Hopper Spacecraft Simulator

Billy Hau and Brian Wisniewski

Agenda

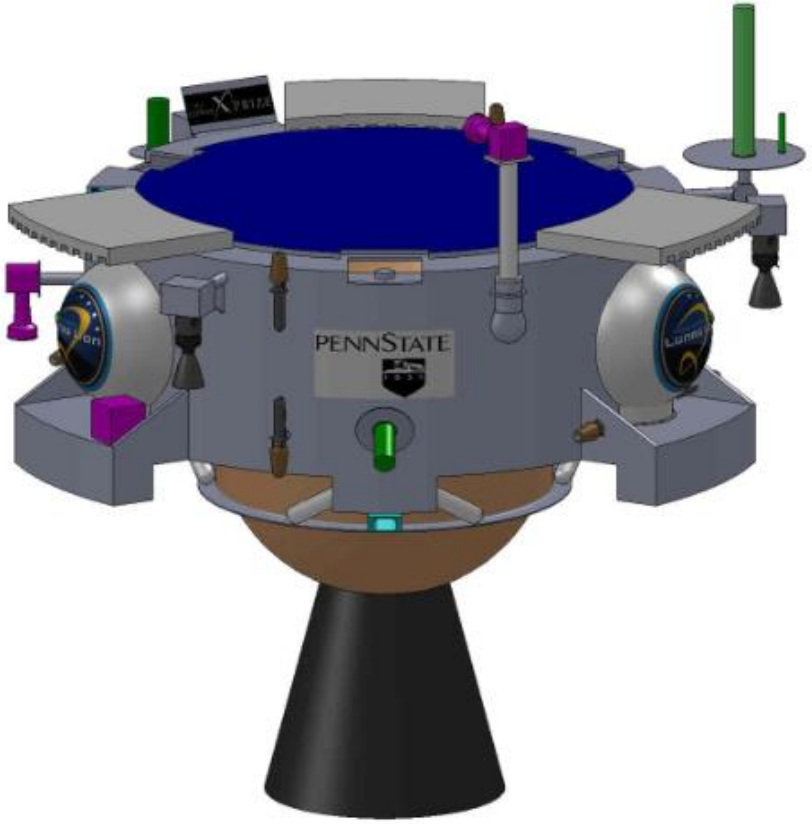
- Introduction
- Flight Dynamics
- Hardware Design
- Avionics
- Control System
- Future Works

Introduction

Mission Overview

- Collaboration with Penn State University on mission to compete in the Google Lunar XPRIZE competition

- Lehigh University focus on development of the Guidance, Navigation & Controls (GNC) system



Google LUNAR XPRIZE

FROM THE EARTH...

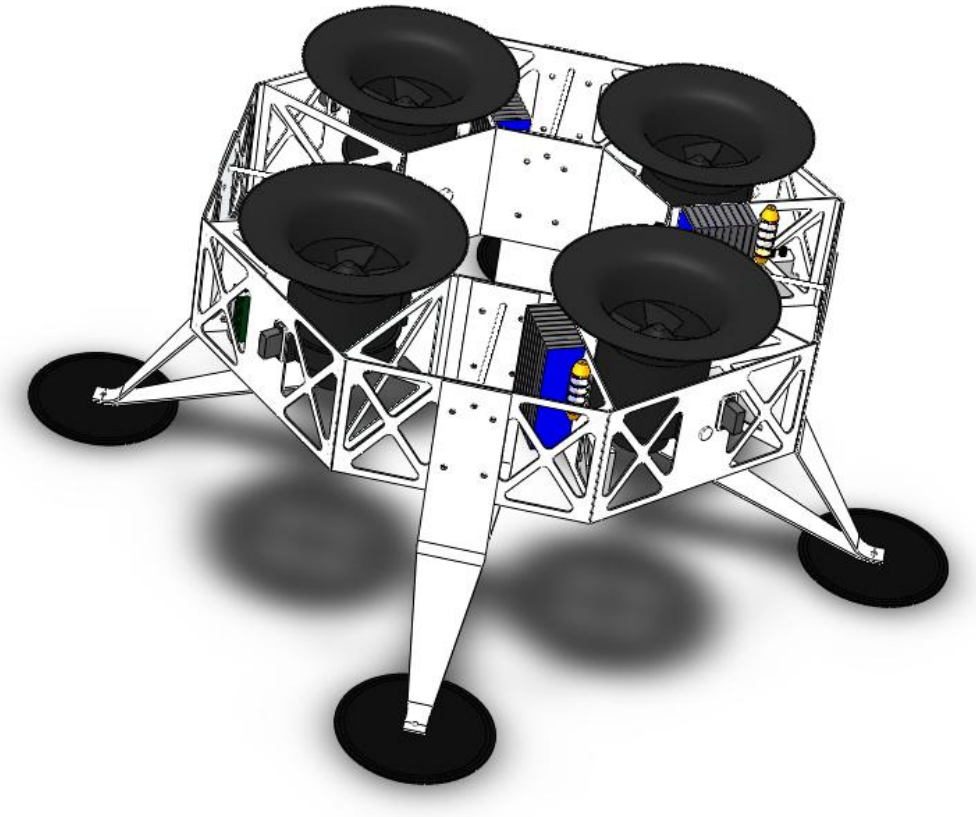
...TO THE moon!

1. Launch into Low Earth Parking Orbit
2. Craft separates from rocket
3. Main engine fires for trip to moon, then detaches
4. After four days of cruising, second engine fires and stops lander a few km from lunar surface, then detaches
5. Thrusters fire as the lander falls, allowing a soft landing
6. Thrusters fire again, and the lander "hops" 500 meters to another location

Lander sends images and video back to Earth

Earth-Based Hopper Spacecraft Simulator

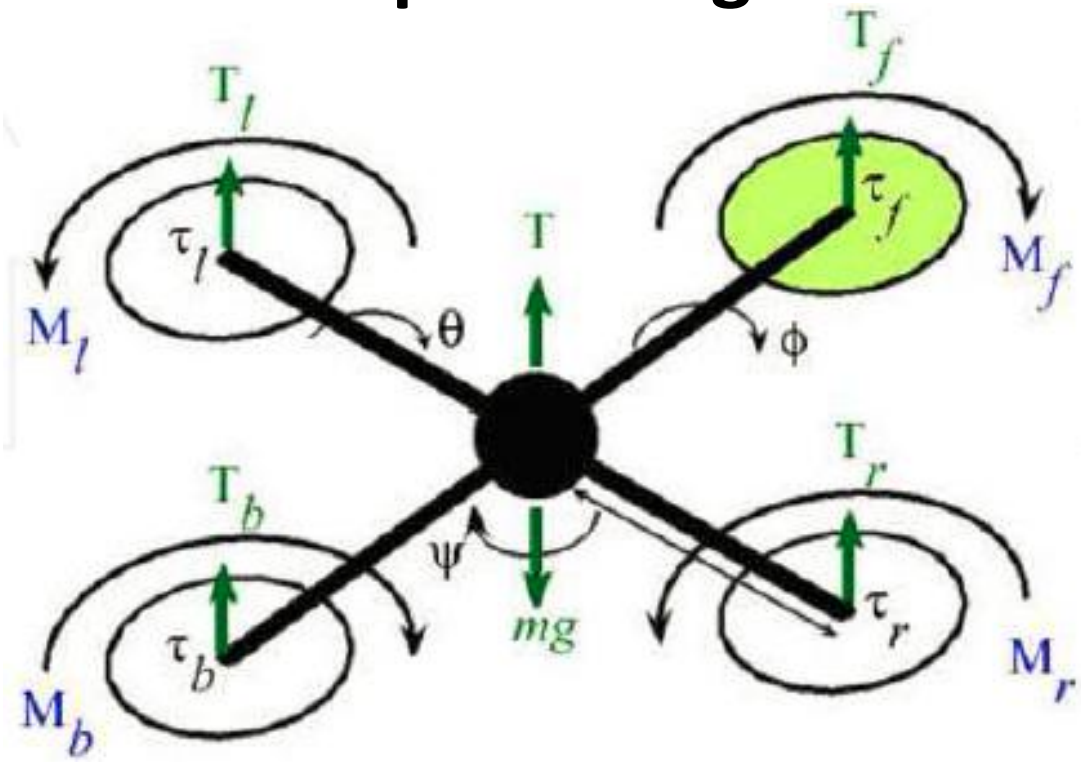
- Multirotor Flying Platform
  - Similar Flight Dynamics
  - Thrust Control via Pulse Width Modulation
  - Simple Implementation of Avionics
  - Rapid Development and Test Cycle
  - Relatively Inexpensive and Safe



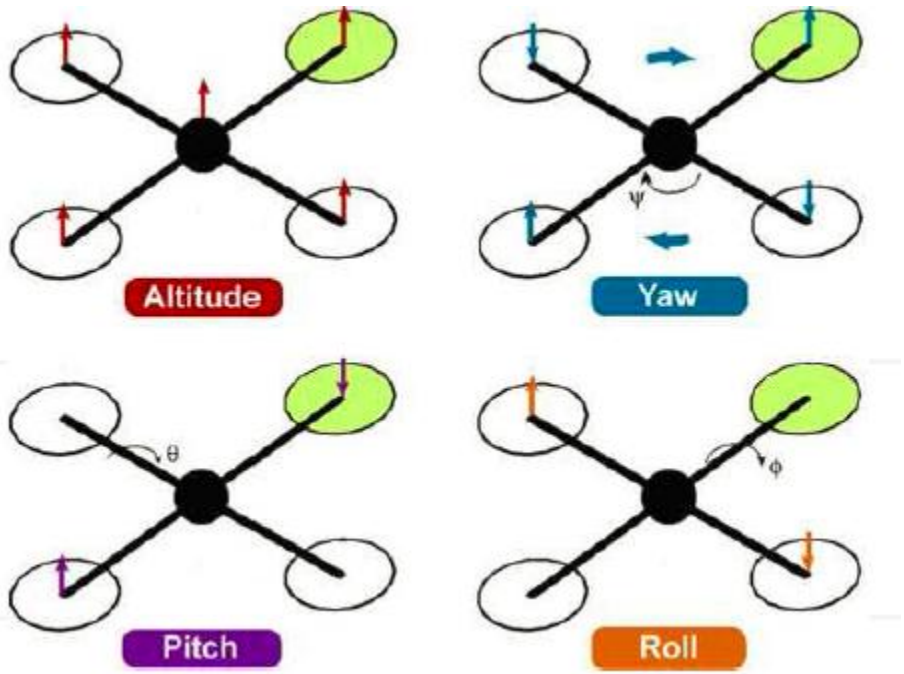
Flight Dynamics

Kinetics & Dynamics: FBD

Conceptual Diagram

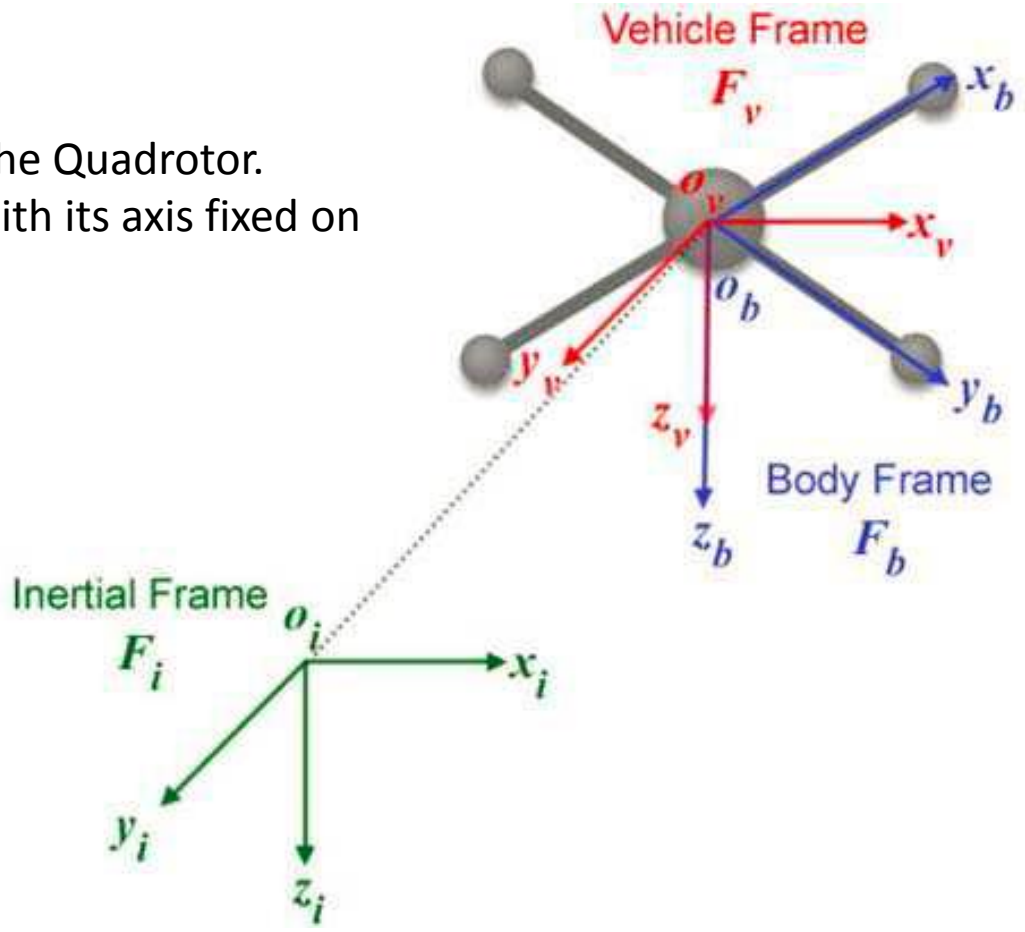


Quadrotor Dynamics



Kinetics & Dynamics: Frames of Reference

Frames of Reference:
Inertial Frame: Earth-fixed coordinate system
Vehicle Frame: Inertial frame located at the COG of the Quadrotor.
Body Frame: Located at the COG of the Quadrotor, with its axis fixed on the arms of the Quadrotor



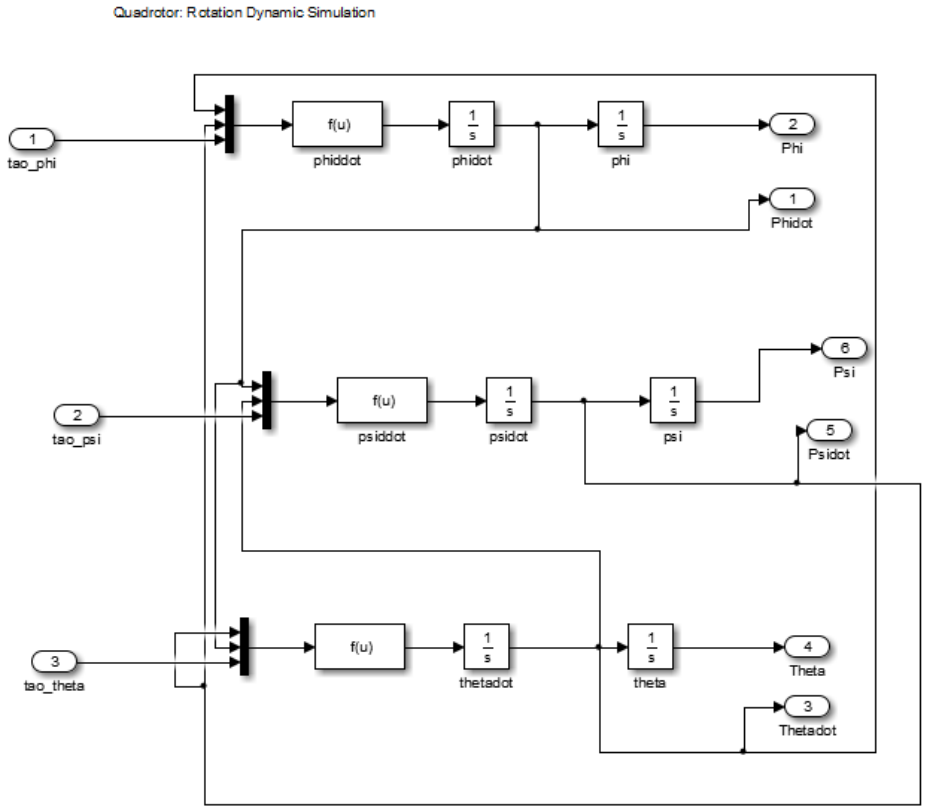
Kinetics & Dynamics: Equations of Motion

Translational Dynamics:

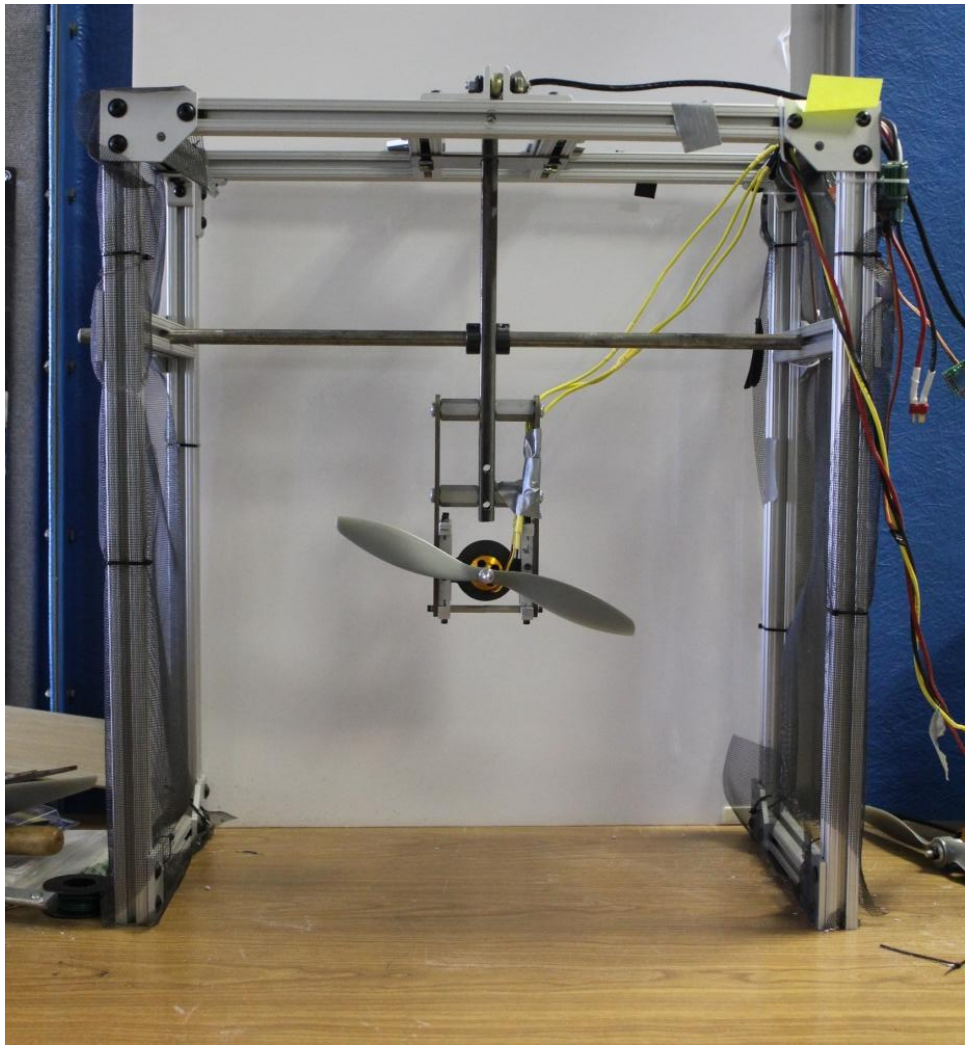
$$\begin{bmatrix} \ddot{P}_x \\ \ddot{P}_y \\ \ddot{P}_z \end{bmatrix} = \begin{bmatrix} \dot{\psi}\dot{P}_y - \dot{\theta}\dot{P}_z - g\sin\theta \\ \dot{\phi}\dot{P}_z - \dot{\psi}\dot{P}_x + g\cos\theta\sin\phi \\ \dot{\theta}\dot{P}_x - \dot{\phi}\dot{P}_y + g\cos\theta\cos\phi - \dfrac{f_z}{M} \end{bmatrix}$$

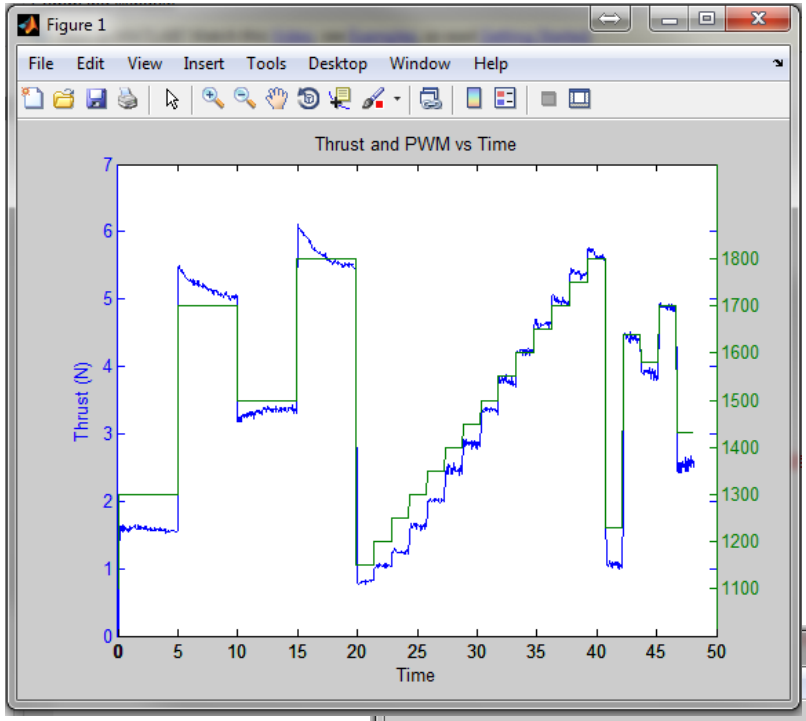
Rotational Dynamics:

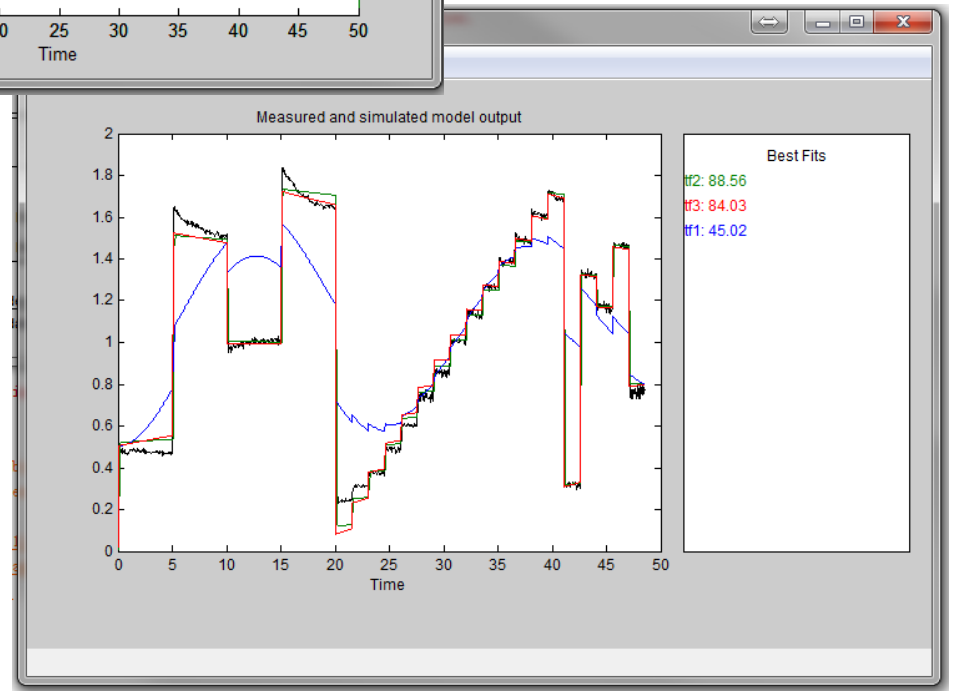
$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{j_y - j_z}{j_x}\dot{\theta}\dot{\psi} + \frac{1}{jx}\tau_\phi \\ \frac{j_z - j_x}{j_y}\dot{\phi}\dot{\psi} + \frac{1}{j_y}\tau_\theta \\ \frac{j_x - j_y}{j_z}\dot{\phi}\dot{\theta} + \frac{1}{j_z}\tau_\psi \end{bmatrix}$$



System Identification

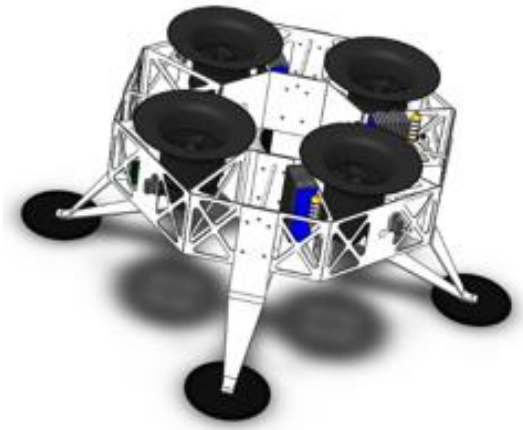






Hardware Design

Design Evolution



First Generation

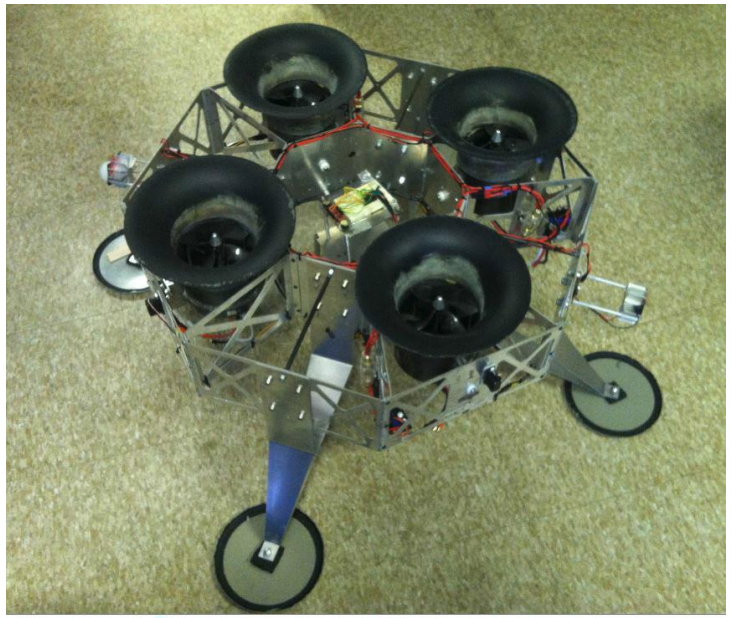
Second Generation

Third Generation
Half Scale Prototype

Design Evolution: First Generation Design

First Generation:

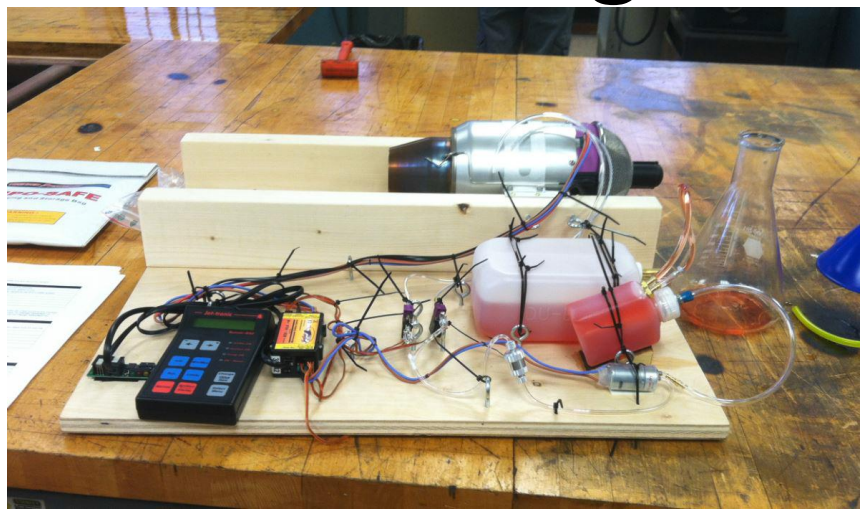
→ Configuration: Classic Quadrotor
→ Aluminum Construction
→ Titling Actuators (Thrust Vector Control)
→ Ducted fans (1.25: 1 Thrust to Weight Ratio)
→ Flexible Landing Gear
→ 4 X 37 Volt Lithium-Polymer Battery (Flight Time: ≈ 2 minutes)

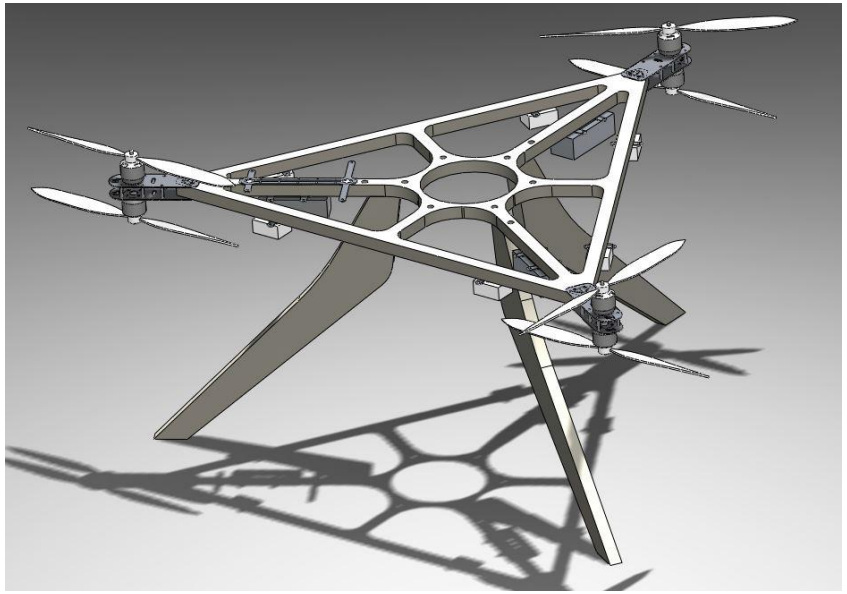


Design Evolution: Second Generation Design

Second Generation:

→ Configuration: Tricopter
→ Carbon Fiber, Glass Fiber and Foam Composite Material
→ Fixed Actuators (Attitude Change Control)
→ DC motors/propellers & Micro turbine (3:1 Thrust to Weight Ratio)
→ Rigid Landing Gear

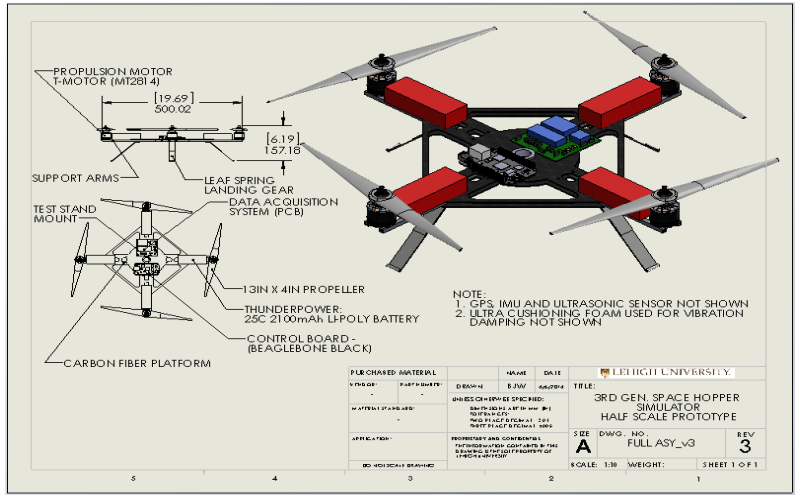


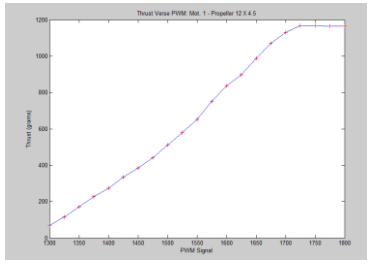


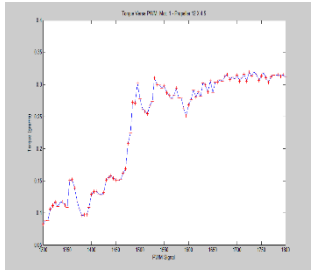
Third Generation: Quad+ Design

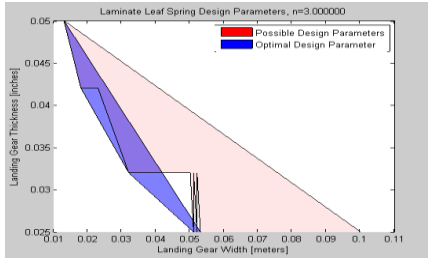
Design Specs:

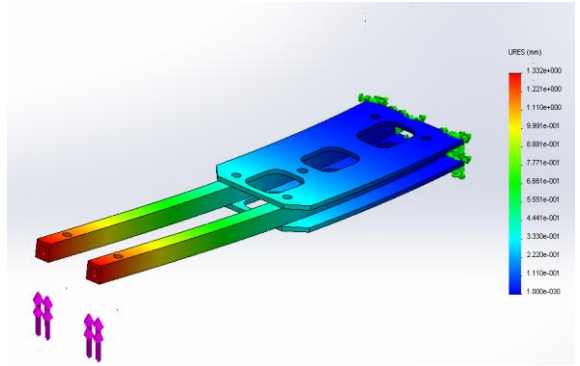
→ Configuration: Classic Quadrotor
→ Modular Carbon Fiber Construction
→ Fixed Actuators (Attitude Change Control)
→ DC motors/propellers (3:1 Thrust to Weight Ratio)
→ Flexible Landing Gear
→ Low Center of Gravity (Increase landing stability)

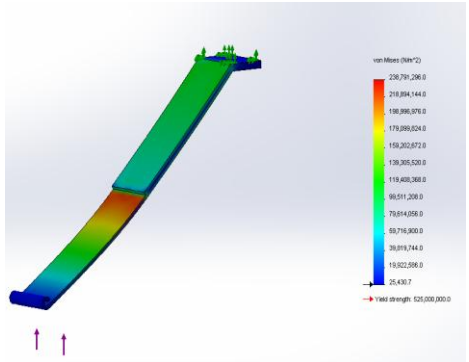


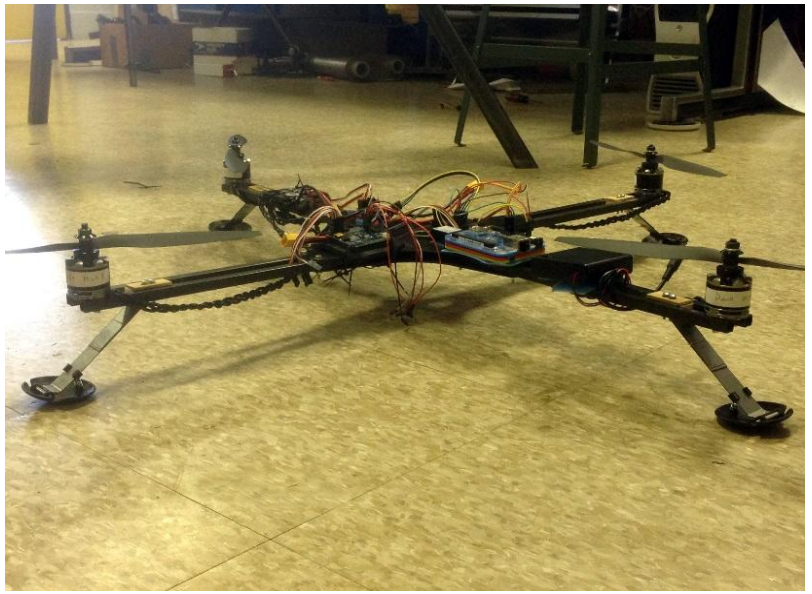




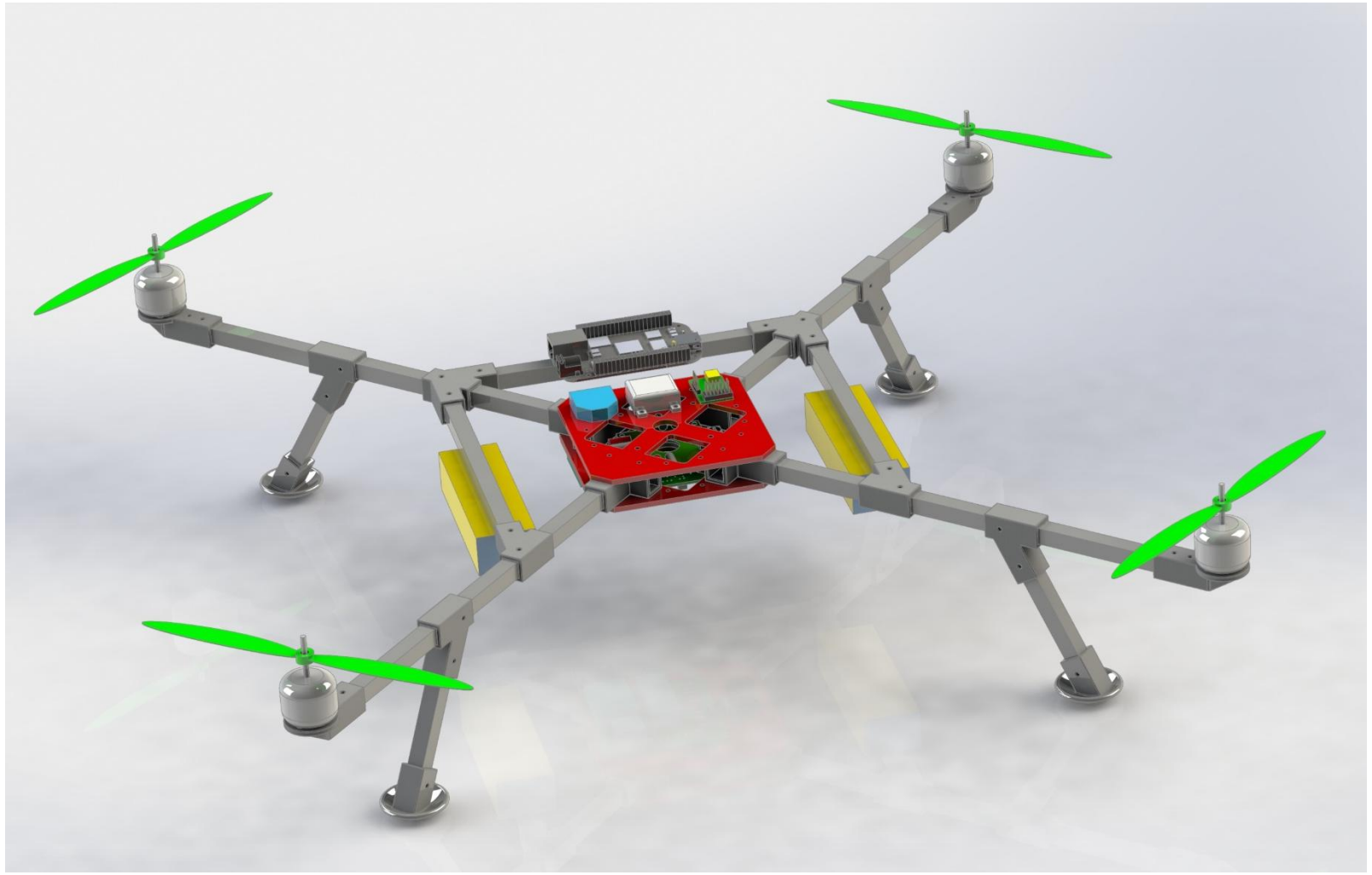








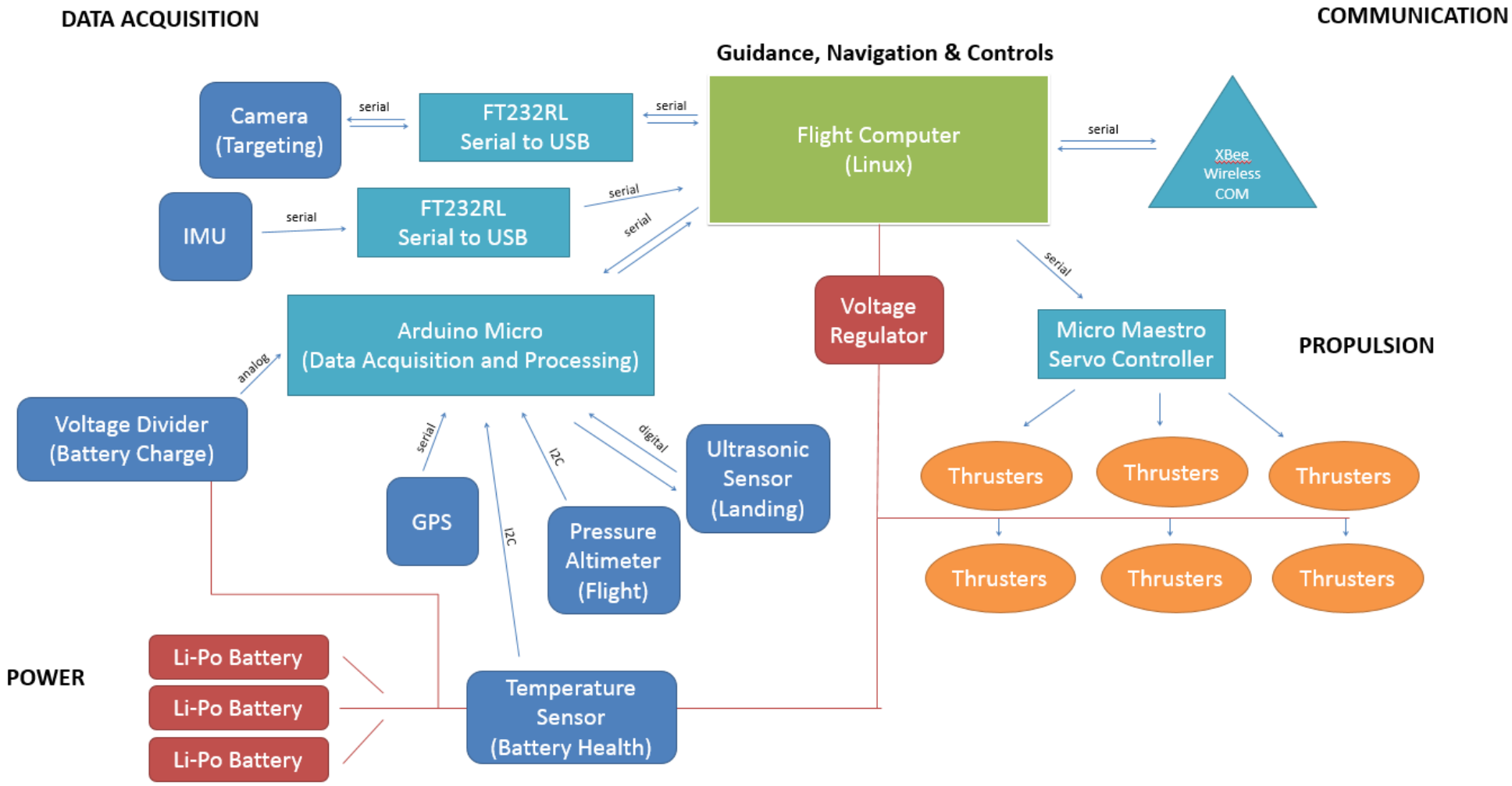
3rd Gen QuadX Hopper (in development)



- Hardware
  - X Flight Configuration
  - Acrylic / Wood Hub
  - Carbon Fiber Struts
  - Molded Plastic Connector
  - ~ 1500 kg
- Propulsion System
  - 750 Kv Motor
  - 11x4.5” Propellers
  - LiPo 4S - 5000 mAh
  - ~ 13 min Flight Time
- Avionics
  - Beaglebone Black
  - CH Robotics UM7 IMU
  - ArduDAQ
  - XBee Pro S1

Avionics

Avionics – Flight System Architecture

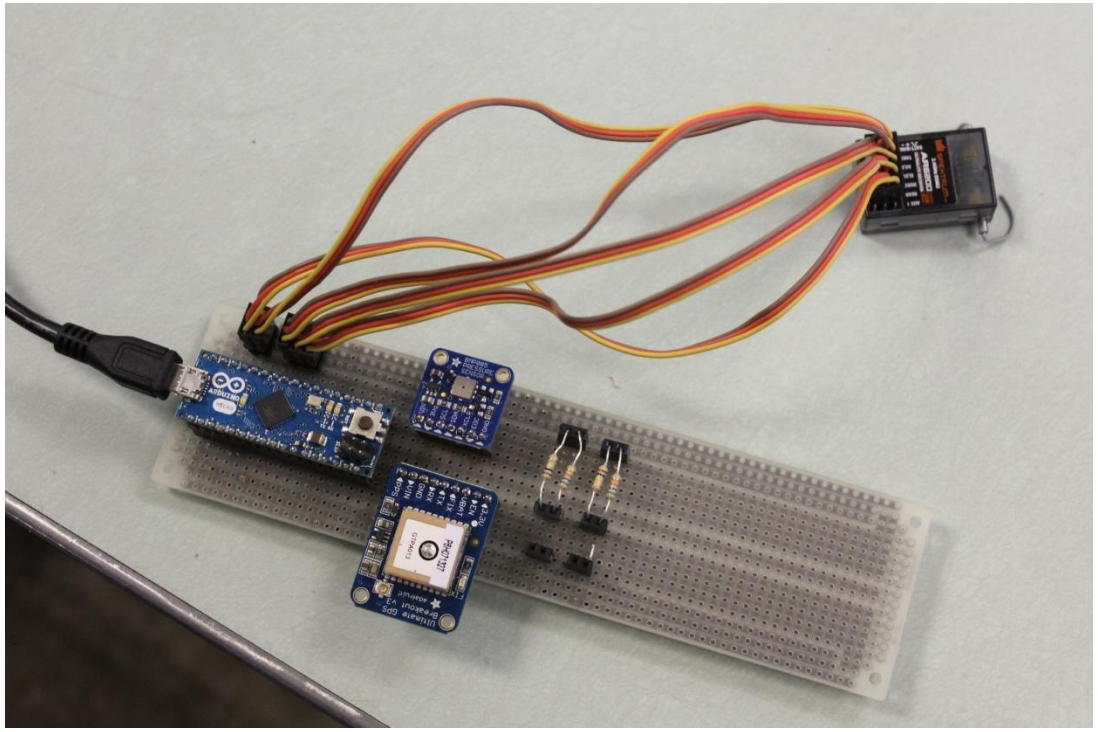


Flight Computer

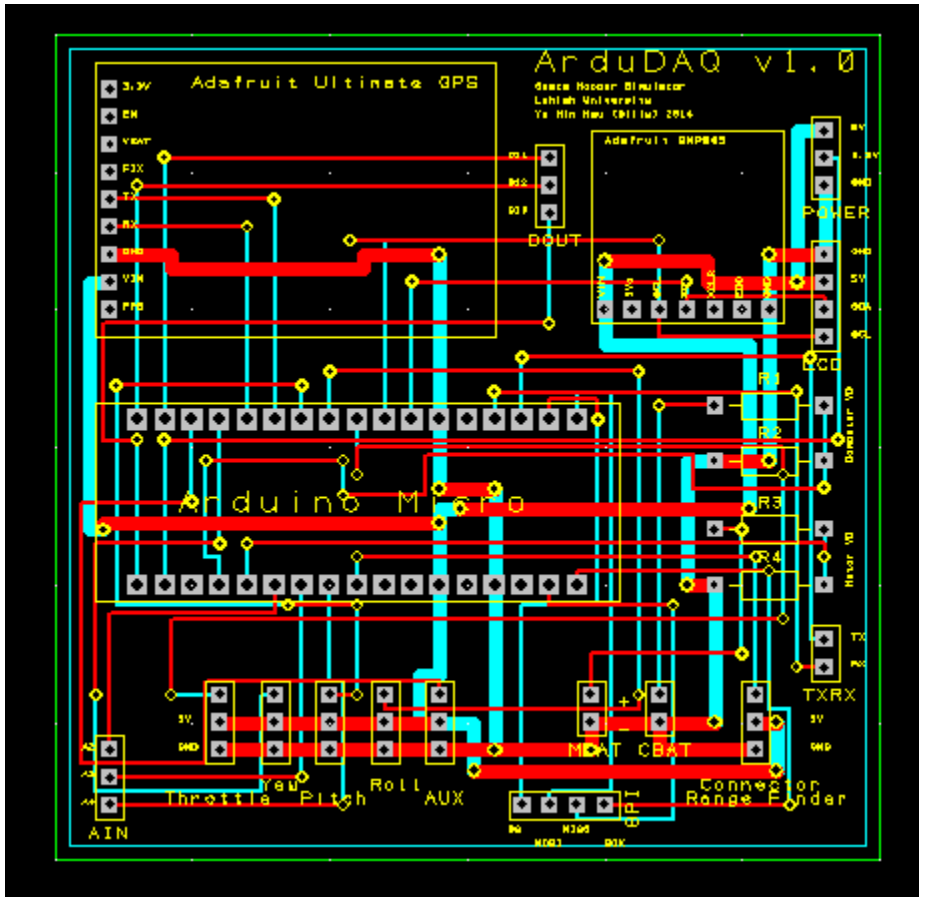
- BeagleBone Black
  - Linux Debian OS
  - ARM Corex-A8 1 GHz CPU
  - 4GB eMMC Flash Memory
  - 512 MB DDR3 RAM
  - 2 x 46 IO Pins – ADC/GPIO/PWM/I2C/SPI/UART
  - HDMI
  - $45

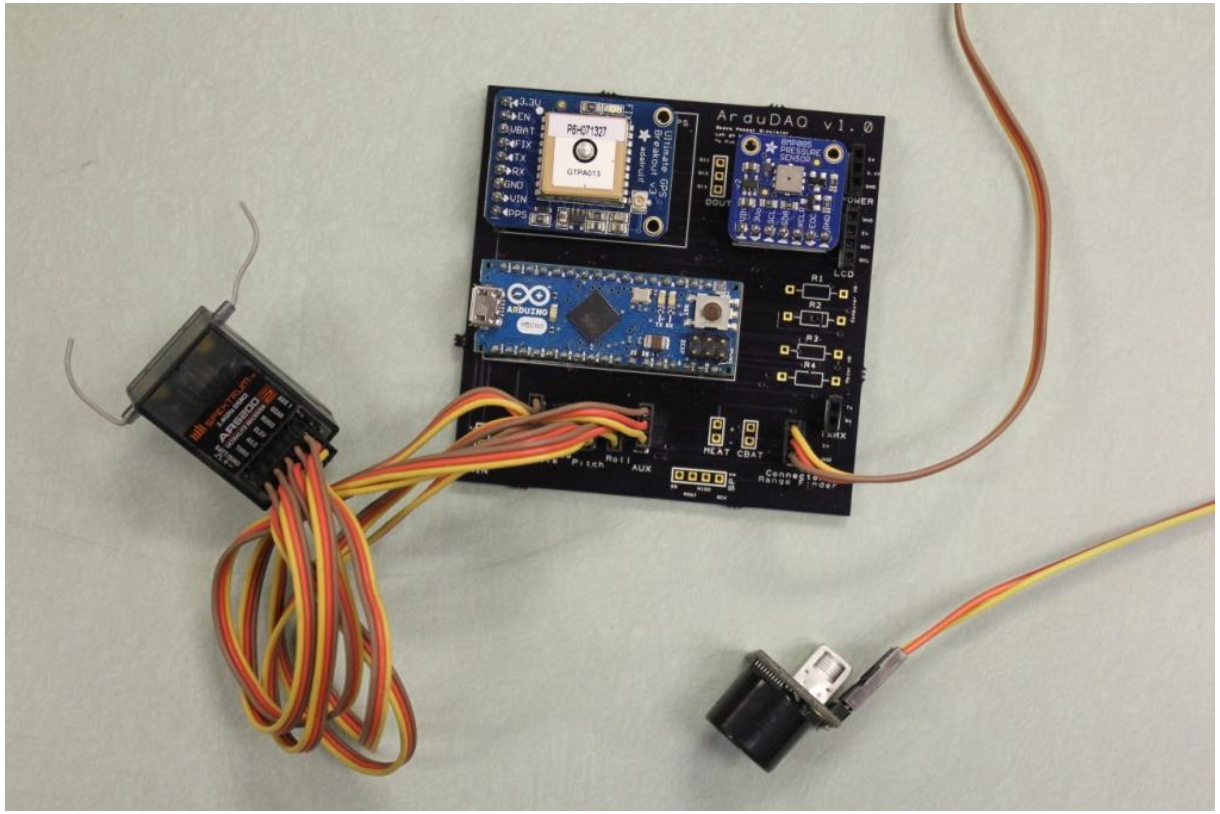
ArduDAQ

- Real-Time Flight Data Acquisition
  - Arduino Micro
  - GPS Receiver
  - RF Receiver
  - Altimeter
  - Ultrasonic Range Sensor
  - LiPo Voltage Sensor
  - Liquid Crystal Display
- Designed For Old Flight Computer without IO Pins

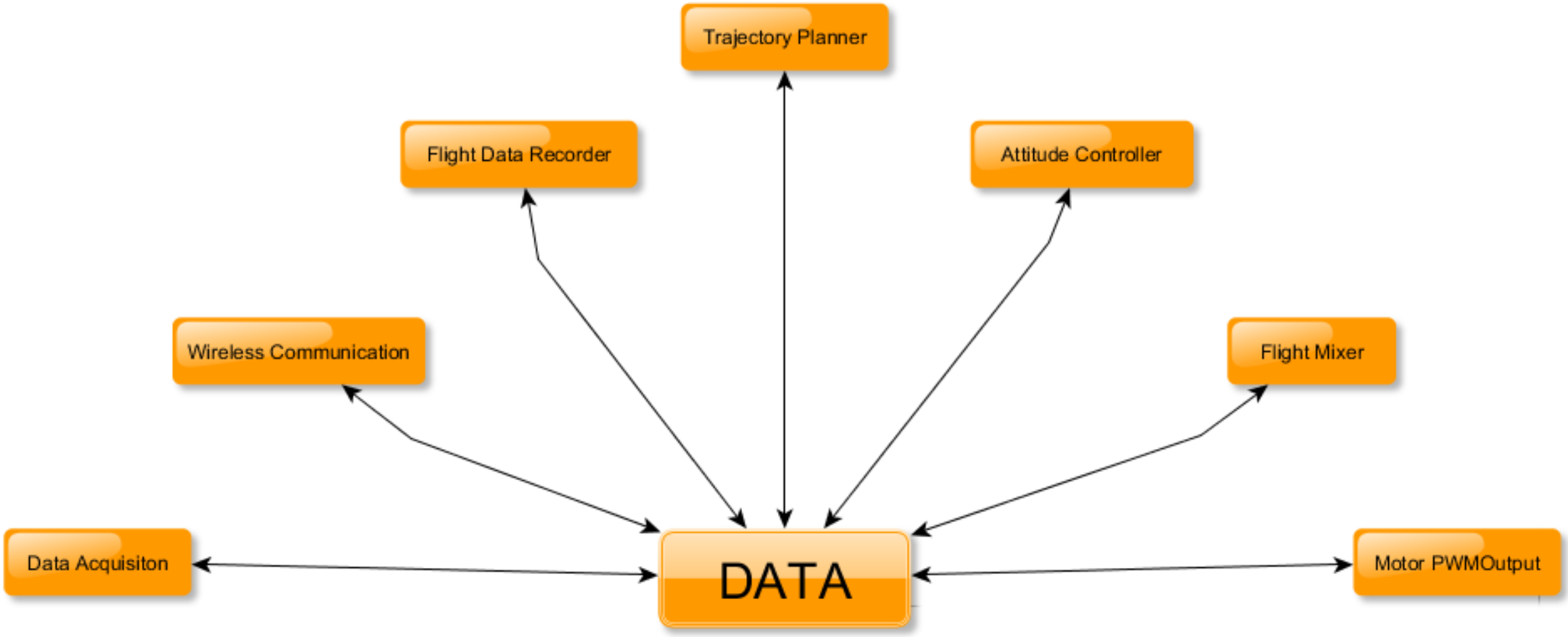


ArduDAQ PCB

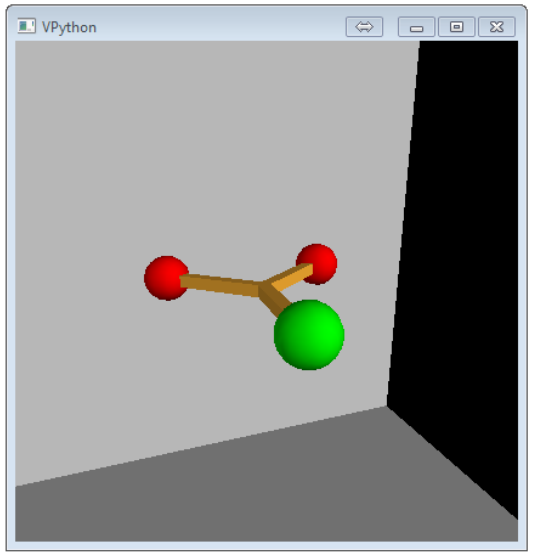

ArduDAQ v1.0
Adafruit Ultimate GPS
Adafruit BMP085
DOUT
POWER
LCD
R1
R2
R3
R4
Arduino Micro
TXRX
MEAT CBAT
Throttle
Yaw
Pitch
Roll
AUX
AIN
Connector
Range Finder

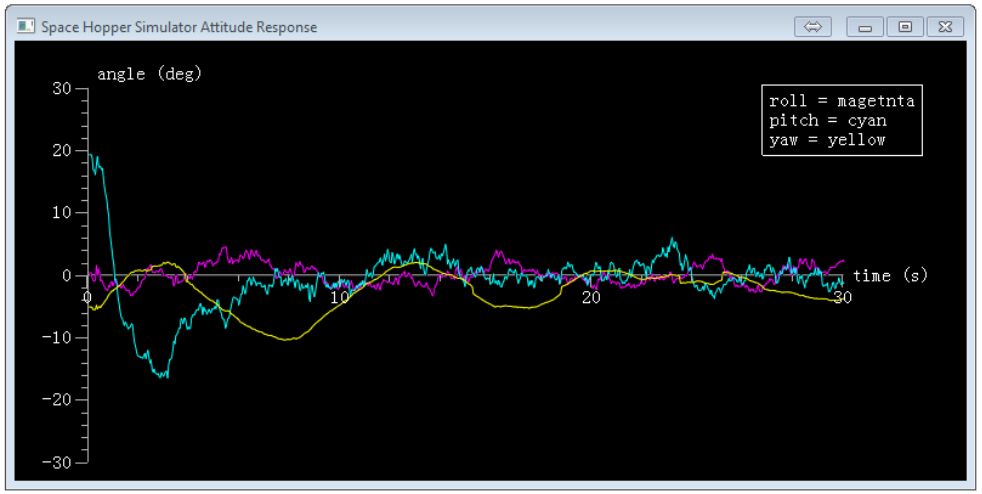

ArduDAQ v1.0

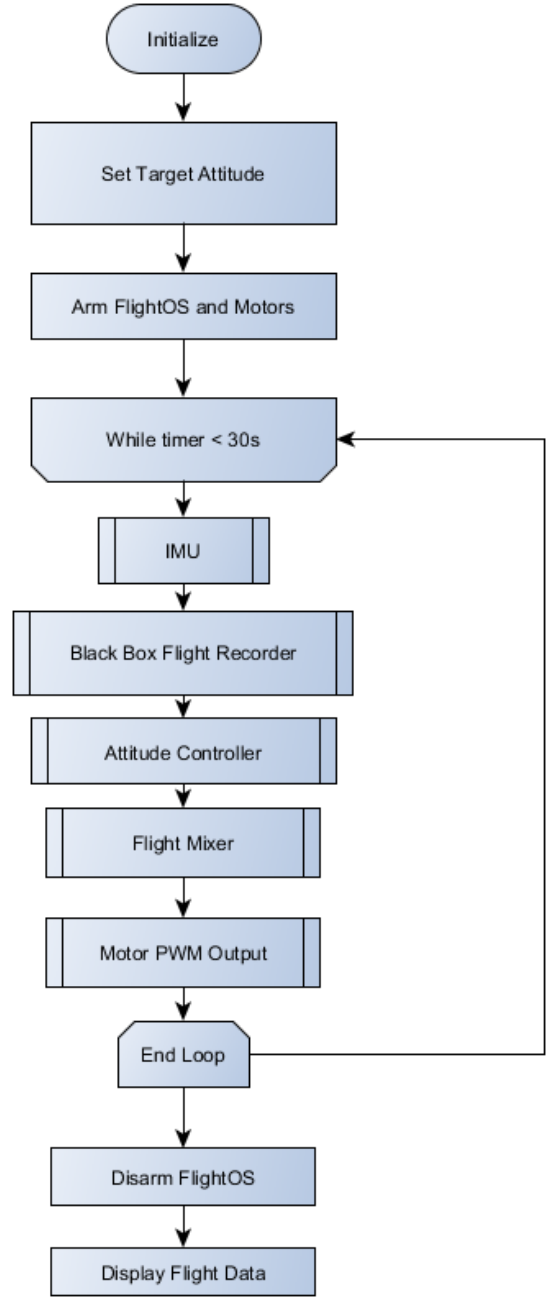
Software Modules



Rig Test Process Flow

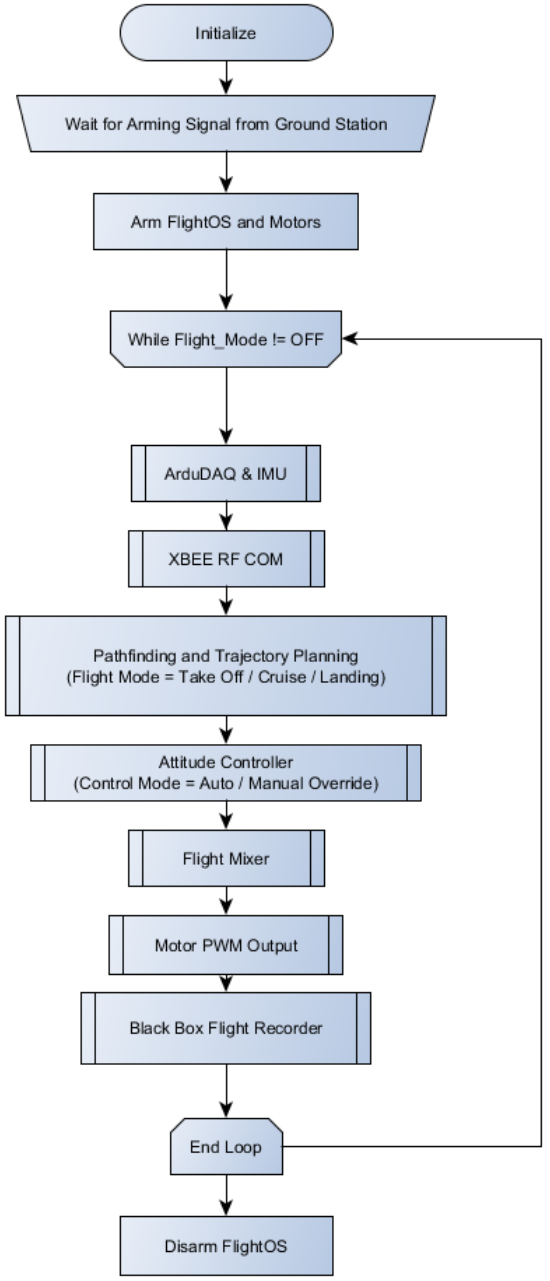




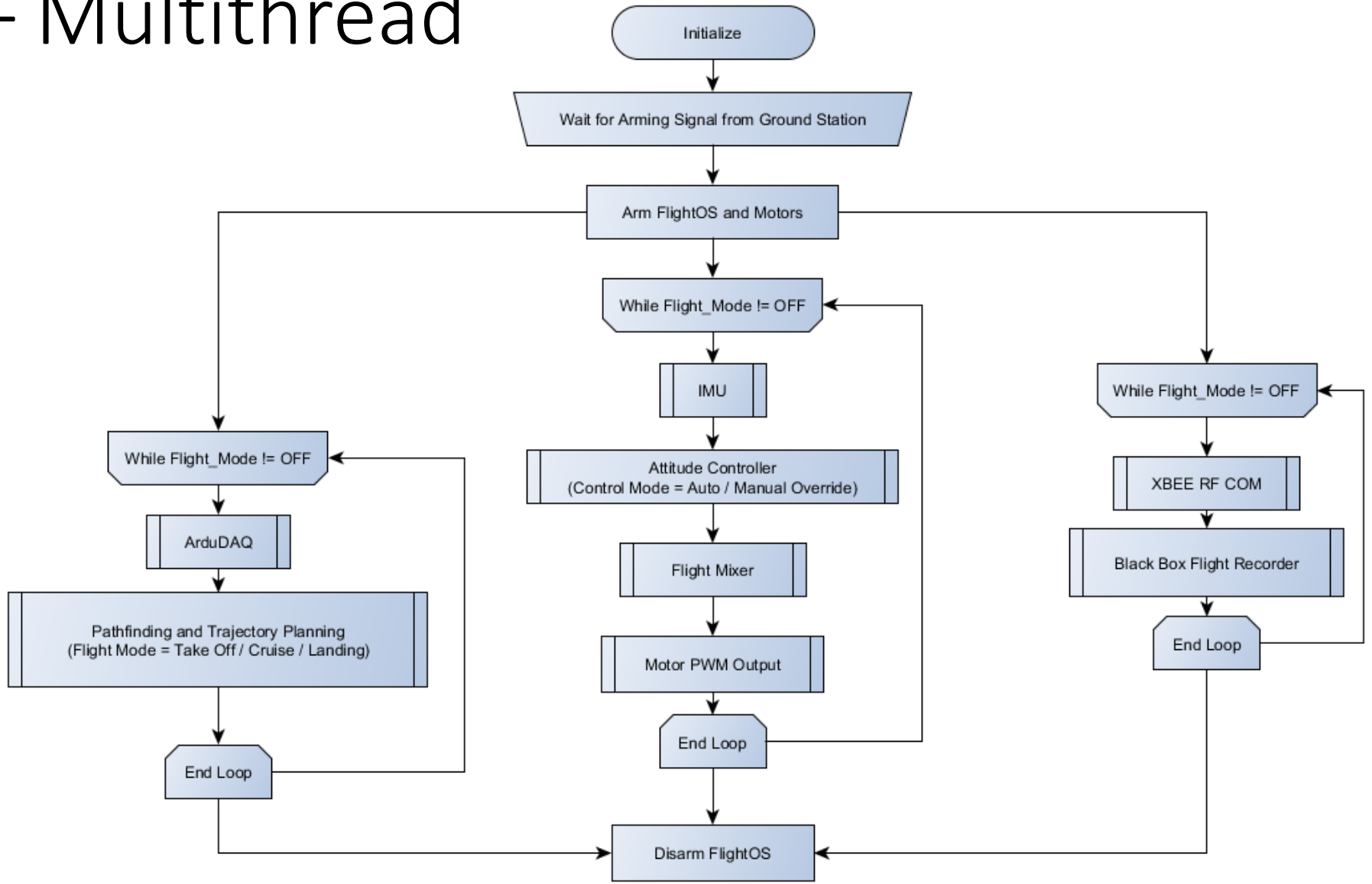


FlightOS

- Linear Processing
- Long Processing Time of Individual Module
  - Data Acquisition
  - Communication Interface
- Increased Delay in Control Loop



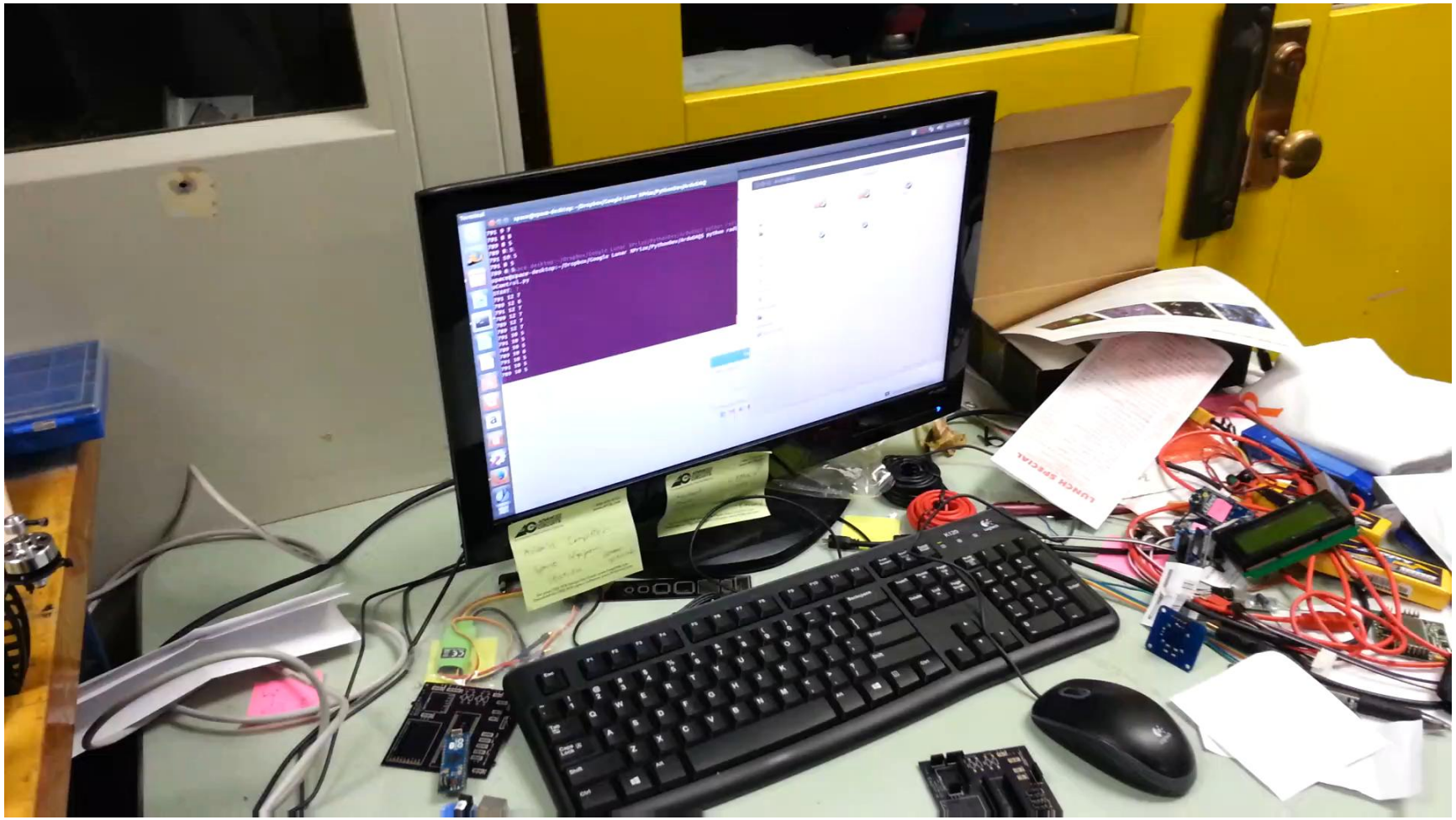
FlightOS – Multithread



Attitude Control System

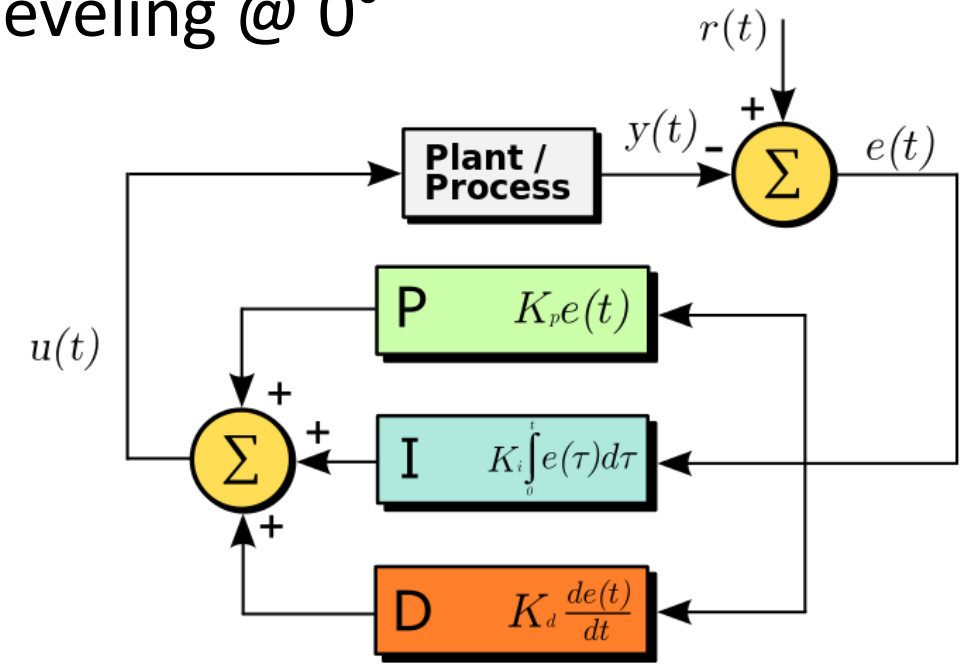
PID Controller

Why do we need a attitude controller?



PID Controller

- Proportional – Integral – Derivative Controller
- Compute Error from Input Signal and Sensor Data (IMU)
- Output Signal to Flight Mixer and Motors
- Example: flight leveling @ 0°



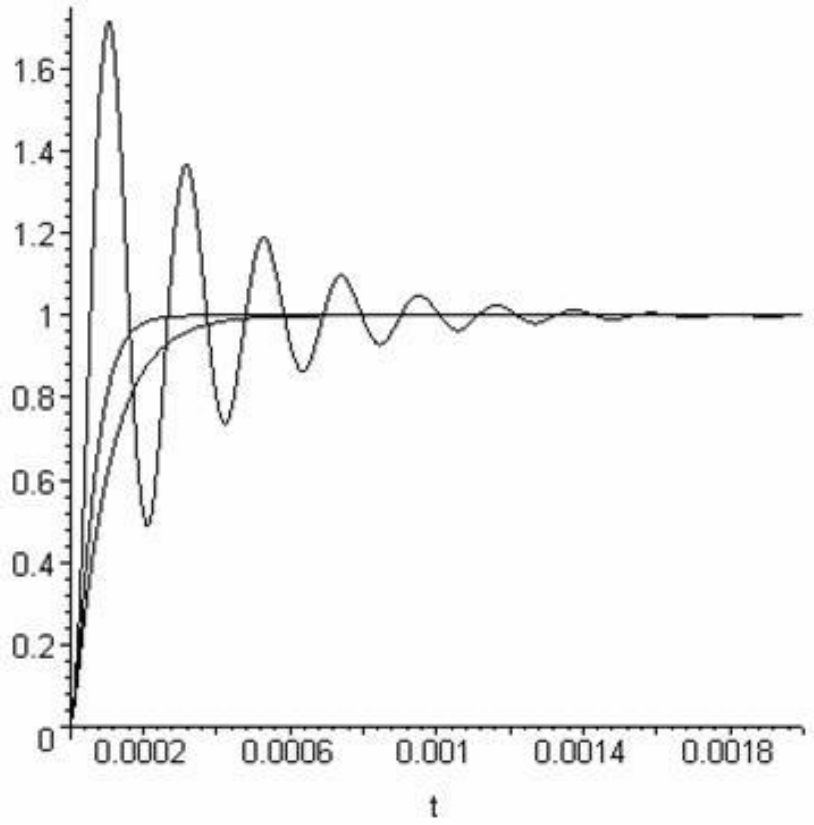
PID Gain Tuning

- Proportional Gain – drive output signal from error
- Derivative Gain – dampen and reduce oscillation
- Integral Gain – reduce steady state error

Challenges: overshoot, raise time, integral windup

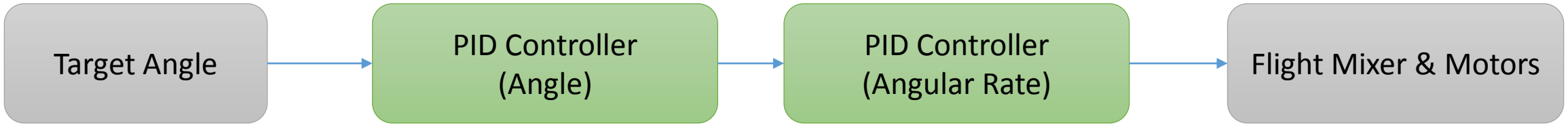
Manual Tuning Method

- Tune Kp till system begin to oscillate
- Tune Kd to reduce oscillation
- Tune Ki to reduce steady state error



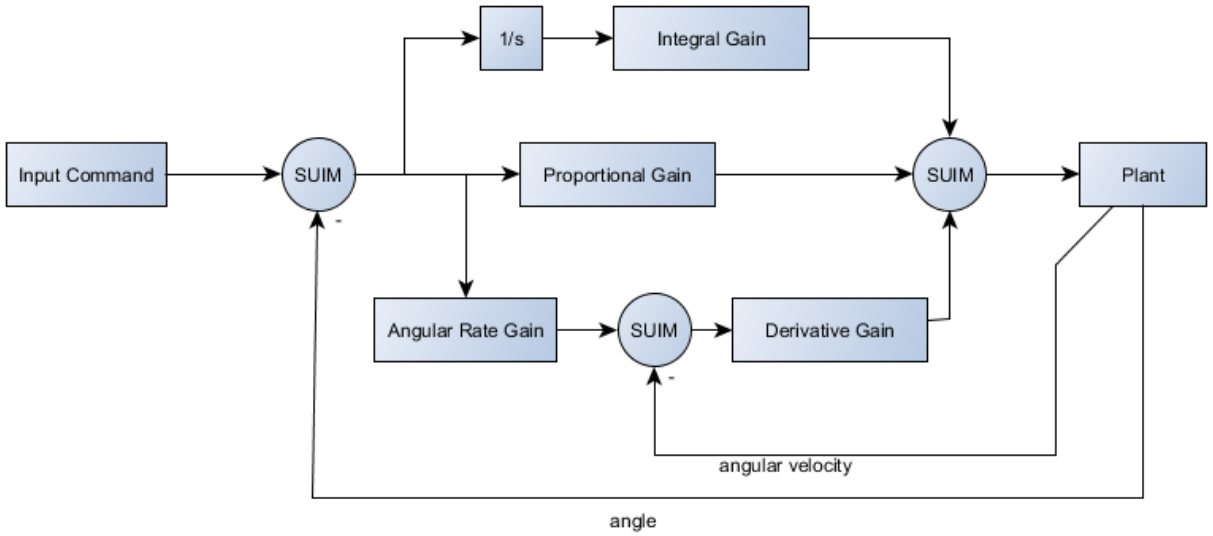
Cascade PID Controller

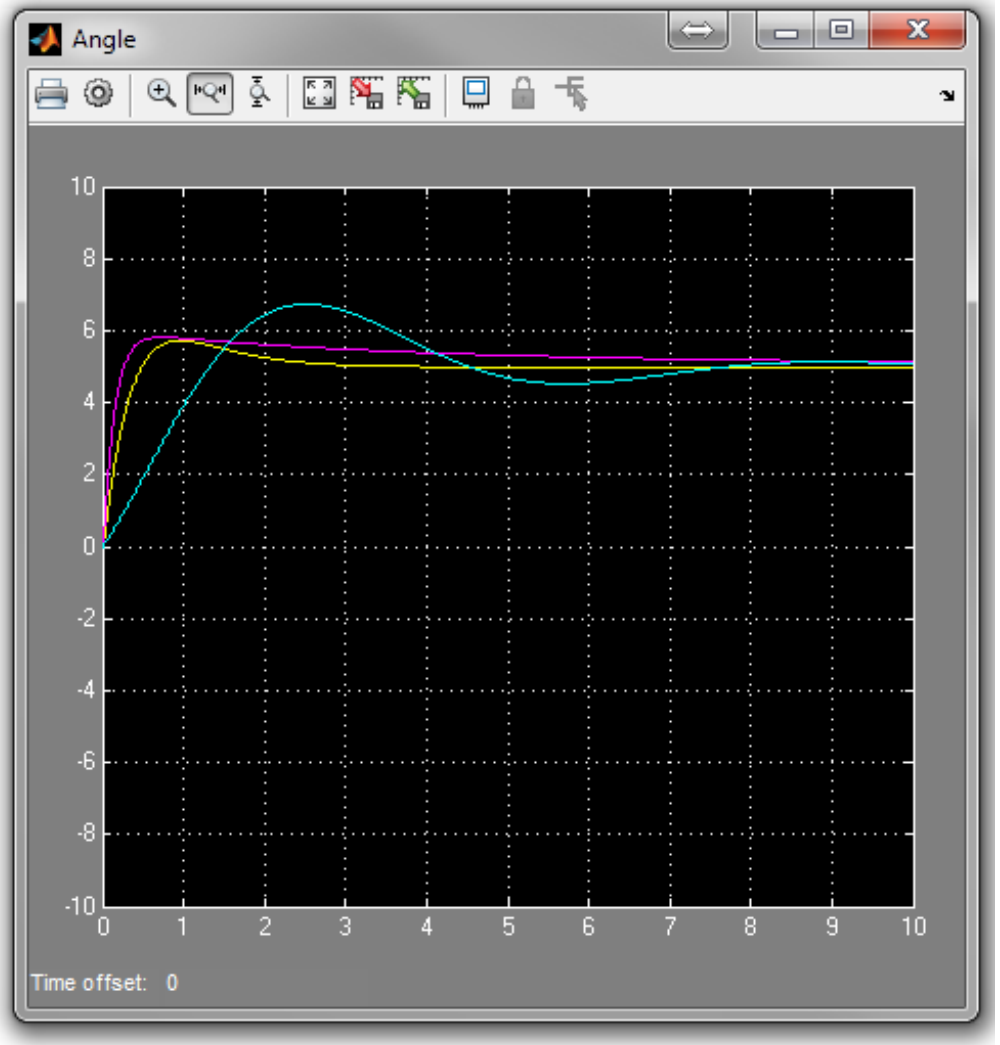
- Stacking PID controllers
- One controller control the set point of another
- Better performance



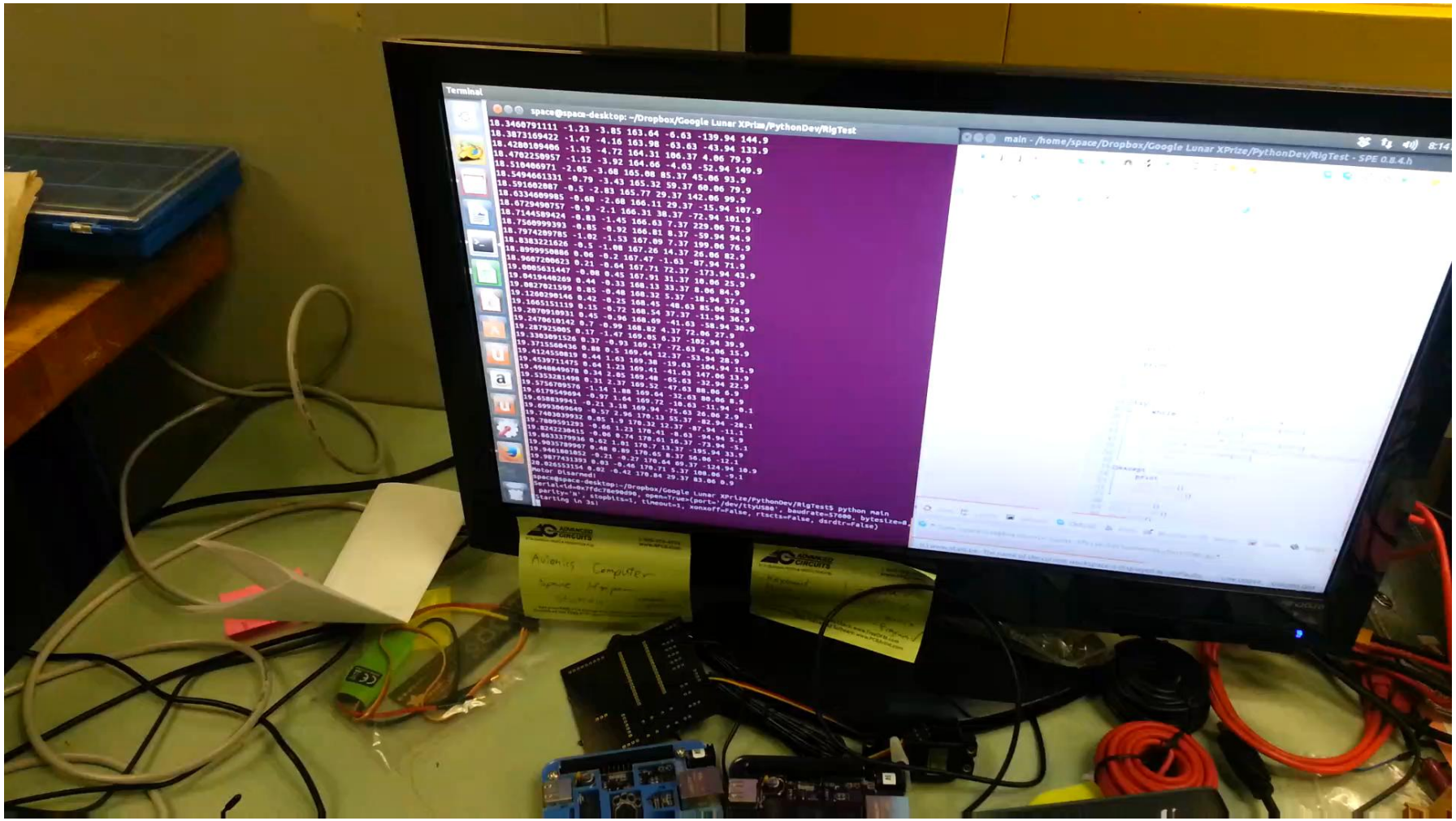
Modified Cascade PID Controller

- Cascade PI-P Attitude Controller
- Avoid Derivative Term due to Amplification of Noise

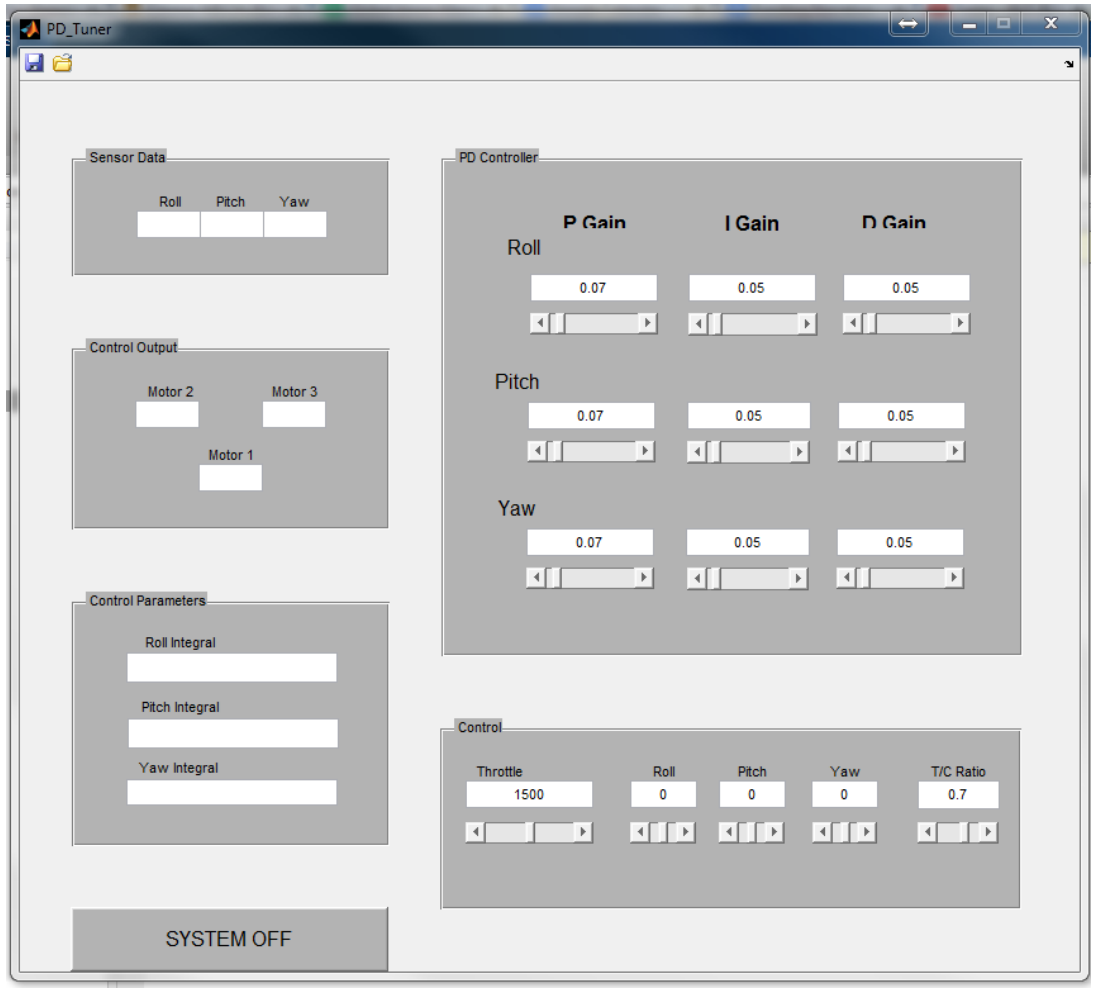


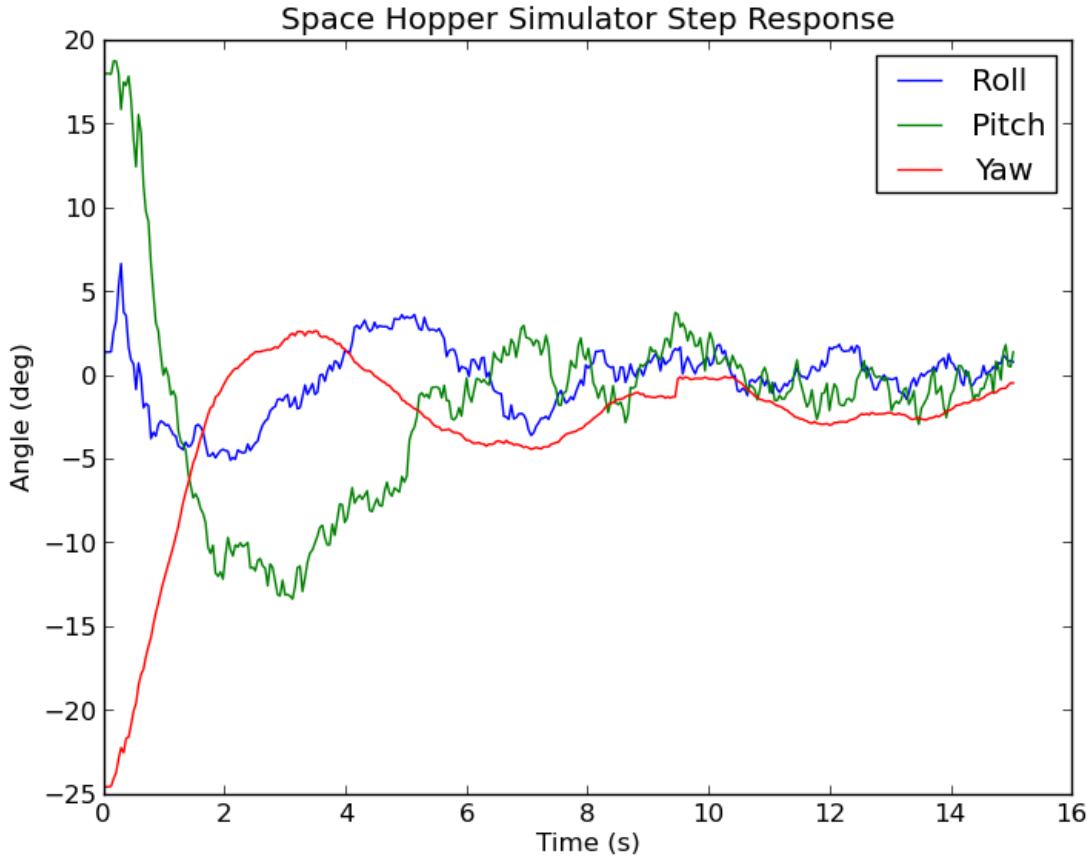


Rig Test with PID Roll and Pitch Controller



Rig Test and Live Gain Tuning



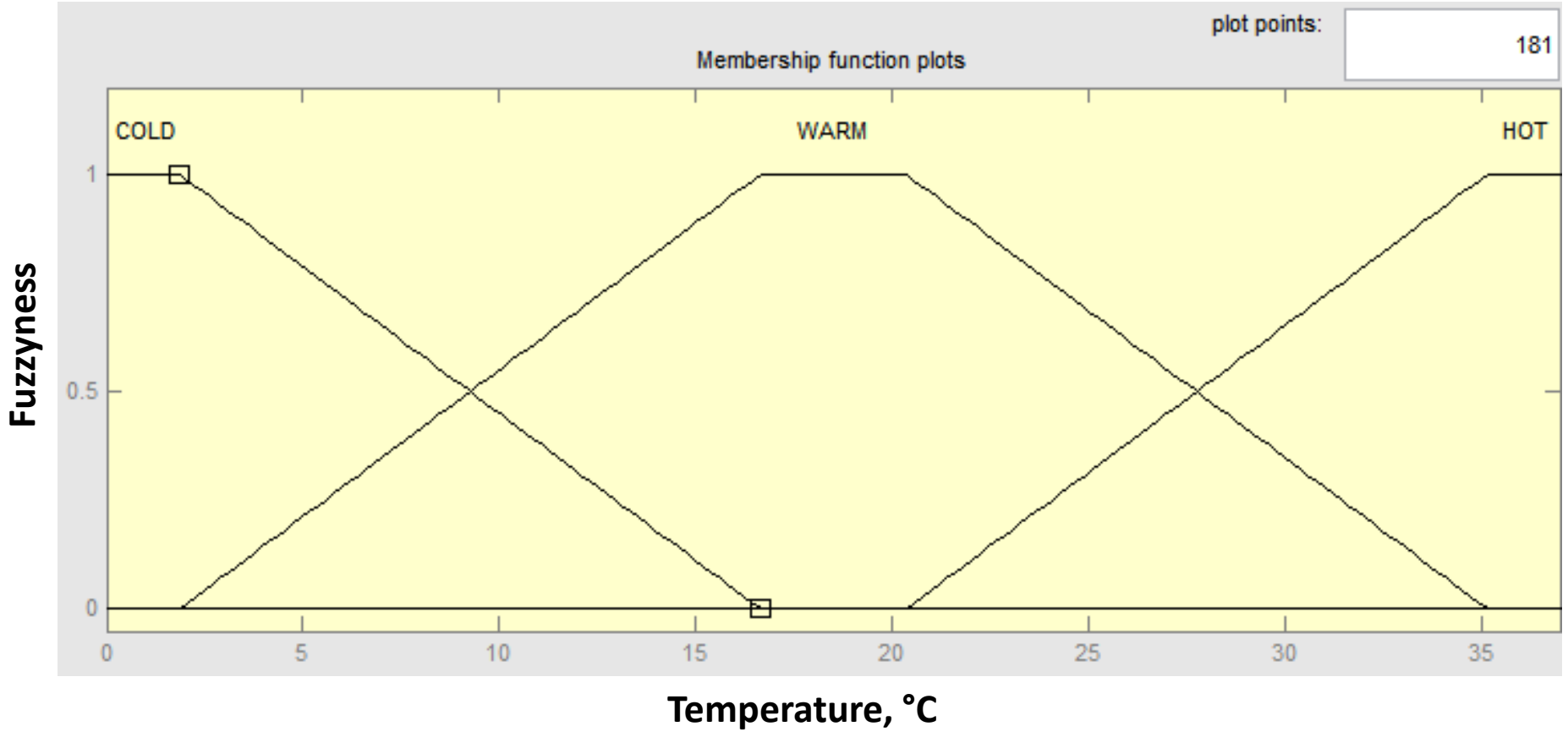


Attitude Control System

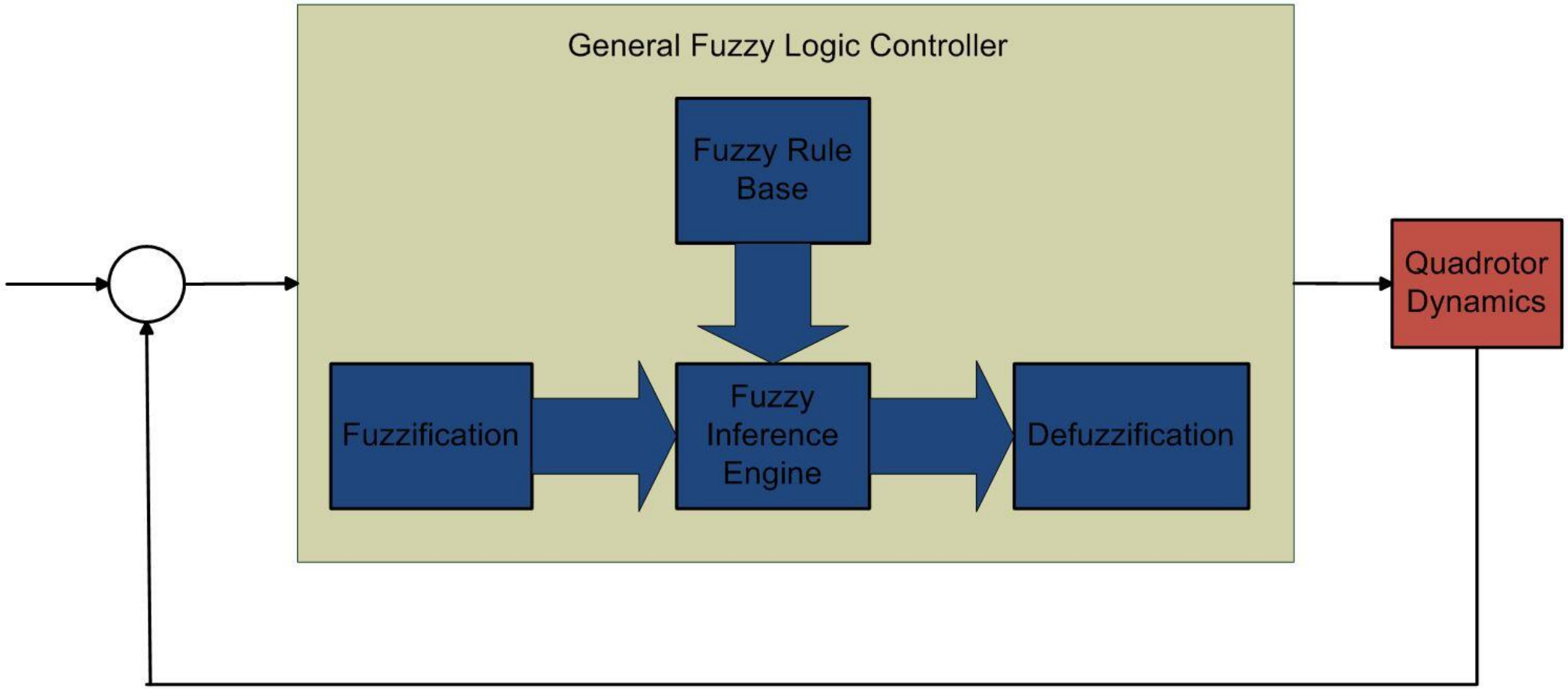
Fuzzy Logic Controller

Fuzzy Logic Control: What is Fuzzy Logic?

Fuzzy Set Theory was first proposed by Lofti A. Zadeh in 1965. Fuzzy Sets are sets whose elements have a degree of membership. This form of logic is refereed to as many valued logic; which deals with approximate reason verse conventional Digital logic reasoning (either completely true or completely false). Fuzzy Logic allows control engineers to design controllers from human (expert) experience, and requires no analytical understand of the systems dynamics.



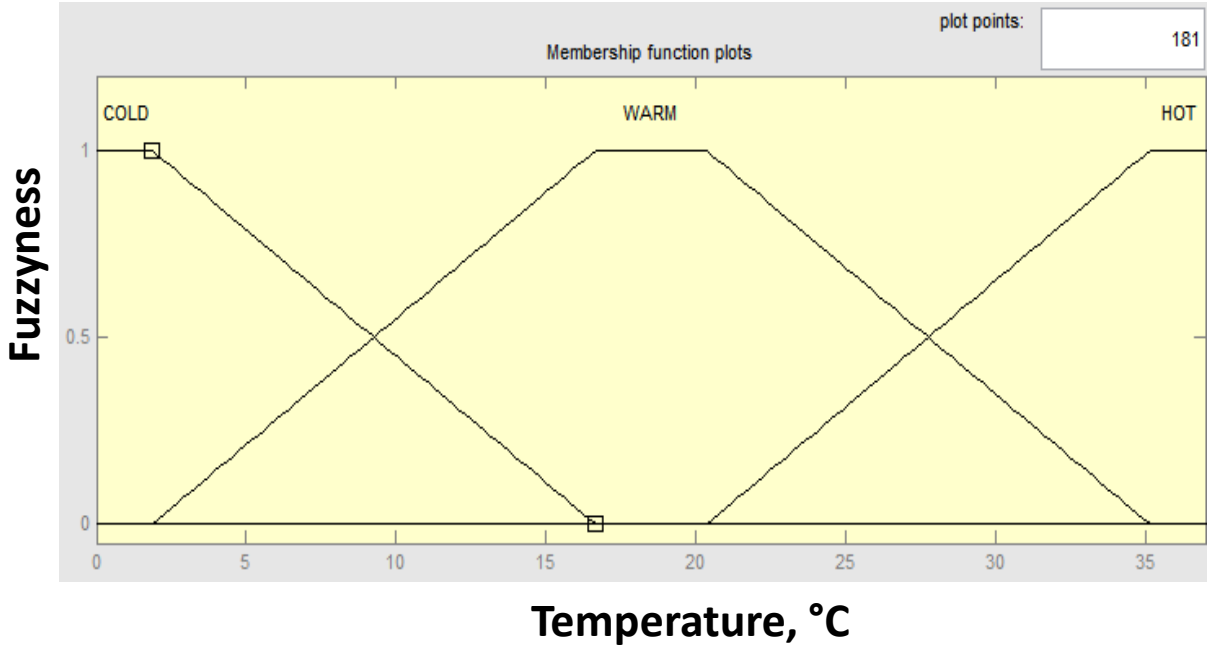
Fuzzy Logic Control: Basic Structure



Fuzzy Logic Control: Fuzzification

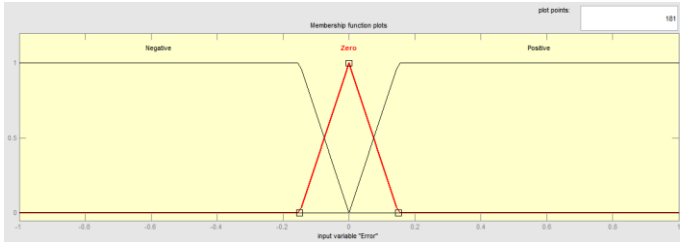
Temperature Example:

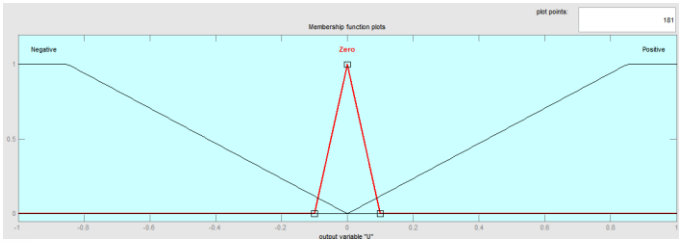
Crisp Temperature	Cold	Warm	Hot
0	1	0	0
15	.1	.75	0
37	0	0	1

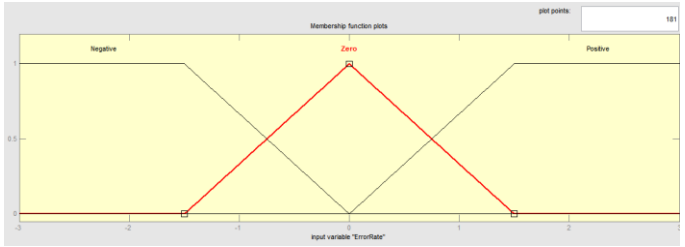


Fuzzy Logic Control: FIS (Mamdani)

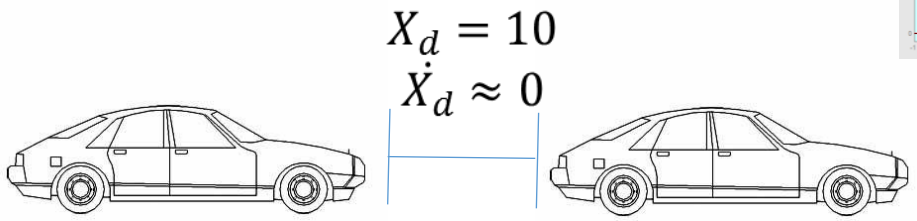
Fuzzy Inference System: Is a way of mapping an input space to and output space using fuzzy logic. FIS uses a collection of membership functions and a rule base to reason data. This methodology is very similar to how humans interpret and reason data.







Car Example:



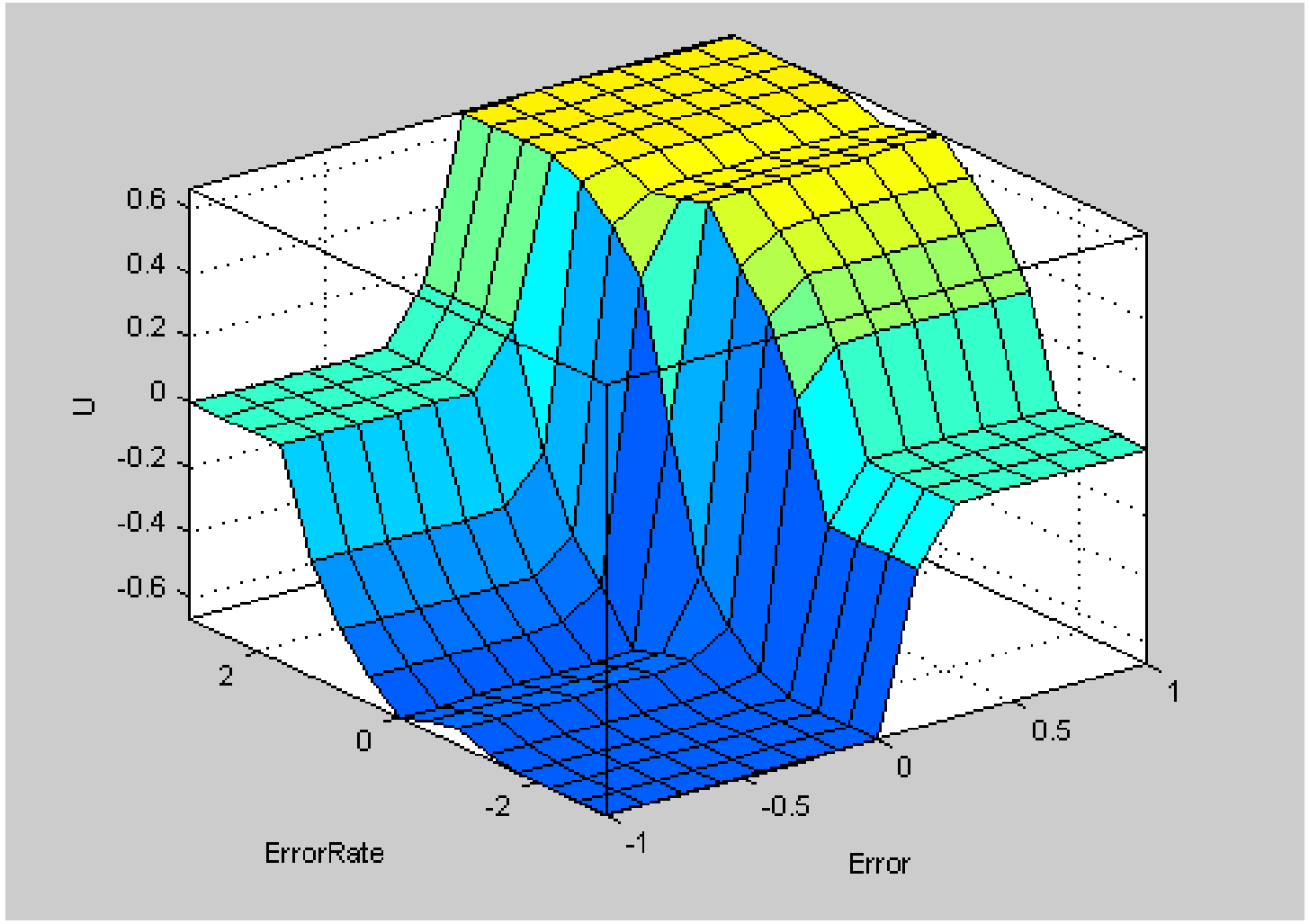
$$e = 10 - X_a$$
$$\dot{e} = 0 - \dot{X}_a$$

Scenario 1: Car B slowly accelerates
Linguistic Reasoning (if, and, then)
If e is negative small *and* $\dot{e}$ is zero *then accelerate positive small*

Scenario 2: Car B quickly decelerates
Linguistic Reasoning (if, and, then)
If e is positive small *and* $\dot{e}$ is negative large *then accelerate negative large*

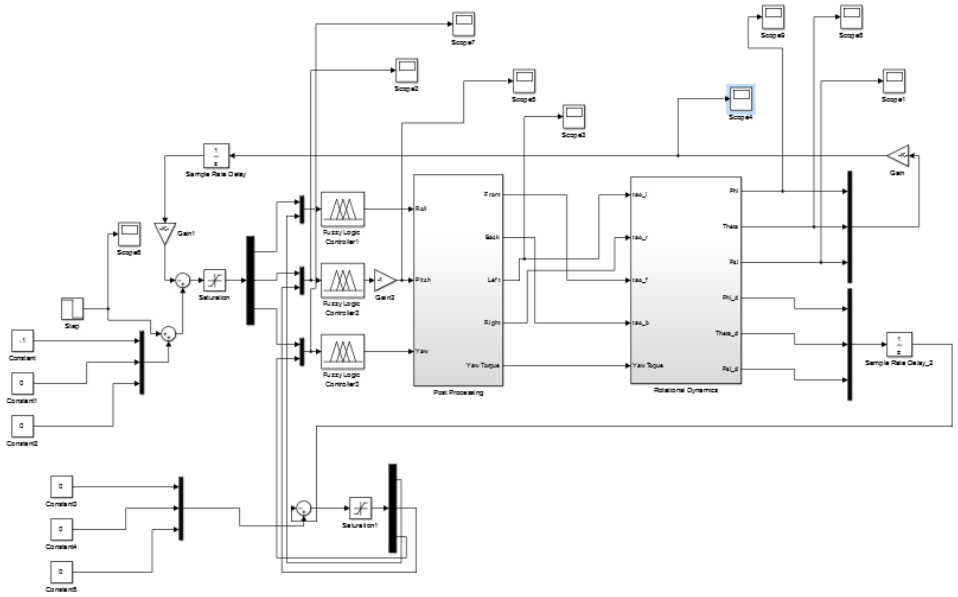
Fuzzy Logic Control: Defuzzification

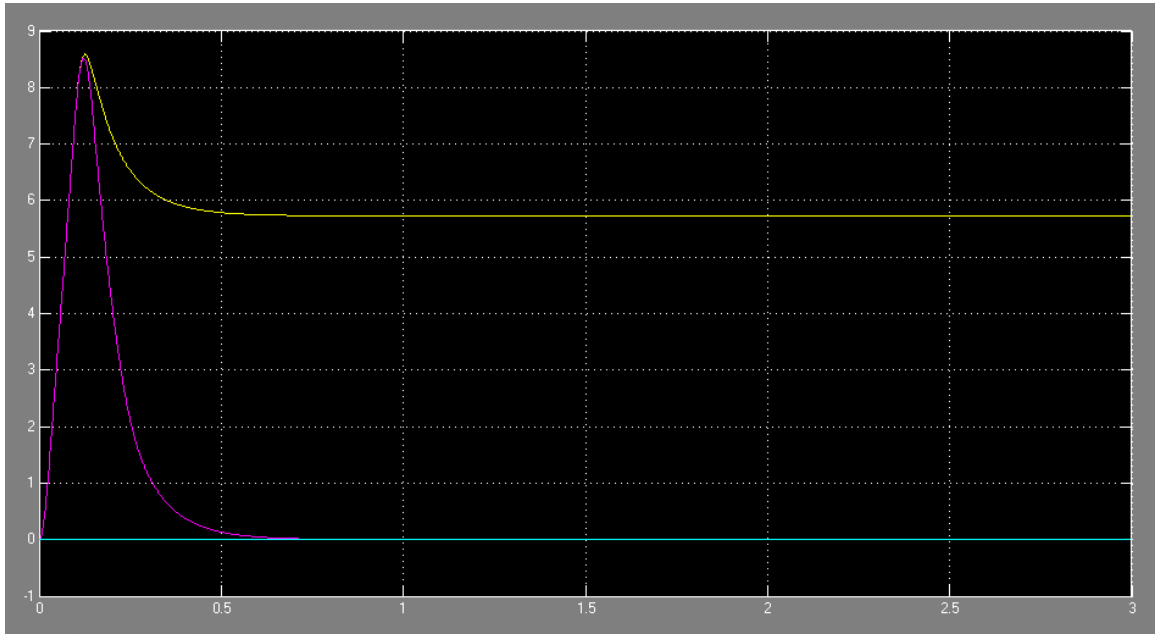
Defuzzification: A method used to map the fuzzy output onto a crisp output



Fuzzy Logic Control: Attitude Control

Attitude Control (P-D) Simulation:





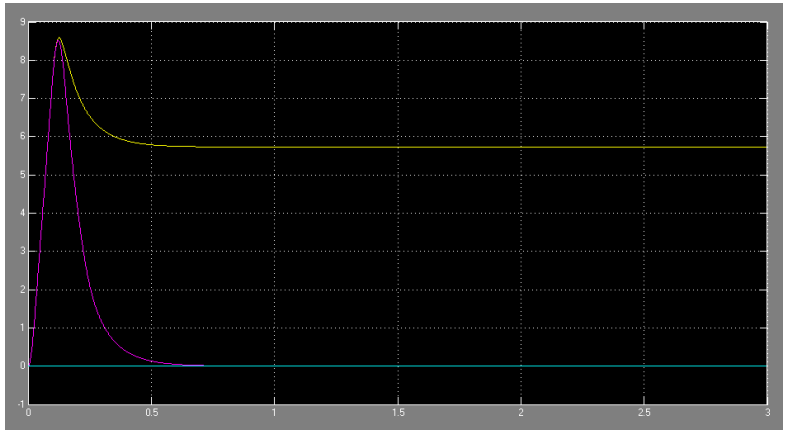
Fuzzy Logic Control: Attitude Control

Attitude Control (P-D) Test Results:

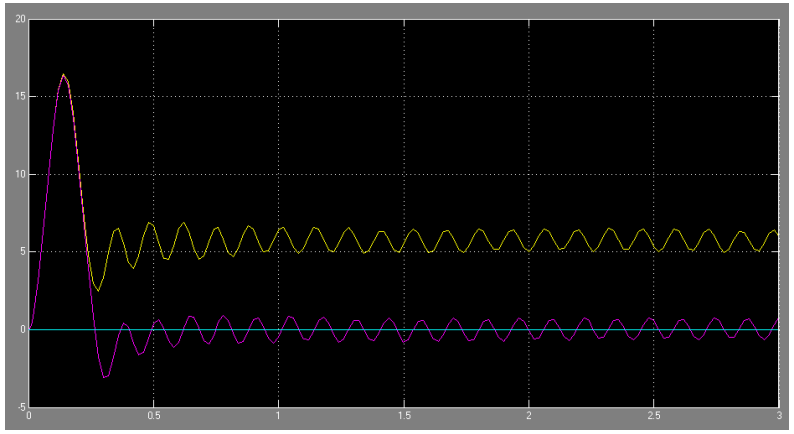
What Went Wong?

Fuzzy Logic Control: Attitude Control

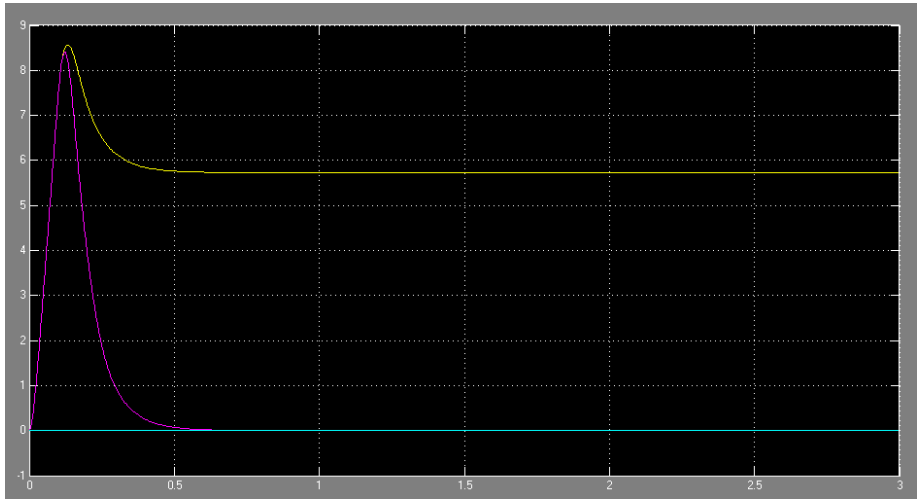
Question: What went wrong?



No Sampling Rate Delay



25hz Sampling Rate Delay



200hz Sampling Rate Delay

Future Works

Software Optimization

- Bottleneck – serial communication
- Reduce packet length by sending binary data (not too effective)

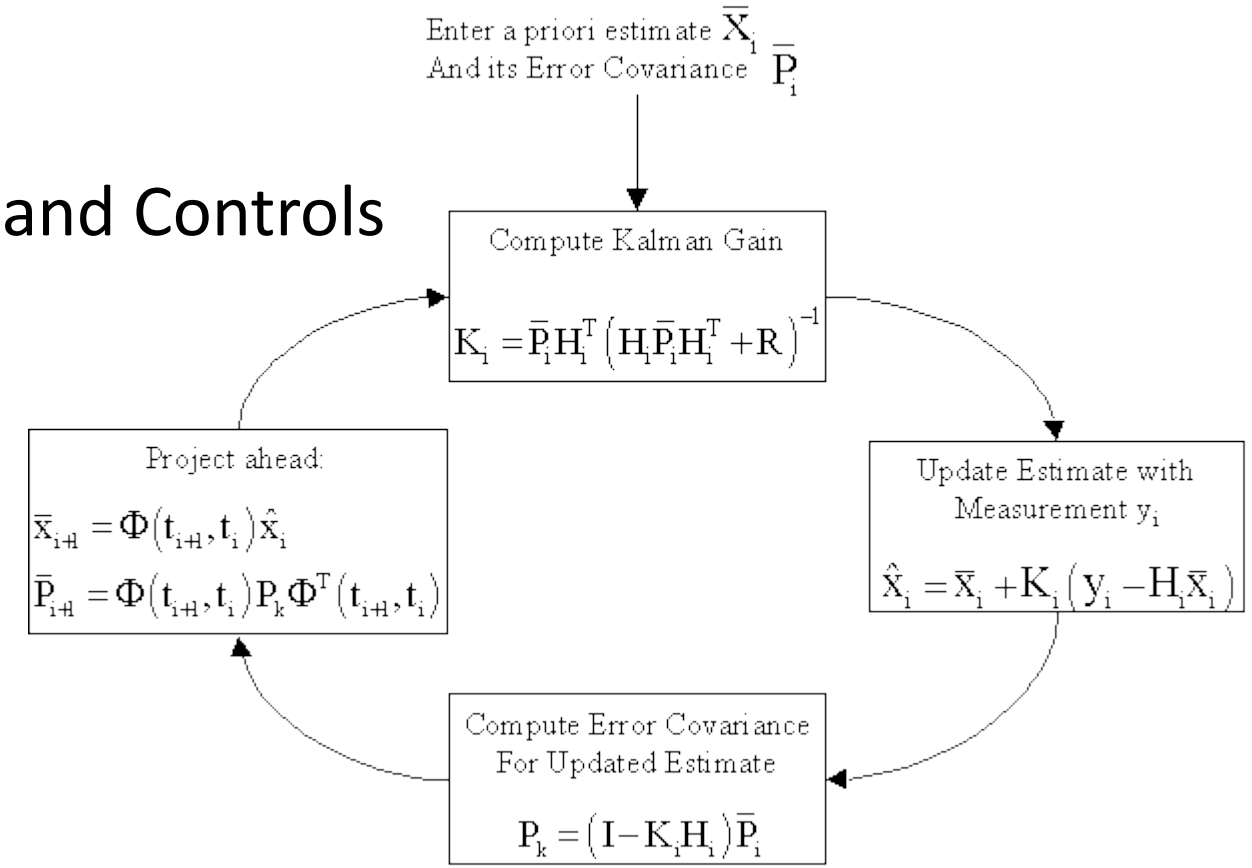
Comparison of Programming Languages and IMU Data Retrieval Benchmark

	MATLAB	Python	C#	Java	C++
BBB	No	Yes	No	Yes	Yes
Cross-Platform	Limited	Yes	Limited	Yes	Limited
Implementation	Easy	Easy	Moderate	Moderate	Difficult
Speed	Slow**	105 Hz*	210 Hz*	190 Hz*	Fastest **

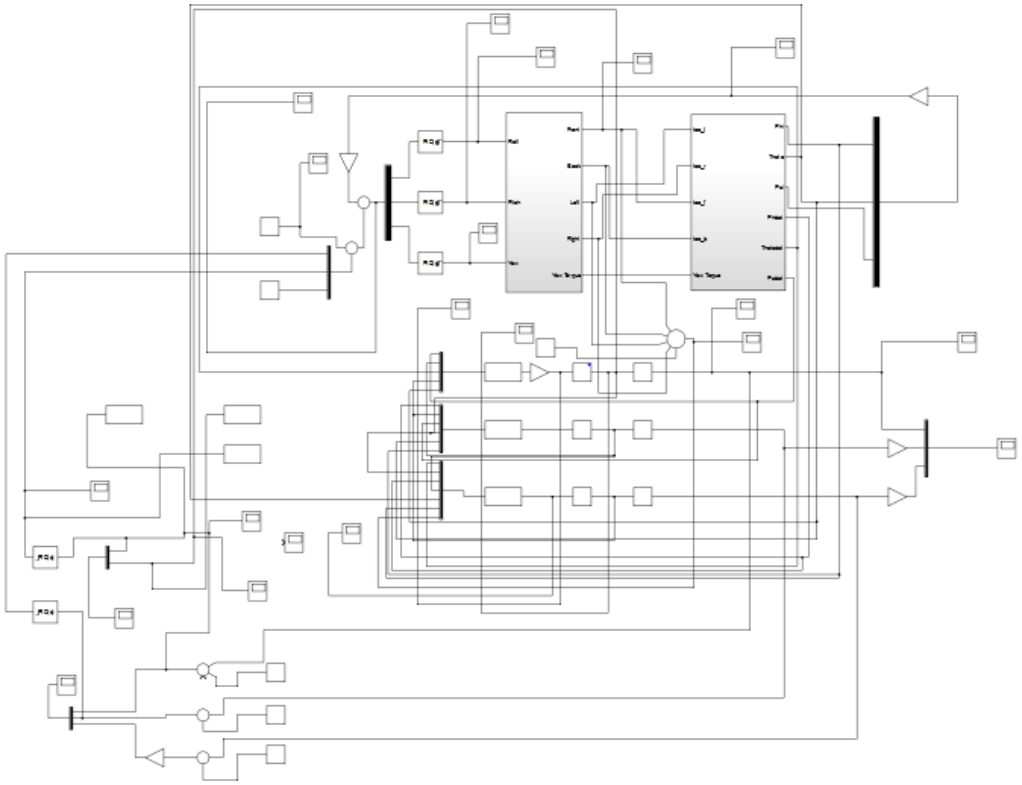
* Benchmarked @ Windows 7, AMD Phenom ii x6 3.4 GHz, 8 GB DDR3 RAM, 128 GB SSD
** Not Included Benchmarking Tests

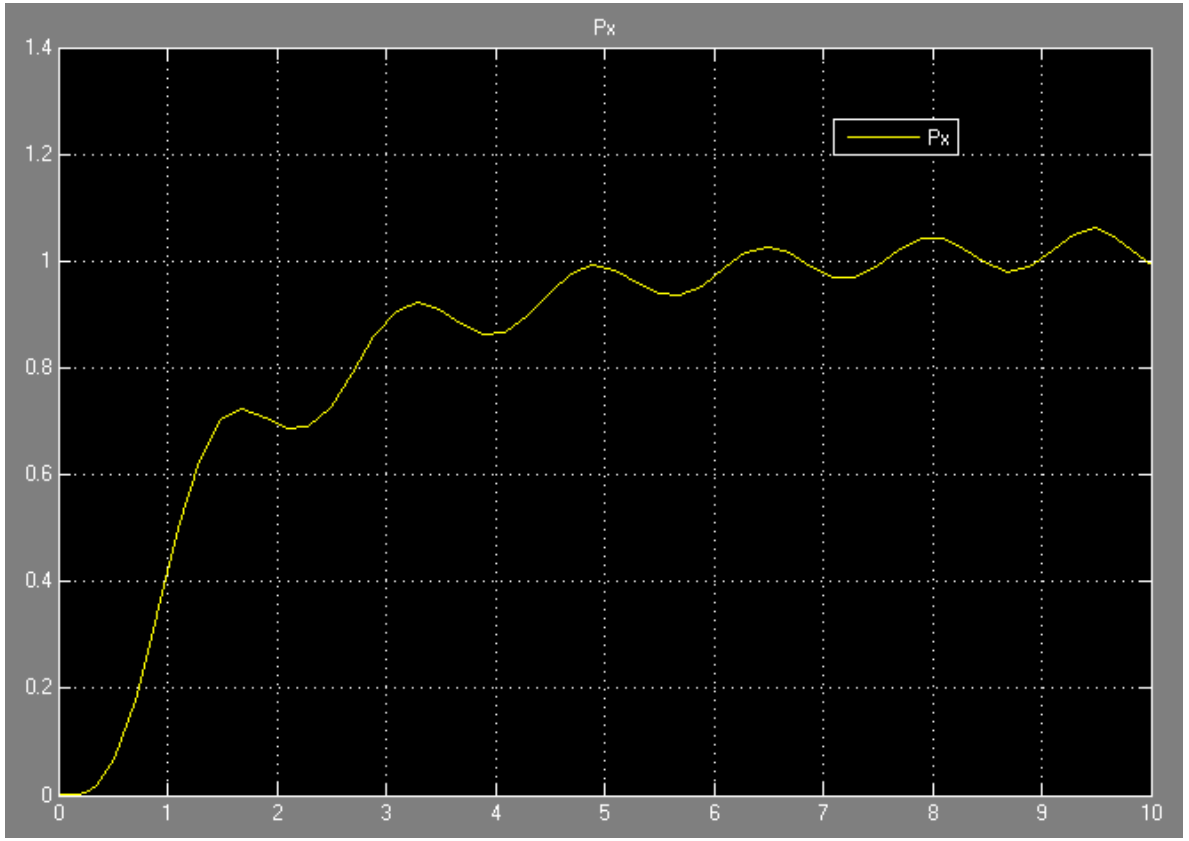
Kalman Filter – Position Tracking

- GPS Accuracy
  - Position : < 3 meters
  - Velocity: 0.1 m/s
- Might Cause Problem in Guidance and Controls
- Kalman Filter Sensor Fusion



Position Control: Cascade PID





Target Location → PID Controller (Position) → PID Controller (Angle) → Flight Mixer & Motors

Position Control: Adaptive Fuzzy Logic

Research Interest - Fuzzy Model Reference Learning Control

A learning controller that attempts to force a real System to behave like a reference model.

FMRLC systems have been proven useful for nonlinear time varying systems.

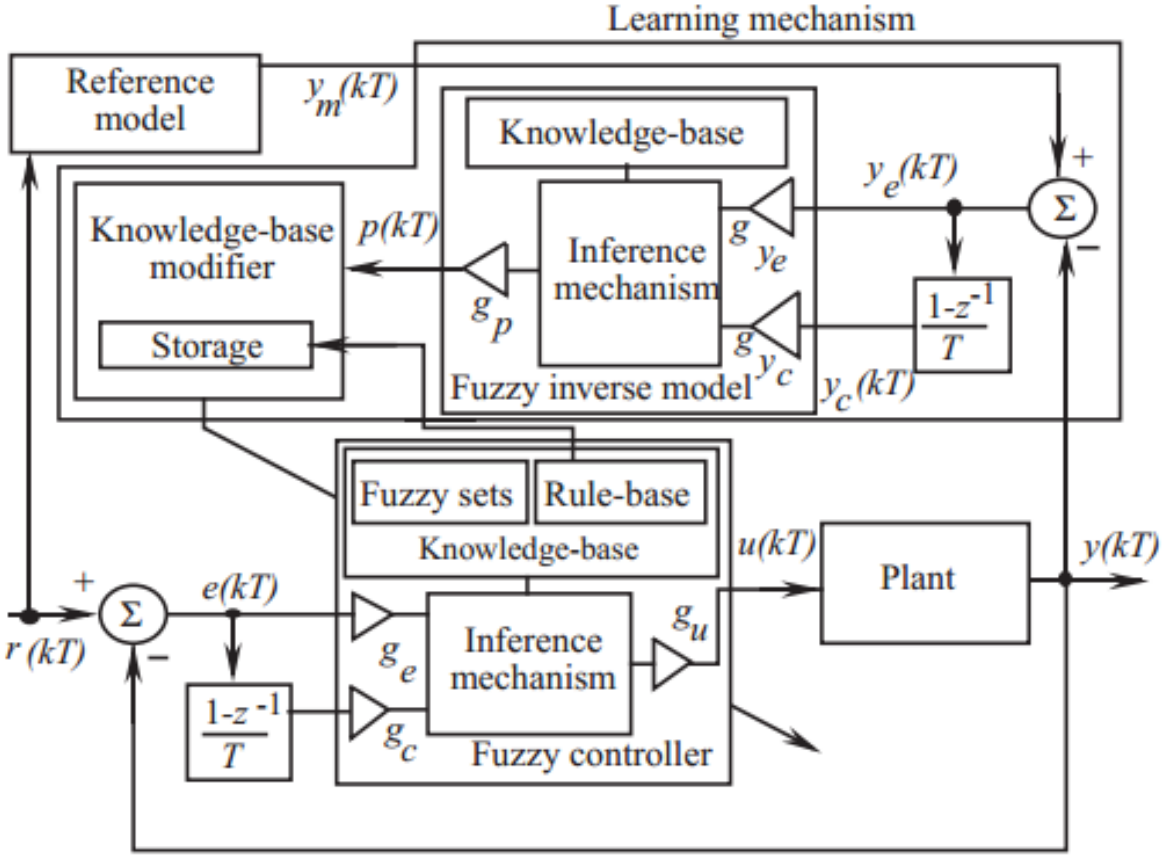


Figure 1: Functional Architecture for the FMRLC.

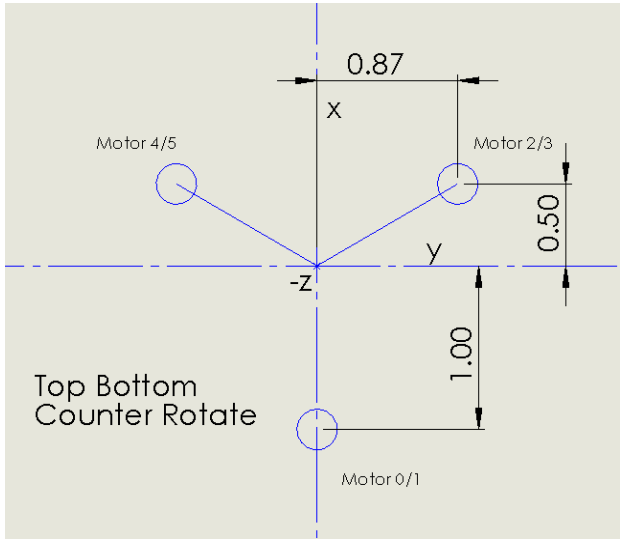
Questions?

Backup Slides

Intel D33217 GKE “Golden Lake”

PID Flight Test

Flight Mixer



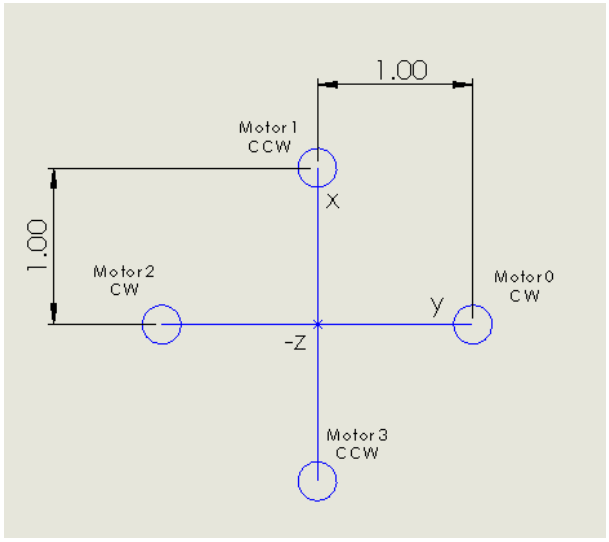


Image Credits

- Wikipedia
- Google
- Penn State Lunar Lion
- NASA
- http://dluong1.bol.ucla.edu/swarthmore/e11/lab4.htm