Purdue University | School of Aeronautics and Astronautics
COLLEGE OF ENGINEERING

UAS Autopilot

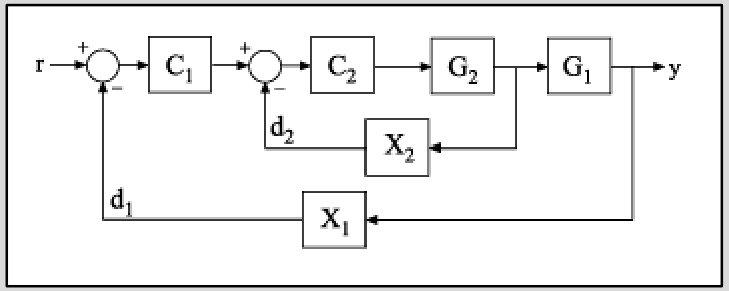
Phil Baldwin

Purdue University – AAE

1/17/2020

Outline

- Theory
  - Guidance
  - Navigation
  - Control
- Hardware
  - Setup
  - Testing
- Software
  - Choices
  - Use
  - Testing
- Integration



Northrop Grumman X-47B: Autonomous aerial refueling

Theory

- Goal: Control full non-linear EOMs of your 6 DoF system (aircraft)
- How :
  - Identify behavior of your aircraft: Parameter Identification
  - Provide controller for each dynamic mode that doesn’t behave how you want it to (Dutch roll, phugoid, etc.)
  - Provide controller for guidance/navigation
    - Navigation: Where you are
    - Guidance: Where you’re going
  - Ground station software rolls all this into one package, with most of the work already done for you, though it is not necessarily all-encompassing

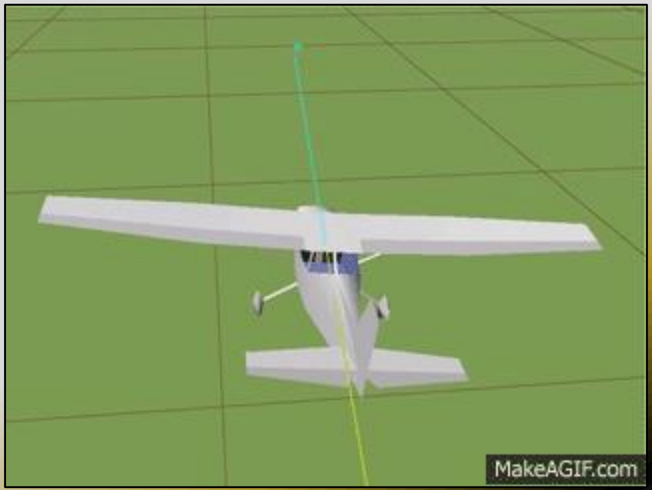
Theory – Parameter Identification

- Goal: Find stability derivatives
  - Relates the movement of a control surface, CG position shift, velocity, etc. to the reaction that the aircraft will have
  - Difficult to calculate by hand
- Predicted values – will need to coordinate values from several different sources
  - CFD
  - CAD
  - AVL
- Measured values
  - Flight tests: Accelerometers and airspeed are most common data
  - Wind tunnel: May be able to provide some data

PURDUE UNIVERSITY | School of Aeronautics and Astronautics COLLEGE OF ENGINEERING

Theory – Controllers

- Goal: Provide controller for each dynamic mode that doesn't behave how you want it to (Dutch roll, phugoid, etc.)
- Generally use linearized EOMs and successive loop closure to create controllers for each system
  - Define system inputs and outputs (e.g., elevator signal input, AoA output)
  - PID Control in Ground Station software
- Can implement additional filters



Theory – Navigation

- Goal: Know where you are
- Sources
  - GNSS (GPS, GLONASS, Galileo, BeiDou)
    - RTK for more precise positioning – requires extra hardware
  - Inertial (Based on accelerometers and/or gyros)
  - Magnetic Compass (Magnetometer)
  - Barometer
  - Sensor fusion – Uses all of the above, with Kalman filtering, to provide a better solution to the problem
    - Not trivial
    - Built in to software/firmware

Theory – Guidance

- Goal: Get where you want to go
- Driven by cross-track error: How far to the left or right (or below/above) are you of your desired track?
  - Generally a controller gain set to define how “aggressively” you want to get to your desired track
  - May be limited by bank angle limit, airspeed, or other factors
  - How close is close enough?
    - Is it the same in all phases of your flight plan?

Hardware

WARNING

None of us are equipped to deal with a severed finger (duct tape can only go so far). Please take steps to ensure that you stay safe when conducting bench, testbed, or production aircraft tests. Ideally, perform all tests *without* a propeller attached. Only attach a propeller once you are read to go fly.

Hardware

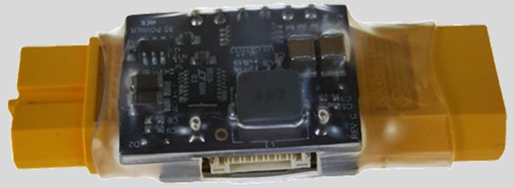
- A hardware solution that will allow for easy integration and that will allow you to connect servos, ESCs, radio receiver, etc., is desirable
- Pixhawk is the cheapest
- Pixhawk 2 is newer, with better connectors
- Pixhawk 4 the newest, with same connector types (JST-GH) as Pixhawk 2
- Some projects based on Raspberry Pi
- Additional "nice to have" (might be required, depending on mission design) components
  - External GPS
  - Pitot-static system

Pixhawk/PX4/Ardupilot

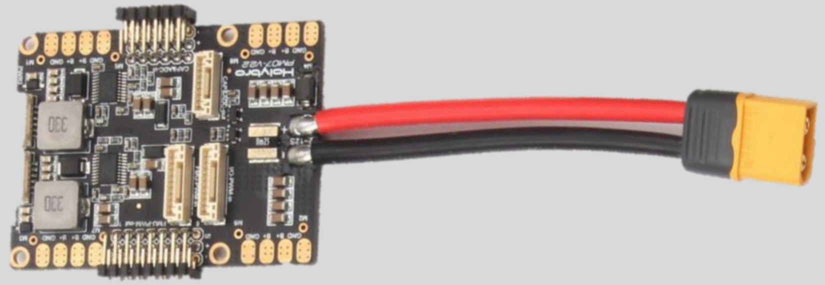
- "Pixhawk" refers to the physical hardware
  - Current models
    - Pixhawk 2.1 (*The Cube*)
    - Pixhawk 4
    - Pixhawk 4 mini
- "PX4" and "Ardupilot/Arduplane" refers to the firmware running onboard
  - QGroundControl GCS software can install PX4 or Ardupilot
  - Mission Planner GCS software can only install Ardupilot natively, but there is a hack for PX4
  - You can interact with either firmware with either GCS
- ***PX4 ≠ Pixhawk 4***

Powering your Pixhawk

- Different models have slightly different power connectors, but should come with the correct items to connect to a LiPo battery
  - Pixhawk 2 has a "power module" that sits between your LiPo and your ESC
    - Comes pre-installed with XT60s
    - Also measures battery voltage and current draw
      - Nice to monitor for mission duration and battery health



  - Pixhawk 4 has options
    1. A huge, annoying power distribution/IO board that is great for quad- and octo-copters
    2. In-line power module and compact IO breakout (please buy this)



Hardware

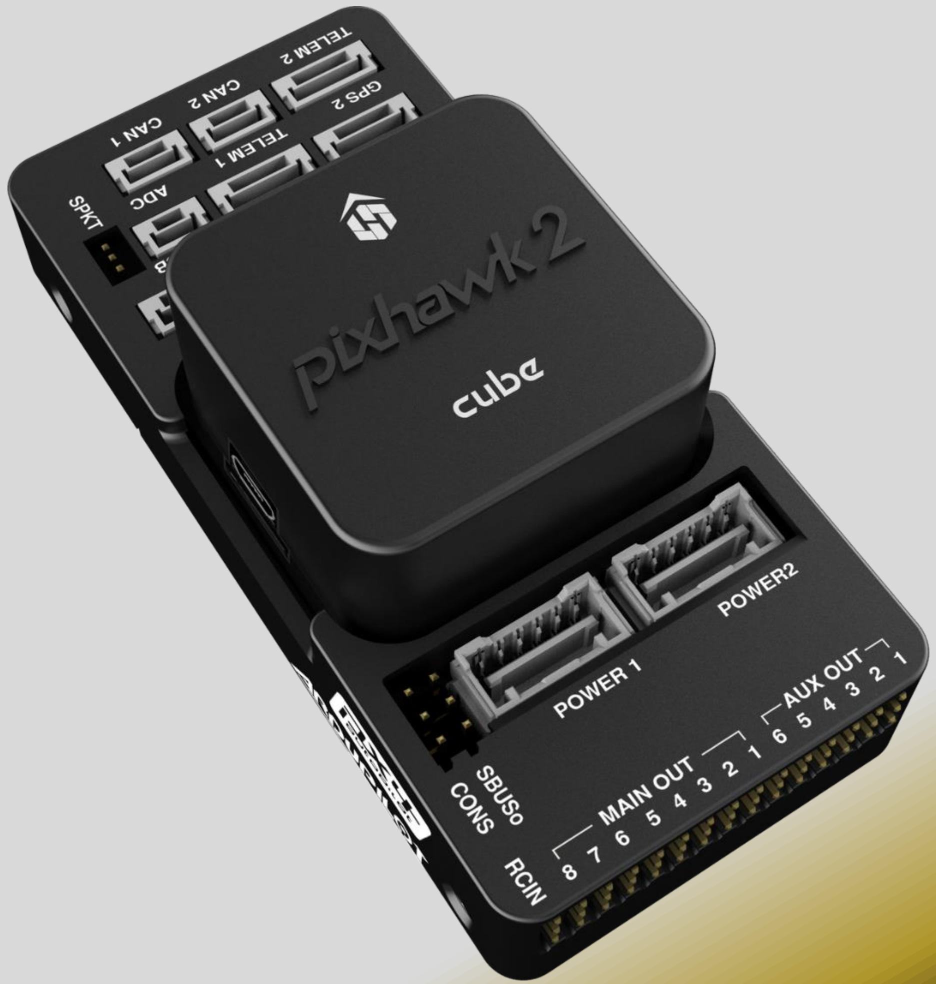
- Pixhawk
  - Onboard IMU (accelerometers, gyros, barometer, magnetometer)
  - Lots of extra connectivity options
    - I2C
    - UART
    - CAN
    - ADC
    - …
- Telemetry/control radio is nice to have
- Each team already working?

Hardware – Setup

- Not going to describe each individual step here
  - Lots of tutorials available online
  - Not enough time
- Install Pixhawk unit as near to the CG as possible (arrow facing forward for all software defaults)
  - Forward-aft adjustment easiest
  - Vertical also important, but less variation
  - Foam or rubber to dampen vibrations
- Attach all electrical connections
  - Verify servo connections not reversed anywhere (very common)
  - Don't stretch wires taut
  - Try to avoid a rat's nest

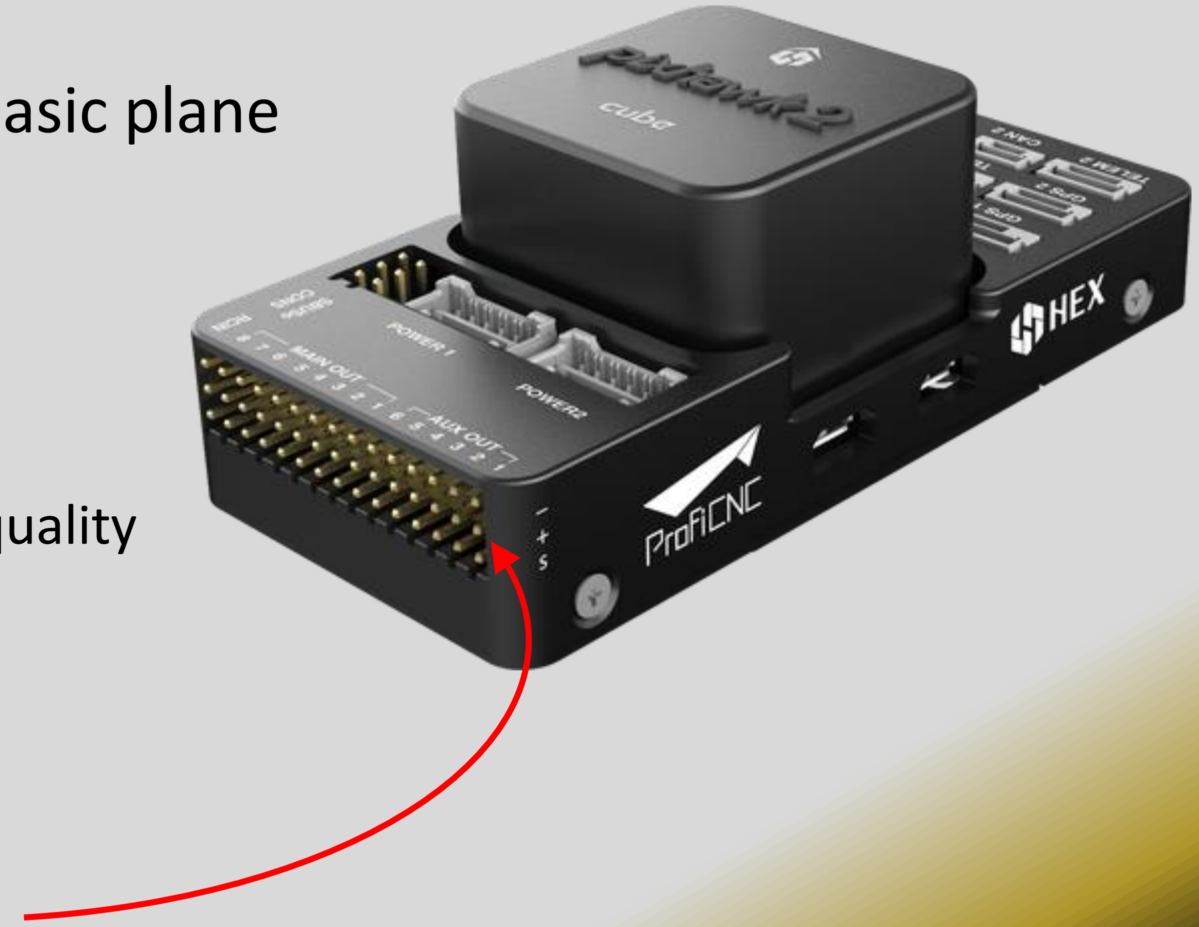
Hardware – Setup

- Use MAIN OUT for all connections if possible (this provides hardware failsafes)
- Mapping between servo channels and Mission Planner
  - Main Out 1 = Servo1
  - Main Out 2 = Servo2
  - …
  - Aux Out 1 = Servo9
  - Aux Out 2 = Servo10
  - …
- Do not use RCIN for anything other than PPM receiver connection



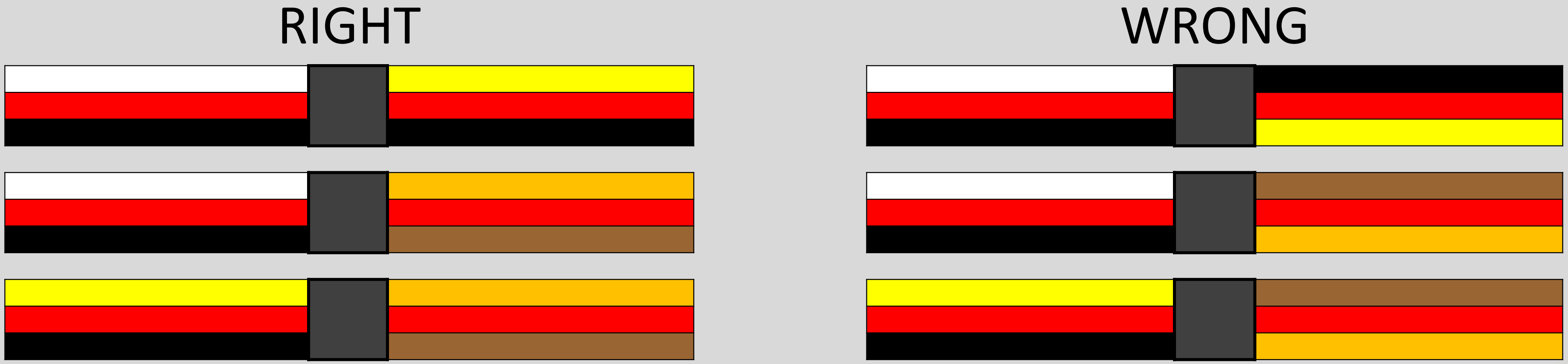
Hardware – Power

- Three main power consumers on a basic plane
  - Motor (a LOT of power)
  - Servos (some power)
  - Pixhawk (really not very much power)
- Other power consumers
  - Other on-board processors (GPUs, air quality sensor pack?)
  - Transmitters (video and telemetry)
- Power sources
  - Pixhawk power brick
    - Only powers "brains", NOT the servo rail
  - ESC's BEC, standalone BEC, or secondary battery
    - Needed to power servo rail
    - Operating voltage of BEC/battery must match servos



Hardware – Servo Wires

White/Yellow/Orange: PWM Signal Red: +VDC Black/Brown: -VDC



- Easiest way to avoid issues is to standardize all servos and wires, but that takes some additional planning
- Custom-length servo wires take longer to integrate but probably will save some weight and space

Hardware – Common issues/FAQs

- The connectors are delicate – be nice
- Be mindful of board orientation and mounting
- Remember that wires take up space
  - And have non-zero weight
- Purchase 915 MHz radios, not 433 MHz
- "The GCS says the battery is too low."
  - Make sure the power module is properly set up in your GCS
- "There is no signal to the servos."
  - Make sure the safety switch has been pressed (aka vehicle is "unsafe")
  - Make sure the servos are plugged in to the right location
  - Make sure the flight mode is set to your desired mode

Hardware – Testing

- Bench testing highly recommended before installation in airframe
  - Chances are your final aircraft will not be ready for integration before you get all of your autopilot components together
  - Pre-assemble/test as much of the system as possible
    - Include as much of the actual system you will use
      - Servo extensions
      - Cameras
      - Transmitters
    - A couple of pieces of wood or foam will work – it need not be complex
  - Allow access to Pixhawk and other components to check connections or current conditions
    - Much harder to dig around inside of a fuselage and trace wires

Purdue University | School of Aeronautics and Astronautics, College of Engineering

Hardware – Testing

- Testbed aircraft
  - Use to test/practice…
    - Assembly/start-up steps
    - **Ground station practice**
    - **Flight modes**
    - Sensors (airspeed, cameras)
  - Can use to create or refine…
    - Stability models and tuning
    - Methods for data collection
  - Use a known stable design
    - Mitigate risk of using the autopilot the first time in your only aircraft

Software – Choices

- Lots of options, especially depending on platform; can use a tablet or desktop (laptop) based system
  - Mission Planner*
  - QGroundControl
  - UgCS
- No "right" answer; download one or all of them, they are all free
- Spend some time using each one
  - Steep learning curve for new users
  - Best way to learn it is to use it

*I'll be using links to this software, mostly. QGC has analogs to most of the links I provide.

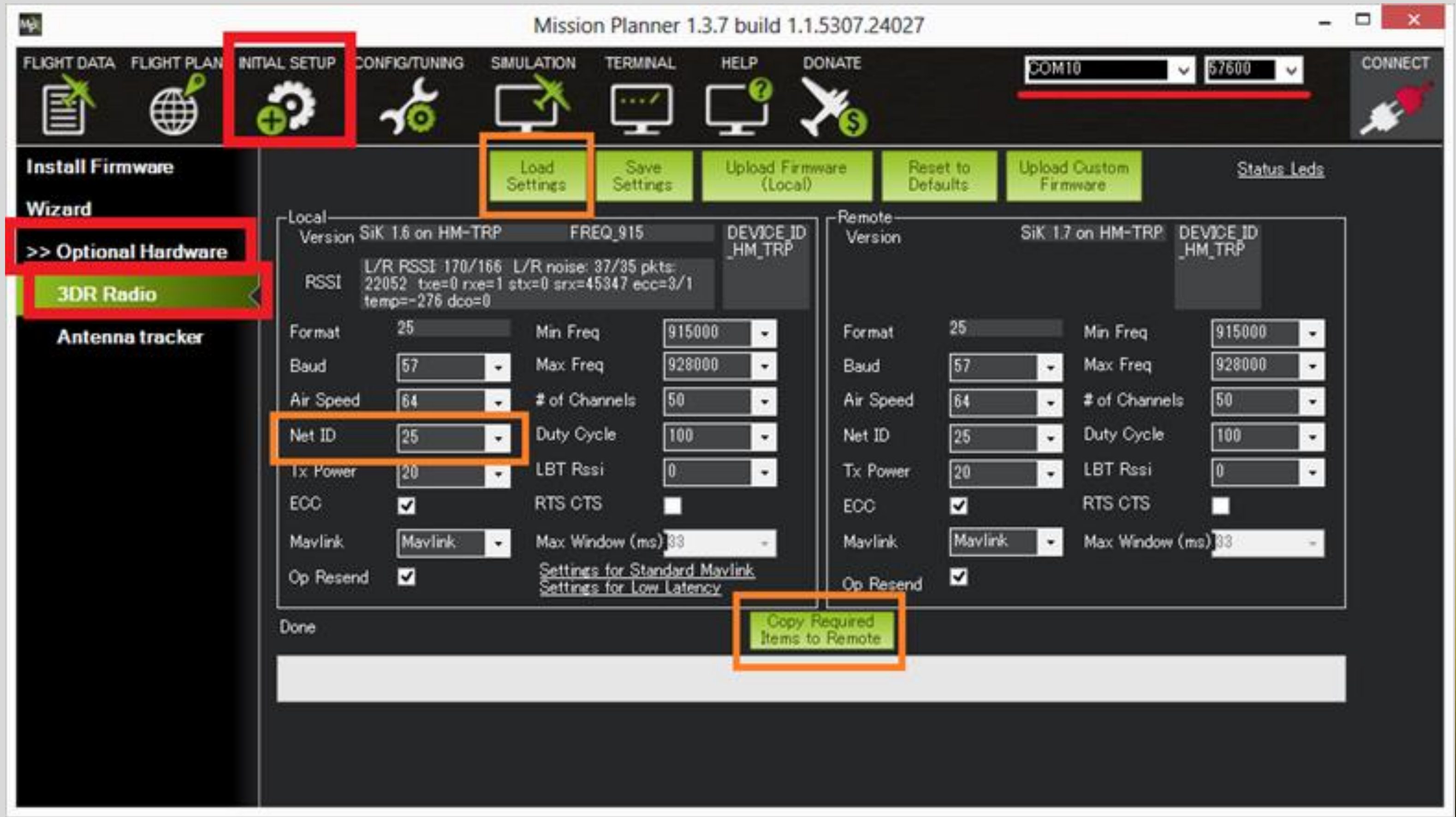
Software – Uses

- Control
  - Mission Planning [QGC]
    - Waypoints
    - Vertical (profile) planning
    - Point of interest (loiter, circle)
  - Component integration
    - Cameras
    - Sensors
  - Joystick (if you want)
- Telemetry
  - Real-time info on location, speed, ETA, etc.
  - Diagnostic/status info

Software – Uses

- Telemetry channels
  - To prevent interference with other teams
- Steps
  1. Plug both radios in (one to GCS via USB, other to Pixhawk telem port)*
  2. Initial Setup → Optional Hardware → SiK Radio
  3. “Load Settings”
  4. Change Net ID
  5. Copy Req’d to Remote

*Alternate: Connect each to GCS via USB independently, then change Net ID



Local

Version: SiK 1.6 on HM-TRP FREQ_915 | DEVICE_ID _HM_TRP

RSSI: L/R RSSI: 170/166 L/R noise: 37/35 pkts: 22052 txe=0 rxe=1 stx=0 srx=45347 ecc=3/1 temp=-276 dco=0

Format	25	Min Freq	915000
Baud	57	Max Freq	928000
Air Speed	64	# of Channels	50
Net ID	25	Duty Cycle	100
Tx Power	20	LBT Rssi	0
ECC	✓	RTS CTS	☐
Mavlink	Mavlink	Max Window (ms)	33
Op Resend	✓		

Settings for Standard Mavlink
Settings for Low Latency

Remote

Version: SiK 1.7 on HM-TRP | DEVICE_ID _HM_TRP

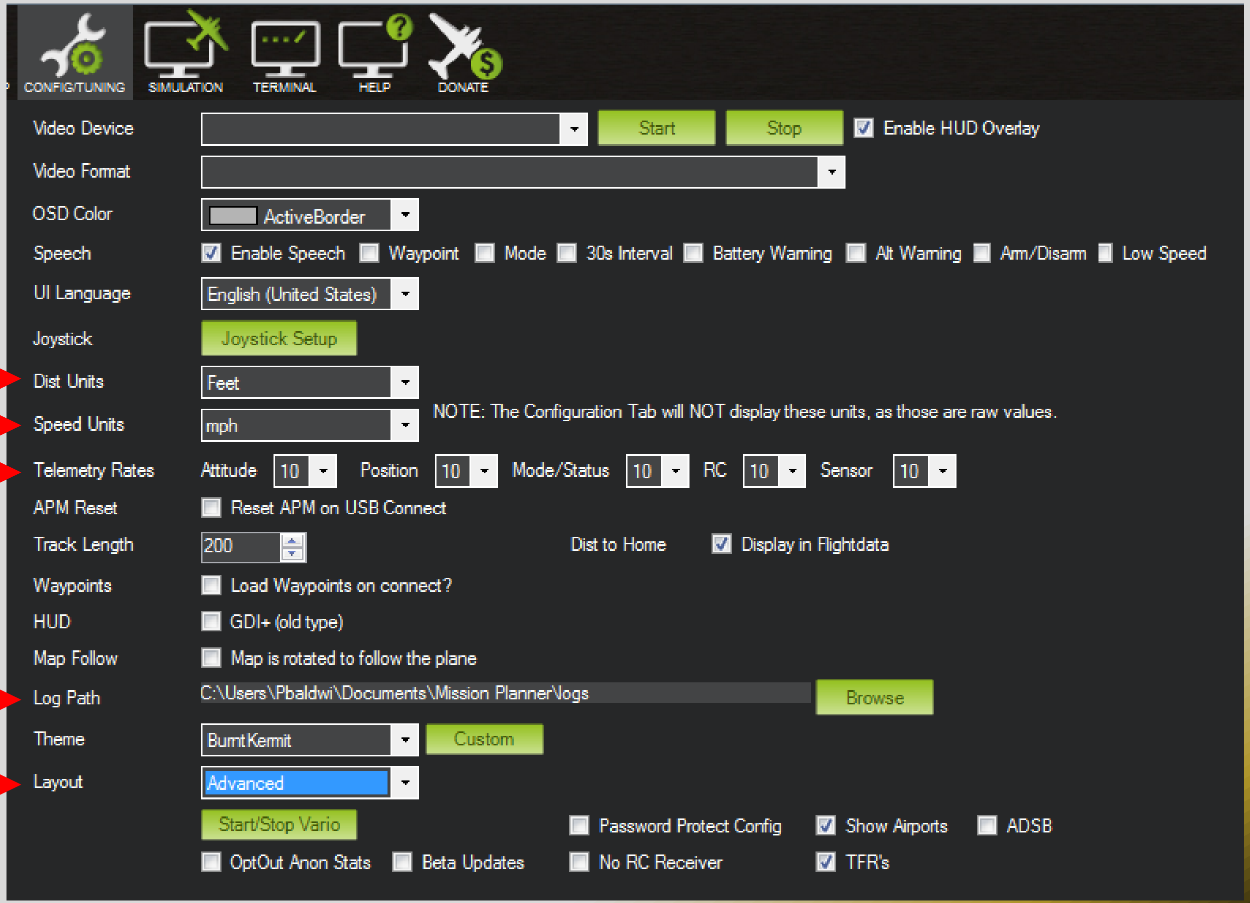
Format	25	Min Freq	915000
Baud	57	Max Freq	928000
Air Speed	64	# of Channels	50
Net ID	25	Duty Cycle	100
Tx Power	20	LBT Rssi	0
ECC	✓	RTS CTS	☐
Mavlink	Mavlink	Max Window (ms)	33
Op Resend	✓		

Done

*Older version of Mission Planner shown

Software – Uses

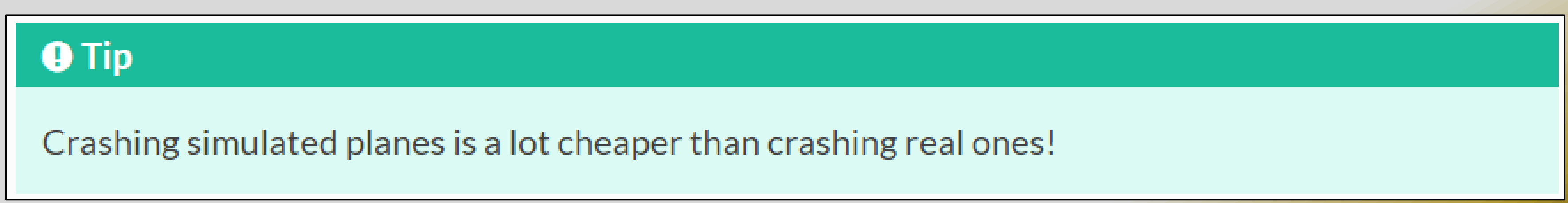
- Mission Planner Setup
  - Units (make sure they match what you are designing in)
  - Telemetry rates
    - Higher rate = finer data, may lose some data over telemetry (depends on aircraft configuration, geography, etc.)
  - Log Path
    - To find telemetry logs after flight
  - Advanced mode
    - Enables full parameter list and tree



Video Device | Start | Stop | ☑ Enable HUD Overlay
Video Format
OSD Color: ActiveBorder
Speech: ☑ Enable Speech ☐ Waypoint ☐ Mode ☐ 30s Interval ☐ Battery Warning ☐ Alt Warning ☐ Arm/Disarm ☐ Low Speed
UI Language: English (United States)
Joystick: Joystick Setup
Dist Units: Feet
Speed Units: mph
NOTE: The Configuration Tab will NOT display these units, as those are raw values.
Telemetry Rates: Attitude 10 Position 10 Mode/Status 10 RC 10 Sensor 10
APM Reset: ☐ Reset APM on USB Connect
Track Length: 200 Dist to Home ☑ Display in Flightdata
Waypoints: ☐ Load Waypoints on connect?
HUD: ☐ GDI+ (old type)
Map Follow: ☐ Map is rotated to follow the plane
Log Path: C:\Users\Pbaldwi\Documents\Mission Planner\logs Browse
Theme: BurntKermit Custom
Layout: Advanced
Start/Stop Vario ☐ Password Protect Config ☑ Show Airports ☐ ADSB
☐ OptOut Anon Stats ☐ Beta Updates ☐ No RC Receiver ☑ TFR's

Software – Uses

- Simulation [QGC]
  - Standard “plane” uses default gain values to guess how your airplane will fly – you need to update the defaults to be able to simulate your airplane without it connected
  - With your real plane connected (via USB or telemetry), see how servos react, cameras point
  - Will not be perfect; fidelity is fairly low
  - Use to test out new mission profiles
  - Use to teach additional team members to operate your plane

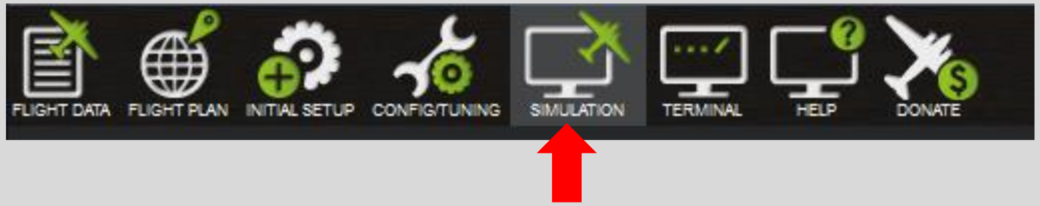


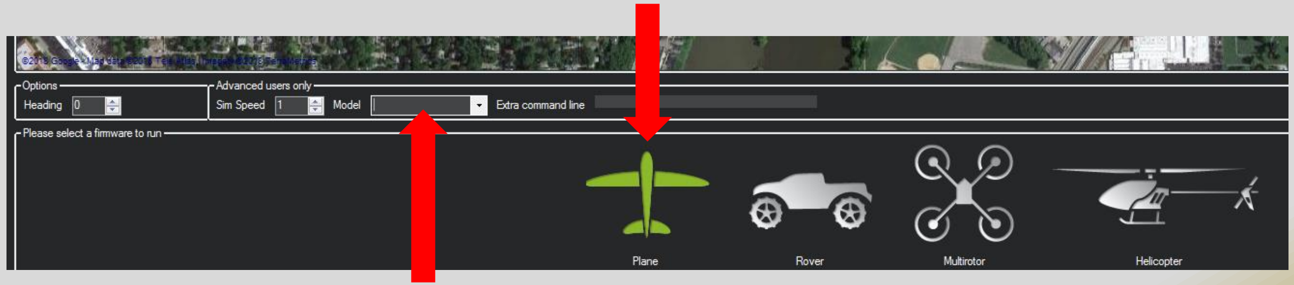
Tip

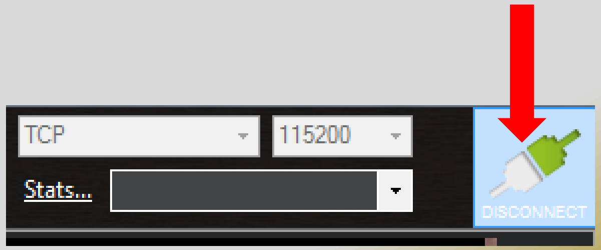
Crashing simulated planes is a lot cheaper than crashing real ones!

Software – Uses

- Simulation
  - Click “Simulation”
  - Select Model
  - Click “Plane”
  - Will load SITL items
  - Initializes a simulated plane
  - Can interact just as if it was a real plane
    - Load waypoints
    - In-situ mission (takeoff, loiter, point camera, etc.)
    - Change parameters on-the-fly
  - Disconnect “TCP” to stop

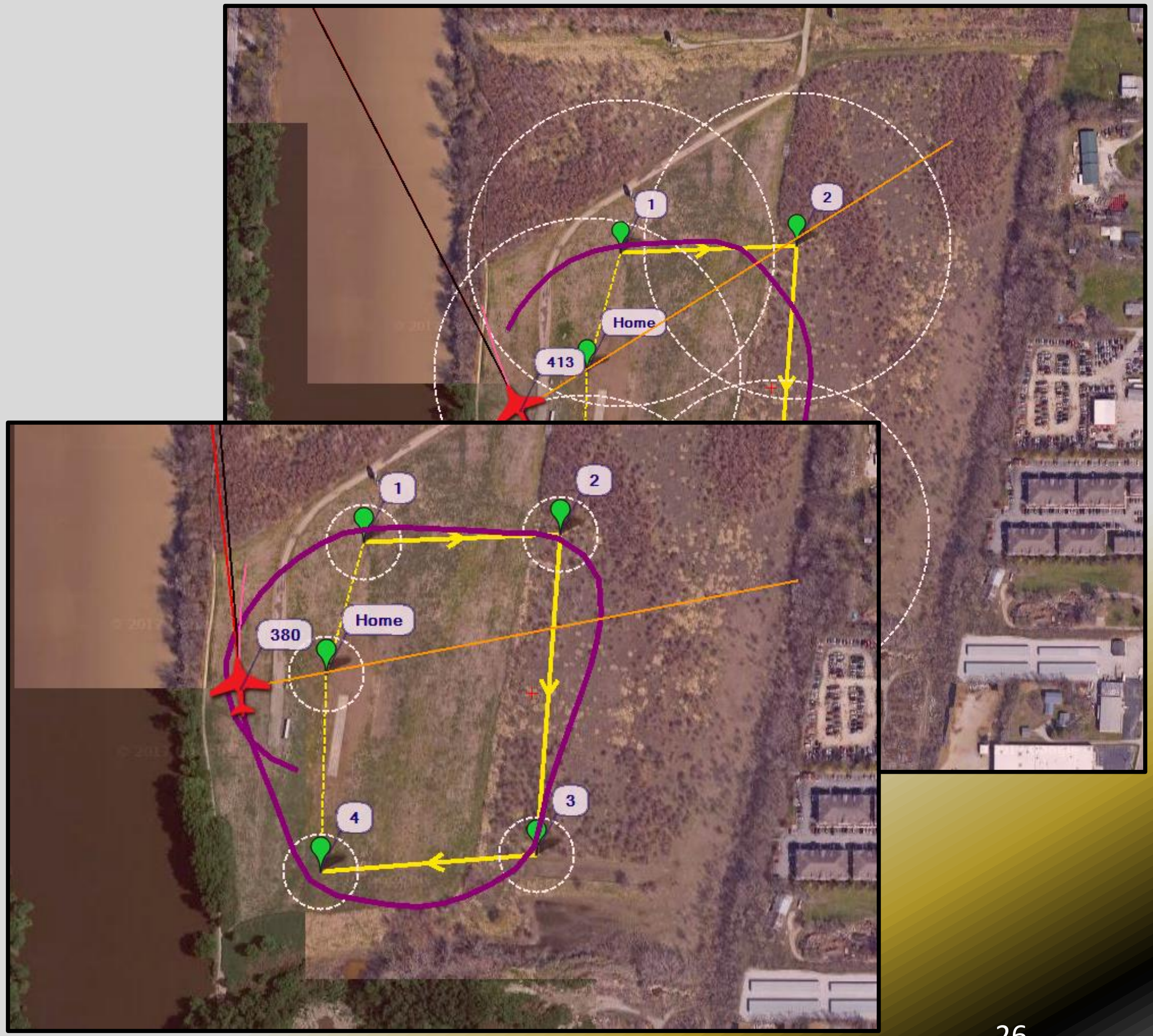



Options
Heading 0
Advanced users only
Sim Speed 1
Model
Extra command line
Please select a firmware to run


TCP
115200

Software – Uses

- A word on waypoint definition
  - Defined as a 3-Dimensional point
    - Latitude, Longitude, Altitude
  - Built-in Lat-Long "tolerance"
    - Radius (NOT diameter)
    - As soon as the aircraft is within radius, the next waypoint is sequenced
  - Be careful with altitude definition
    - Absolute = MSL
    - Terrain = AGL
    - Relative = Compared to Home point



Software – Testing

- Same pretense as hardware testing: Know what you're getting into and practice it
- Set up Pixhawk to be able to talk to your ground station
- Test out different modes in testbed
  - Use ground station and RC transmitter to change modes
- Understand what the software is doing; don't just assume it will do what you want it to
- Know how to set the failsafe and what to expect if it takes effect
- Always be ready to take control back from the autopilot
  - Know where the "Manual" or "Override" switch is and how to use it

Integration

- Probably the hardest part of all of this (in the build process), but the least material to talk about
  - All of the airplanes are different
  - All of the missions are different
  - Must relate specific performance of each given system to your KPIs and decide what needs done first
- This will take you longer than you think, unless you have specifically designed how to integrate each individual component (down to servo wires) into your airframe
  - Generally there is ample tape used in this process ☹

Integration

- Make sure you can access Pixhawk to plug components in
- Install components individually
  - Servos
  - ESC
  - Camera(s)
  - Pitot-static tube
  - Lidar/other rangefinder
  - GPS Antenna
  - Pixhawk unit (and associated accessories)
  - Radio transceivers

Integration

- Make sure everything works individually before plugging in to Pixhawk
  - Common to think the Pixhawk is malfunctioning, but there is actually a servo extension installed upside down in line
  - Use a servo tester
  - Ensure prop rotation direction is correct (also a common “Oops” moment, but sometimes fixable with ESC programming)
- Plug in all components to Pixhawk
  - Should already have wireless ground control connection made, right?
  - Take care with the tiny connectors (pins are easy to bend, plastic is brittle)
- Go through start-up process [QGC]

Integration

- Go fly!

  NOTE: It is highly recommended to fly by hand initially, then switch modes to "Stabilize" to check that the system works

  - Arm the system
  - Launch and climb to a safe altitude
  - Engage "Stabilize" or "FBW" mode
  - Always be ready to remove control from the autopilot
  - Ardupilot includes an "Autotune" mode that is supposed to set some of the PID values automatically; trust but verify is a good principle here.
  - For the rest of the controller values, set them individually [QGC]
    - May be able to get a good first guess from community-submitted values
    - Only use a similarly sized aircraft if its weight and configuration is also similar

Integration

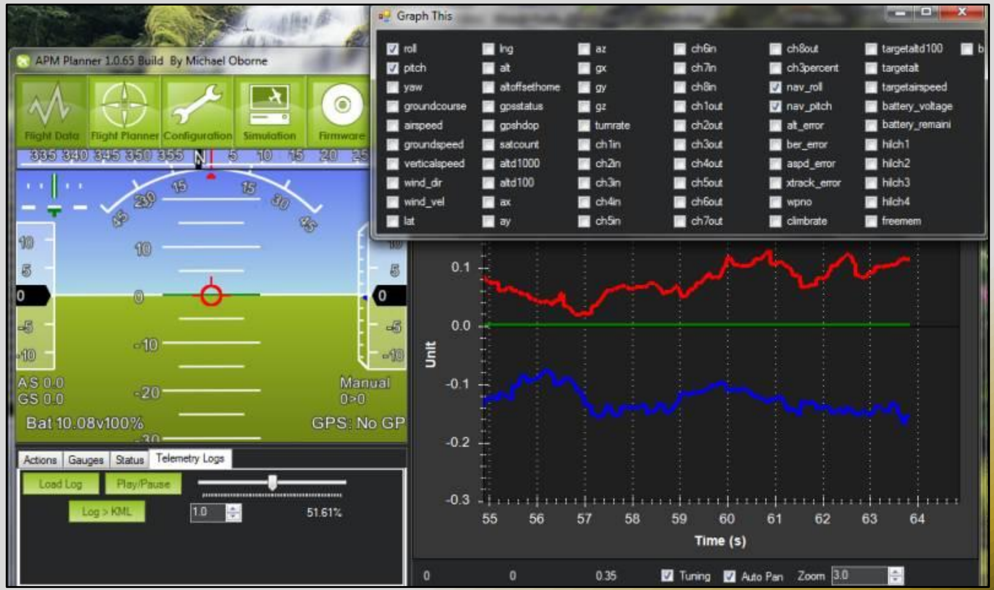
- Flight Modes [QGC]
  - Control specifics of what your aircraft is doing
    - Can pre-plan with "Auto" (MP) or "Mission" (QGC)
      - Set up automated takeoff (even hand launching)
      - Follow waypoints
      - Loiter over a specific point for a specified amount of time
      - Drop a payload?
      - Return to launch
      - Automated landing
    - Can hand-fly with "Stabilize", "FBW", or "Cruise" (or "Manual", obviously)
    - Ad-hoc automated modes, "Loiter" and "Circle"
  - Lots of options available, and lots more you can add and customize

Integration

- Built-in safety
  - Envelope protection
    - Prevents stall, overspeed, or other undesirable conditions
    - “Free insurance”
    - Once again, trust but verify its operation
  - Geo-fencing
    - Keep your airplane IN a specific area (Flight box)
    - Keep your airplane OUT of a specific area (Airports, sporting events, etc.)
  - Terrain following
    - Might be useful in mountains or other adverse terrain
    - Might be less useful on the ocean

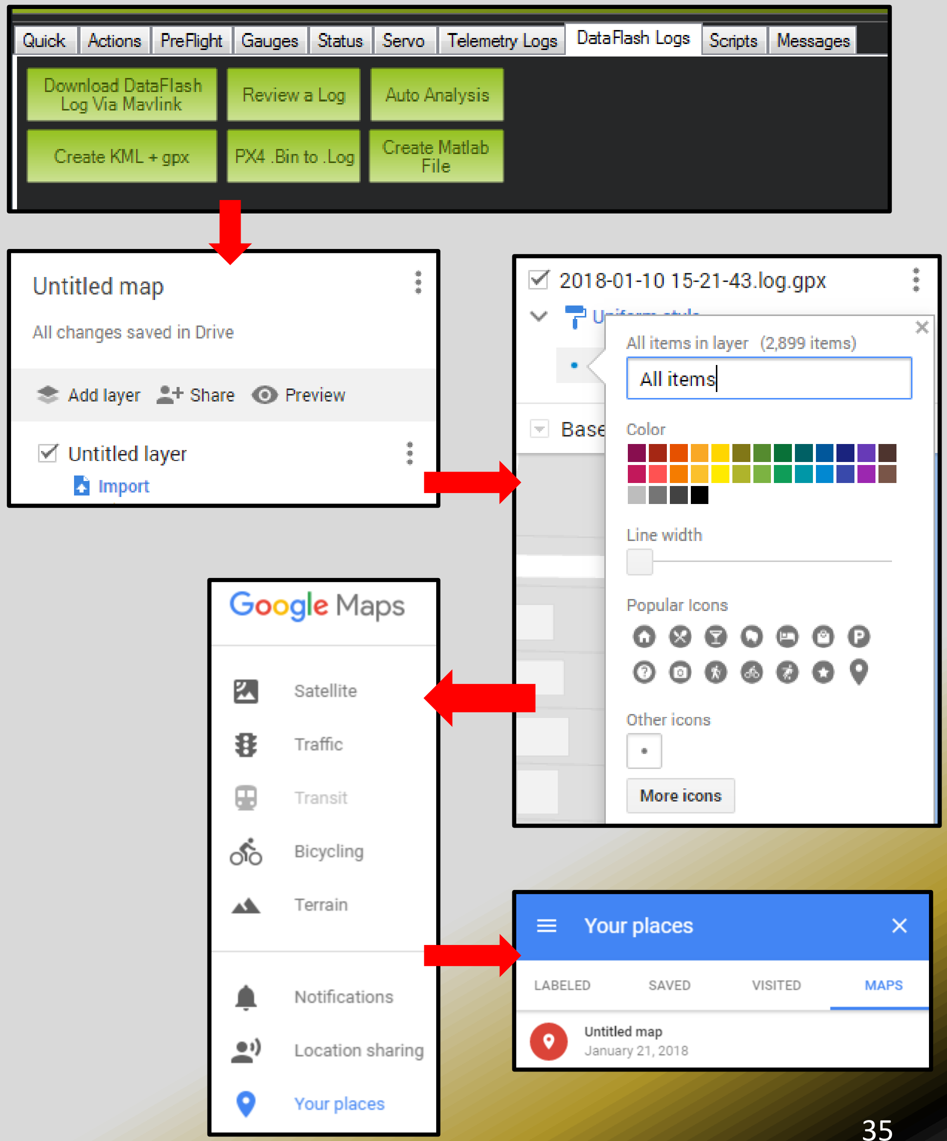
Integration

- Debrief
  - Use data collected during flight [QGC]
    - Prove a parameter is met
      - Altitude during cruise
      - GPS location of payload drop
      - Flight time
      - Battery capacity (mAh) consumed
    - See if there are any improvements to be made
      - Change controller gains
      - Change climb-out angle
      - Change landing approach speed
  - Discuss what went right
  - Discuss anything that went poorly
    - Use data to analyze and correct for next flight

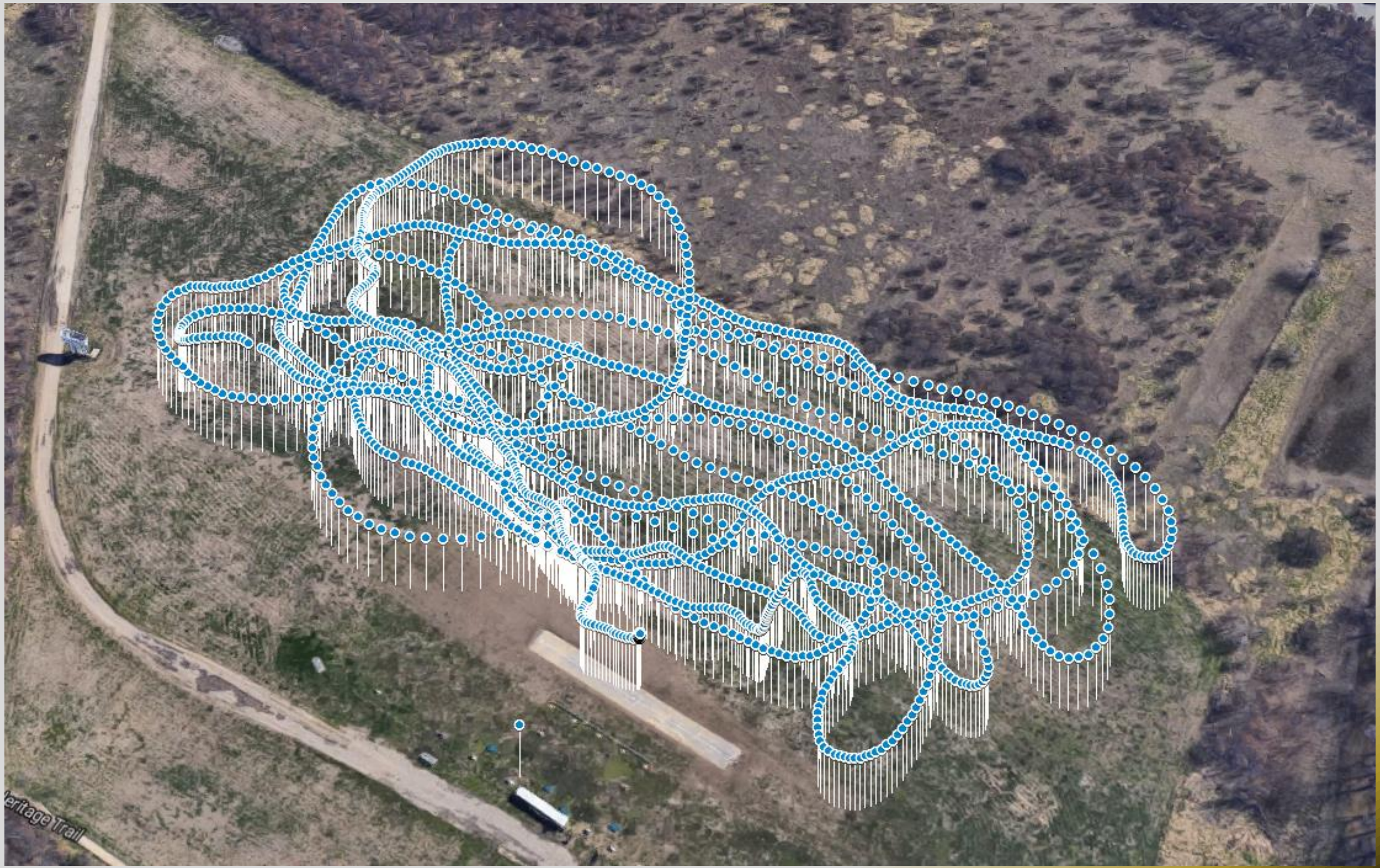


Integration

- Debrief
  - View 3D flightpath
    - Convert dataflash log to GPX
    - Upload GPX to google.com/mymaps
      - Under Untitled Layer → Import
    - Uniform Style → Whatever icon you want → Smallest Line Width
    - Save
    - Go to maps.google.com
    - Click on Your places → Maps
    - Select the map you generated
    - Click 3D View
  - This used to be a lot easier with Google Earth…



Integration

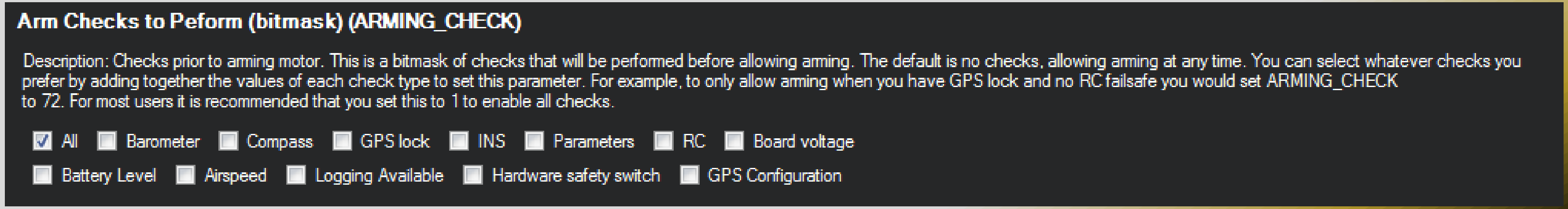


Integration

- Use your now fully-autopilot-enabled aircraft to carry out an extensive flight test program
- Then save sea turtles, , or look for sad Clemson tailgaters

Integration – Common Issues

- Can't connect Mission Planner to Pixhawk over USB
  - Check COM port assignment in Windows Device Manager
- Can't connect Mission Planner to Pixhawk over Radio
  - Check both radios have the same Net ID
- Servos won't work
  - Check servo wires, make sure they are receiving power (5-6 VDC)
  - Check orientation of wires at Pixhawk and any connections
  - Check if channels enabled when A/C Unsafe and/or Not Armed
- Aircraft won't arm
  - Investigate Arm Checks parameter – If you don't have an airspeed sensor, but require one to arm, you will have issues



Arm Checks to Peform (bitmask) (ARMING_CHECK)

Description: Checks prior to arming motor. This is a bitmask of checks that will be performed before allowing arming. The default is no checks, allowing arming at any time. You can select whatever checks you prefer by adding together the values of each check type to set this parameter. For example, to only allow arming when you have GPS lock and no RC failsafe you would set ARMING_CHECK to 72. For most users it is recommended that you set this to 1 to enable all checks.

☑ All ☐ Barometer ☐ Compass ☐ GPS lock ☐ INS ☐ Parameters ☐ RC ☐ Board voltage
☐ Battery Level ☐ Airspeed ☐ Logging Available ☐ Hardware safety switch ☐ GPS Configuration

Homework for teams

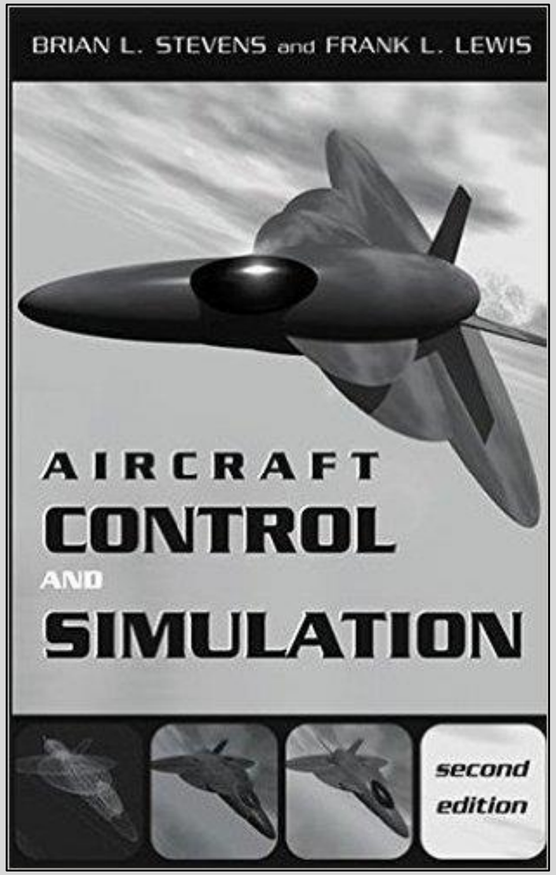
- Set your telemetry radio to the following channels

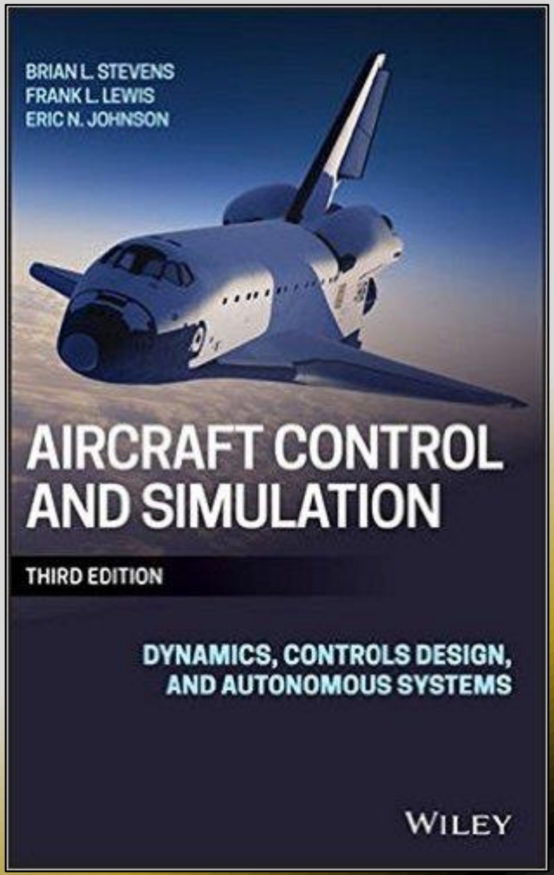
Team	Net ID
1 – BOOST	10
2	20
3 – Goldeneye	30
4 – SAILR	40
5 – Guardian Angel	50
6	60
7	70
8	80

- What video frequencies are teams using?
  - Try to avoid interference from identical/adjacent/harmonic frequencies

Related reading

- Aircraft Control and Simulation, Brian Stevens/Frank Lewis/Eric Johnson (3rd Ed.)
  - 2nd Edition*
  - 3rd Edition (has a chapter on sUAS)
- ArduPlane Docs
- QGroundControl Docs


BRIAN L. STEVENS and FRANK L. LEWIS
AIRCRAFT CONTROL AND SIMULATION
second edition


BRIAN L. STEVENS
FRANK L. LEWIS
ERIC N. JOHNSON
AIRCRAFT CONTROL AND SIMULATION
THIRD EDITION
DYNAMICS, CONTROLS DESIGN, AND AUTONOMOUS SYSTEMS
WILEY

*I have this if anybody wants to borrow it

Questions?