning | Camera | **Parameters**

Component #:	Parameter	Value	Description
*Default Group	BAT_A_PER_V	15.39103031	Battery current per volt (A/V)
Battery Calibration	BAT_CAPACITY	-1 mA	Battery capacity
Camera trigger	BAT_CNT_V_CURR	0.00080566	Scaling from ADC counts to volt on the ADC input (battery current)
Circuit Breaker	BAT_CNT_V_VOLT	0.00080566	Scaling from ADC counts to volt on the ADC input (battery voltage)
Commander	BAT_CRIT_THR	7 %	Critical threshold
Data Link Loss	BAT_EMERGEN_THR	5 %	Emergency threshold
EKF2	BAT_LOW_THR	15 %	Low threshold
FW Attitude Control	BAT_N_CELLS	3S Battery	Number of cells
FW L1 Control	BAT_R_INTERNAL	-1.000 Ohms	Explicitly defines the per cell internal resistance
FW Launch detection	BAT_SOURCE	Power Module	Battery monitoring source
FW TECS	BAT_V_CHARGED	4.05 V	Full cell voltage (5C load)
Follow target	BAT_V_DIV	10.17793941	Battery voltage divider (V divider)
GPS Failure Navigation	BAT_V_EMPTY	3.40 V	Empty cell voltage (5C load)
Geofence	BAT_V_LOAD_DROP	0.30 V	Voltage drop per cell on full throttle
Land Detector	BAT_V_OFFS_CURR	0.00000000	Offset in volt as seen by the ADC input of the current sensor

QGround Control

Back < Vehicle Setup

Summary

PID Tuning Setup

Tuning Setup is used to tune the flight controllers.

Firmware

Rate Controller

Attitude Controller

Joystick

Airmode (disable during tuning) ? : Disabled

Airframe

Thrust curve ? : 0.000

Sensors

Radio

Flight Modes

Power

Motors

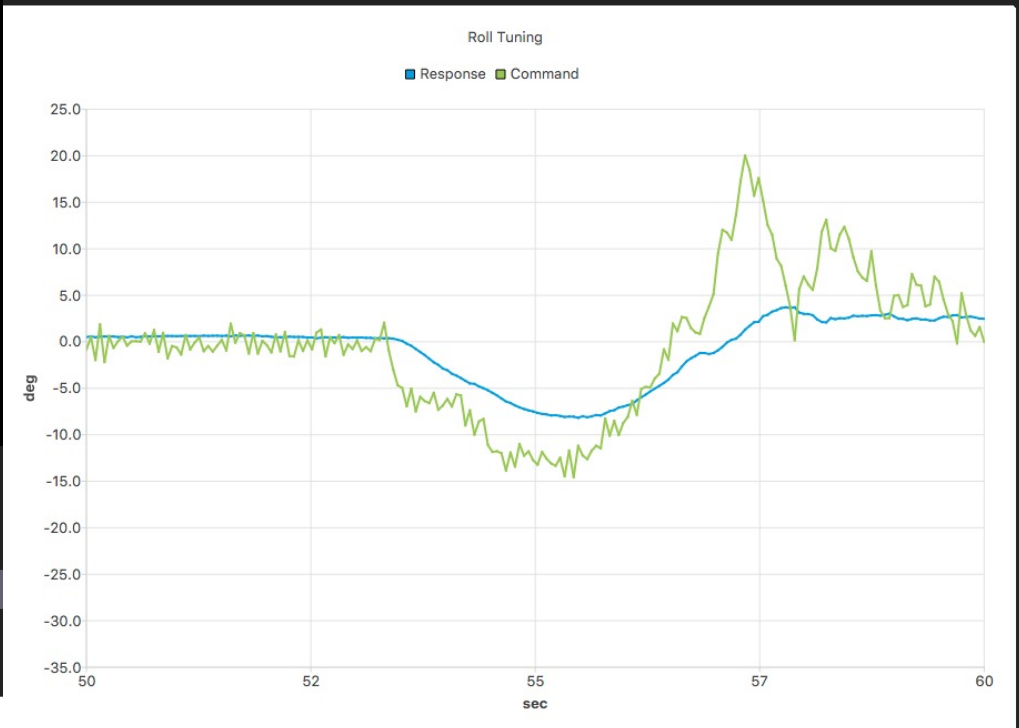
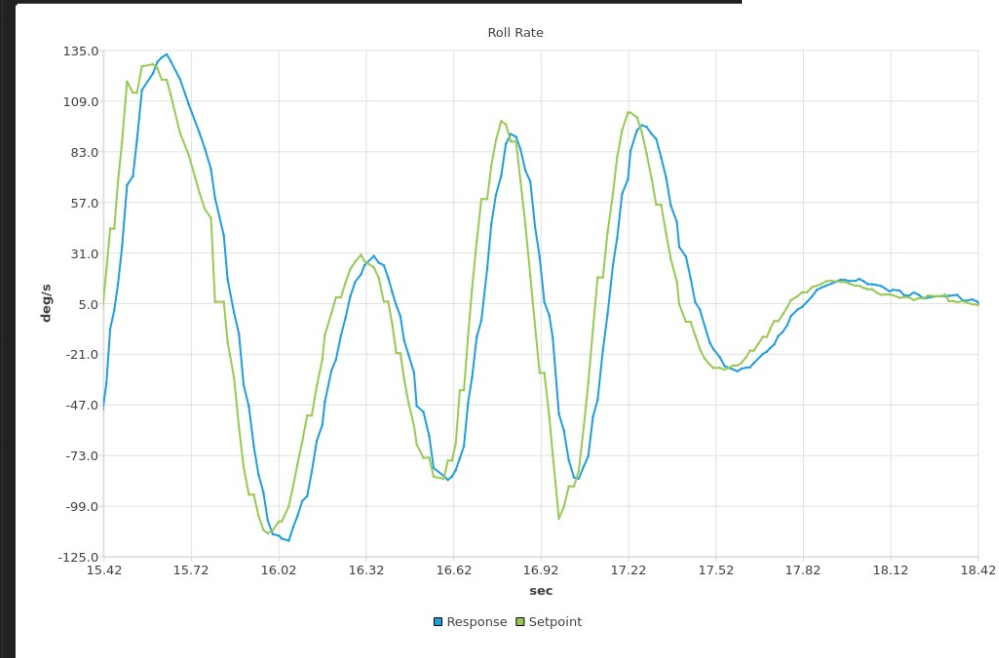
Safety

PID Tuning

Flight Behavior

Camera

Parameters



Overall Multiplier (MC_ROLLRATE_K)

0.3 1.1 3

Multipier for P, I and D gains: increase for more responsiveness, reduce if the rates overshoot (and increasing D does not help).

Differential Gain (MC_ROLLRATE_D)

0.0004 0.0032 0.01

Damping: increase to reduce overshoots and oscillations, but not higher than really needed.

Integral Gain (MC_ROLLRATE_I)

0.1 0.2 0.5

Generally does not need much adjustment, reduce this when seeing slow oscillations.

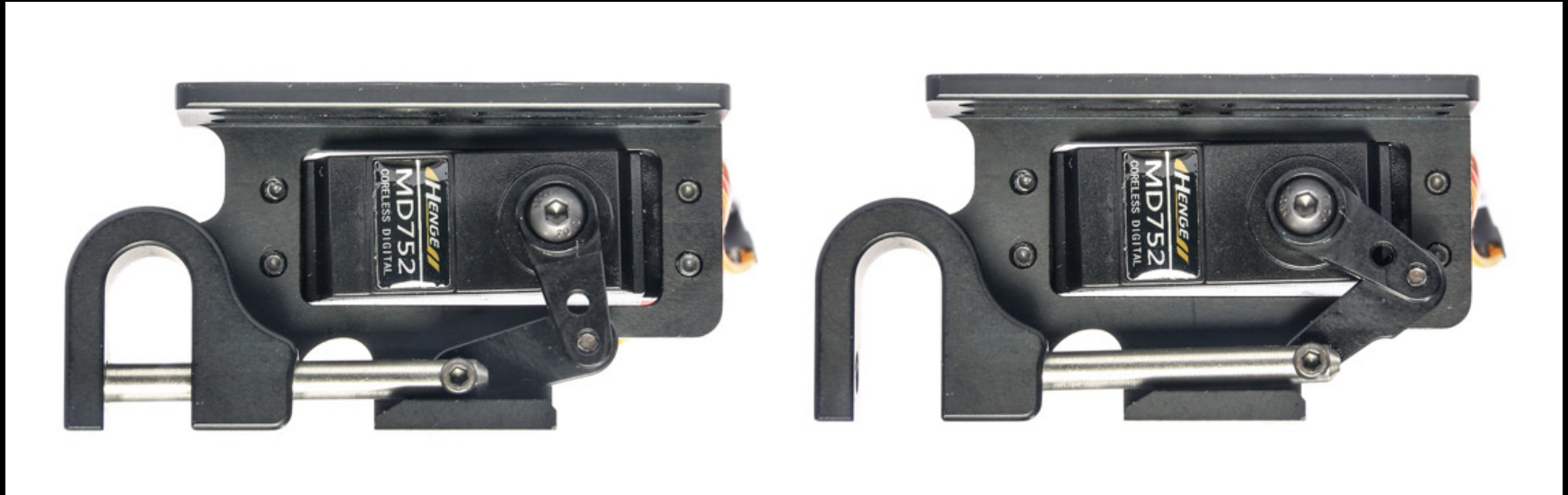
Clipboard Values:
MC_ROLLRATE_K 1.1000
MC_ROLLRATE_D 0.0032
MC_ROLLRATE_I 0.200

Save To Clipboard Restore From Clipboard

Clear Start

Automatic Flight Mode Switching

Package Release Mechanism



Sensors



Camera

Barometer

GPS

3-axis

Gyroscope

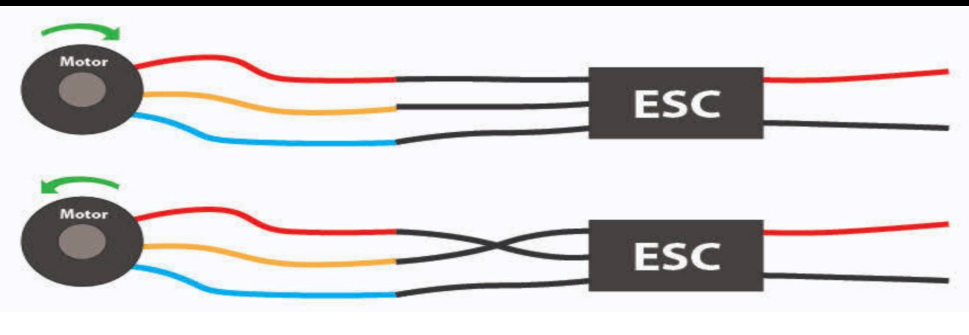
Accelerometer

Compass

Electric Speed Controller



- Electric Speed Controller
 - Translator from the 'pilot' to the motor by giving specific voltage/current based on input from 'user'
 - Options we had to choose from
 - 4 in 1 packages or individual
 - Individual is most cost effective because if one breaks only one must be replaced
 - BEC vs no BEC
 - Battery Elimination Circuit: provides control of power to other components
 - Not necessary for our drone project, but was included in the kit purchased as a cost-effective decision



PX4 vs Ardupilot



	PX4	Ardupilot
Coding Language	C++	C++
Majority Hardware Capabilities	Yes	Yes
Majority Software Capabilities	Yes	Yes
Release Date	2009	2012





Administration Content

Constraints



Economic

- Covid-19
- Self-funded

Health and Safety

- LiPo batteries
- PCBs and wires

Environmental

- Batteries
- Litter the environment

Manufacturability

- Fusion360 to make the STL files to 3D printing

Legal

- Open airfield
- Not on UCF grounds

Ethical

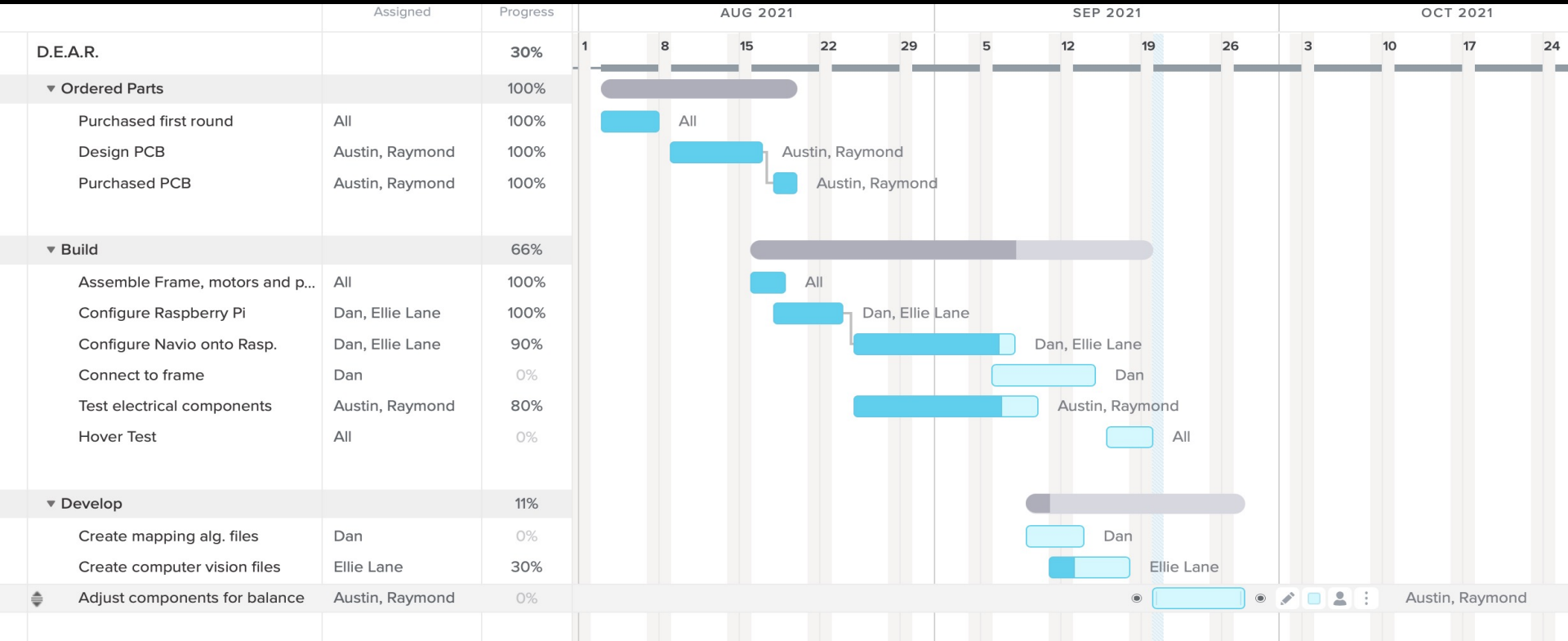
- Our project should only positively affect the businesses we are getting supplies from and businesses that will be helped by our drones' success

Budget



Frame	2	
Brushless Motors	8	
ESC	4	
Propellers	8	\$125.72
Navio 2 - Flight Controller	1	\$246
Raspberry Pi 4b	1	\$75
Battery	2	\$92.84
PCB	3	\$133
Electronic Components	6	\$6.99
Ultrasonic sensors	6	\$10.59
Transmitter	1	\$29.81
Antenna Mount	1	\$7.33
Vibration Dampeners	24	\$6.38
Camera	1	\$13.10
Total		\$746.76

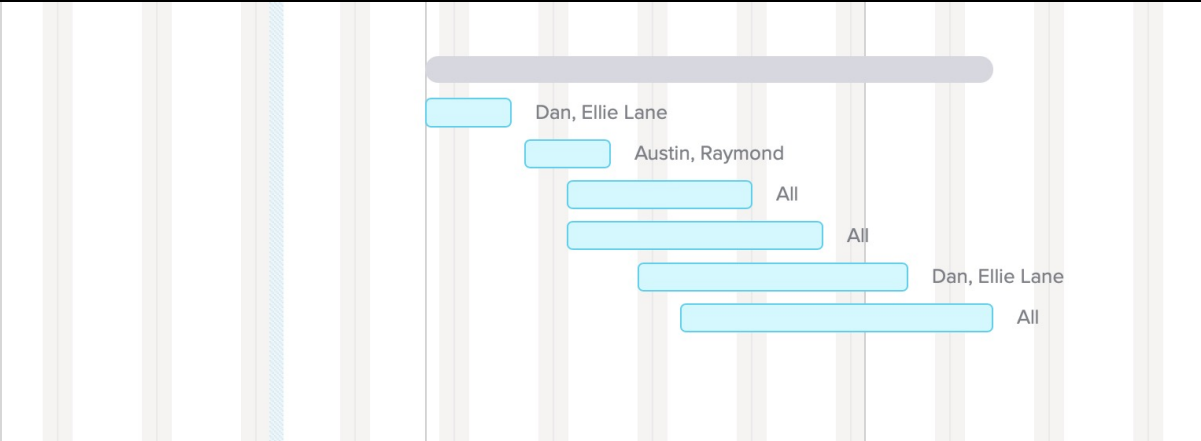
Gaant Chart



Gaant Chart Cont.



▼ Test		
Test using simulation	Dan, Ellie Lane	0%
Test electrical components	Austin, Raymond	0%
Take off test	All	0%
Path test	All	0%
Object detection Test	Dan, Ellie Lane	0%
Full path test	All	0%



Challenges



- Hardware
 - Components Available
 - Manufactures
 - Shipping Time
- Software
 - Setting up development environment
 - Communication between development computer and flight controller
 - Navigating the options
- Overall Challenge
 - Self-Funded

Improvements



- Hardware
 - Incorporating Shell
 - Improving Motor Power/Lift off ratio
 - Improving Receiver/Transceiver
- Software
 - Reducing Latency
 - Seamless Linux set-up
- Overall Improvements
 - More funding

Thank you!

Q&A



UCF