SOFTWARE TOOL INSTALLATION & RPILOT FLIGHT CONTROLLER TEST INSTRUCTIONS

Version -

2021 July

VERSION HISTORY

This is the initial release of the PythonPilot software development tool installation guide and RPiLot electronics assembly test procedure, Rev. -.

Version #	Authored By	Revision Date	Approved By	Approval Date	Reason
-	D. Haessig	44 July			Initial release

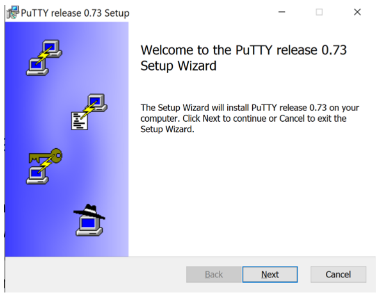
TABLE OF CONTENTS

1. SOFTWARE TOOL INSTALLATION

PUTTY

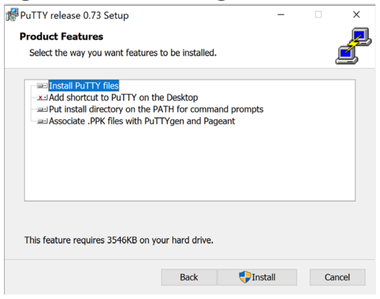
PuTTy is an open source terminal emulator, serial console, and file transfer application. We use it with the PiQuad to communicate with the Raspberry Pi (rPi) over WiFi. When powered up the PiQuad will boot and execute the PythonPilot^{TM} flight controller without PuTTy, starting in Idle Mode with props disabled. PuTTy can be started and used to examine the telemetry data streaming from the PiQuad, and to enter input commands to the Linux OS or to the PiQuad for operational control, and for altering telemetry data content for real-time visualization (topics to be covered later).

Go to the Download PuTTy site: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html and install using '`putty-0.73-installer.msi`', the 32 bit version. You will be asked 'What do you want to do with putty-0.73-installer.msi? Click 'Run', then click through the following install screens, including:



Welcome to the PuTTY release 0.73 Setup Wizard

The Setup Wizard will install PuTTY release 0.73 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.



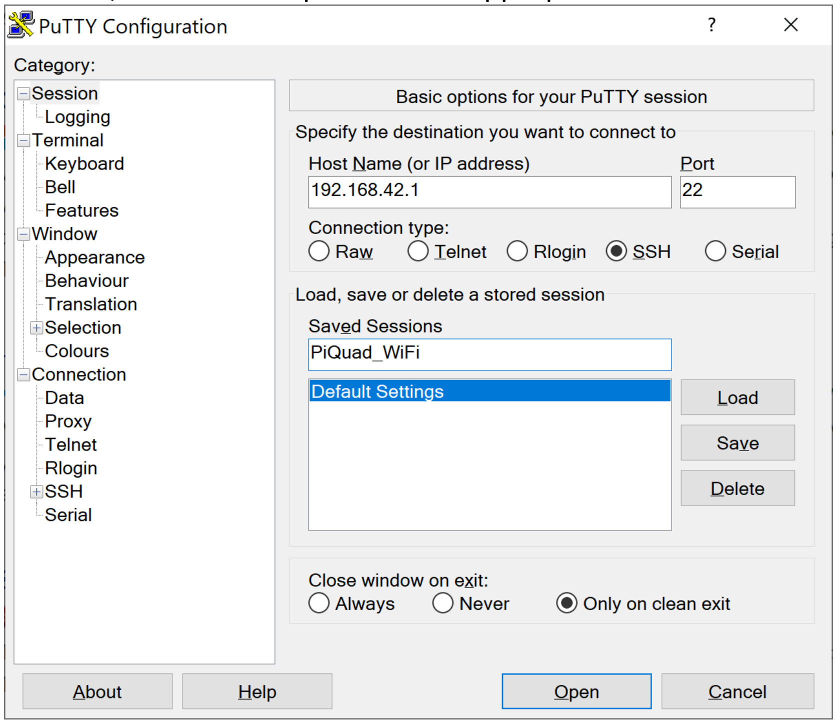
Product Features

Select the way you want features to be installed.

- Install PuTTY files
- Add shortcut to PuTTY on the Desktop
- Put install directory on the PATH for command prompts
- Associate .PPK files with PuTTYgen and Pageant

This feature requires 3546KB on your hard drive.

Start PuTTy from the Windows Start button. The following will open. Into this enter the IP address as shown below, and save this profile to an appropriate name like 'PiQuad_WiFi':



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 192.168.42.1

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions: PiQuad_WiFi

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

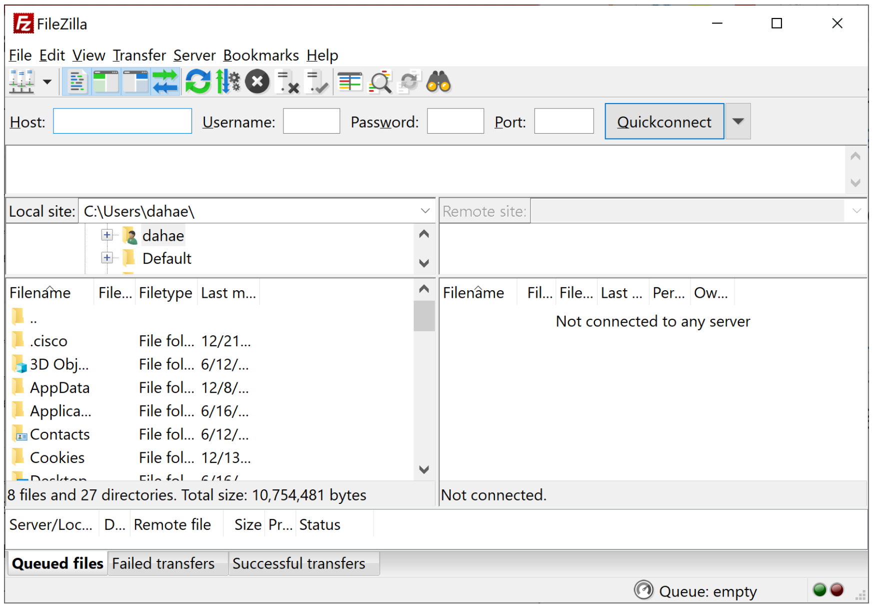
Click 'Save' and then 'Open'. A PuTTy terminal window will open looking for that IP address, which it will not find. We'll hook that up later.

NOTEPAD++

NotePad++ is a free source code editor that is great for editing Python. I'm running the 64-bit version 7.8.6. To download go to: https://notepad-plus-plus.org/downloads/v7.8.6/. It will ask 'What do you want to do with npp.7.8.6.installer.x86.exe'? Click 'Run' and click through the installation screens.

FILEZILLA

FileZilla is a convenient tool for managing files on the rPi running under Embedded Linux. To download it: https://filezilla-project.org/. Select 'Download FileZilla Client', then under 'Download FileZilla Client for Windows (64-bit) ', click a second button with the same 'Download FileZilla Client' name again. Click Download FileZilla on the left column. When downloading the 64-bit version you may receive a 'Virus Detected' message. On the web this is apparently a problem being experience by many and is being worked by FileZilla.org. Instead I downloaded the 32-bit version. Click 'Show other options' at bottom of screen (i.e. https://filezilla-project.org/download.php?show_all=1). Select 'FileZilla_3.48.1_win32_setup.exe'. It will download, than ask you to 'Open file', which will begin installation. Click through the install screens (Allow change to computer, License Agreement, etc.) taking defaults. When done FileZilla opens. We'll use it later to transfer files from the PC to the rPi and back.



Local site: C:\Users\dahae\

dahae
Default

Filename	File...	Filetype	Last m...
..			
.cisco		File fol...	12/21...
3D Obj...		File fol...	6/12/...
AppData		File fol...	12/8/...
Applica...		File fol...	6/16/...
Contacts		File fol...	6/12/...
Cookies		File fol...	12/13...

8 files and 27 directories. Total size: 10,754,481 bytes

Filename	Fil...	File...	Last ...	Per...	Ow...

Not connected to any server

Not connected.

Queue: empty

To connect to the rPi with FileZilla, enter:

Host	192.168.42.1
Username	pi
Password	raspberry
Port	22

MATLAB

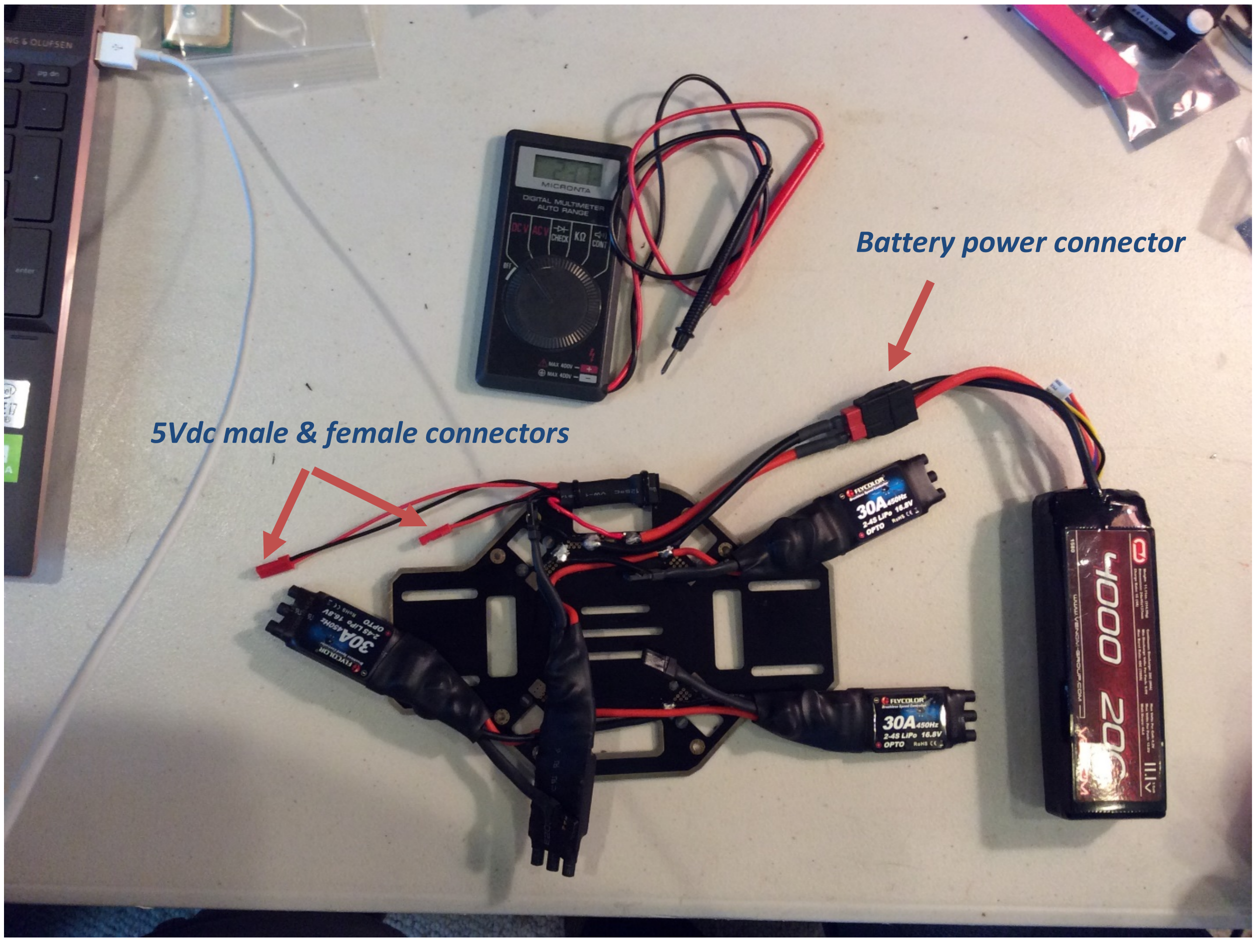
If you have access to a current license, install Matlab 2019b or later. This will be used in the plotting of real-time telemetry. If you are unable to install Matlab, this is not a problem. You will be able to store and plot telemetry data using other tools, just not in real-time.

2. ELECTRONICS PRE-ASSEMBLY AND TEST

Pretest of the power harness and 5 Vdc regulator

Lay the harness and female red tipped 5 V supply connector out on a table. Attached the provide test cable, a female 5V red connector with black and red wires, to the red male 5 Vdc connector coming from the base plate. Make sure that the wires from the test cable do not touch the other. You may consider using scotch tape to hold the black and red wires apart on the table.

First we will check the battery, and then the Power Distribution Base Plate and 5 volt regulator. With a voltmeter check that the power from the battery is at or above 11 volts depending on the charge. Attach the T-style connector on the LiPo battery to the power distribution plate; you will hear a small charging sound as current rushes into the plate. This sound is normal.



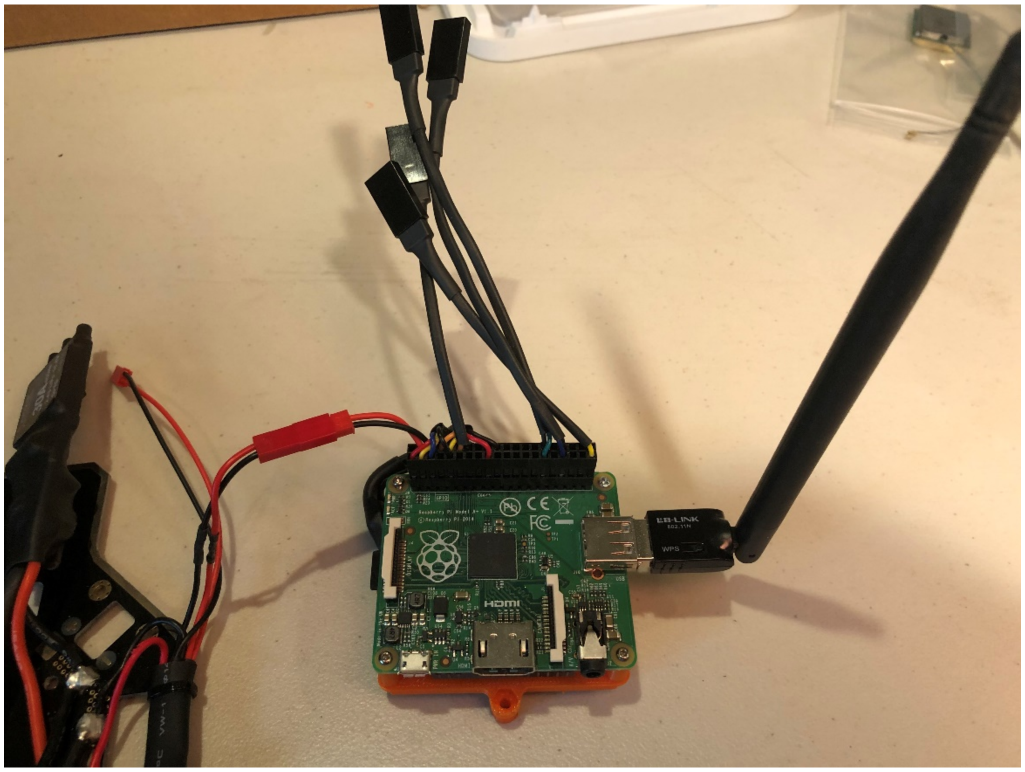
With the voltmeter, measure the voltage across the red and black test cables. You should read 5 Vdc +/- 0.1 volts.

Upon completion of the test disconnect the battery connector and remove the 5 Vdc test cable.

3. WIRELESS

Into the Raspberry Pi (rPi), the board at the top end of the Electronics Assembly, if not already installed, insert the SD card provided. It enters with the connection pads facing upward and should slide easily into the slot against a ejector spring, clicking into place. Also install the USB WiFi antenna.

Note the large dual row primary connector on the rPi. It is a 2x20 pin connector hosting the BDC motor PWM control signals, the input power source to the rPi, the IMU interface wires, and the Pi to Crius serial interface lines. Locate the input power connector, a 2-pin red male connector having a black and red wire. Connect this to the red, female mating connector coming from the Power Distribution Board as shown below.

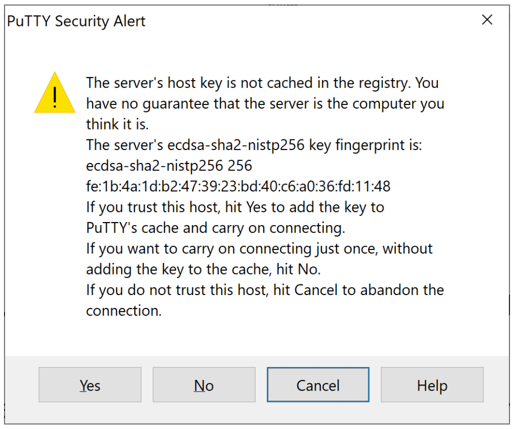


Plug in the Venom LiPo battery. You will note that the rPi green and red LEDs will flash rapidly while the rPi reads the SD Card. Flashing slows down to a slowly pulsing green after reaching the point at which the WiFi network is broadcasting.

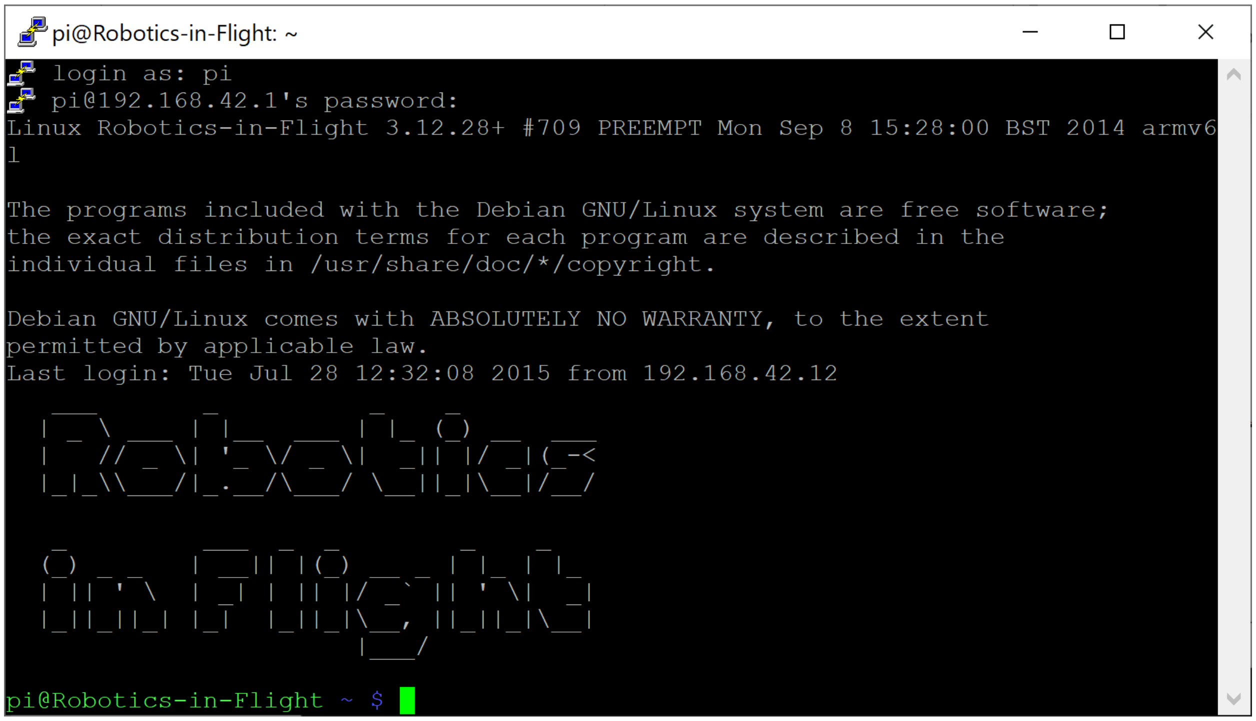
Go to your Windows Wireless Router Manager app – in Windows 10 a Globe Icon at the bottom right of the screen. Select 'Robotics_In_Flight_WAP'. Enter the security key: 'roboticsinflight' without the quotes. When connected it will show a 'Disconnect' button and state – 'No internet, secured'.

Start PuTTy. Double-click the Save Session 'PiQuad_WiFi'. If you see the window at right, click 'Yes'.

- Enter the 'login as': pi
- Enter password: raspberry


PuTTY Security Alert

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's ecdsa-sha2-nistp256 key fingerprint is:
ecdsa-sha2-nistp256 256
fe:1b:4a:1d:b2:47:39:23:bd:40:c6:a0:36:fd:11:48
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.


pi@Robotics-in-Flight: ~

```
login as: pi
pi@192.168.42.1's password:
Linux Robotics-in-Flight 3.12.28+ #709 PREEMPT Mon Sep 8 15:28:00 BST 2014 armv6
l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jul 28 12:32:08 2015 from 192.168.42.12
   ___     _         _   _
  | _ \___| |__  ___| |_(_)__ ___
  |   / _ \ '_ \/ _ \  _| / _(_-<
  |_|_\___/_.__/\___/\__|_\__/__/

   _         ___ _ _      _   _
  (_)_ _    | __| (_)__ _| |_| |_
  | | ' \   | _|| | / _` | ' \  _|
  |_|_||_|  |_| |_|_\__, |_||_\__|
                    |___/

pi@Robotics-in-Flight ~ $
```

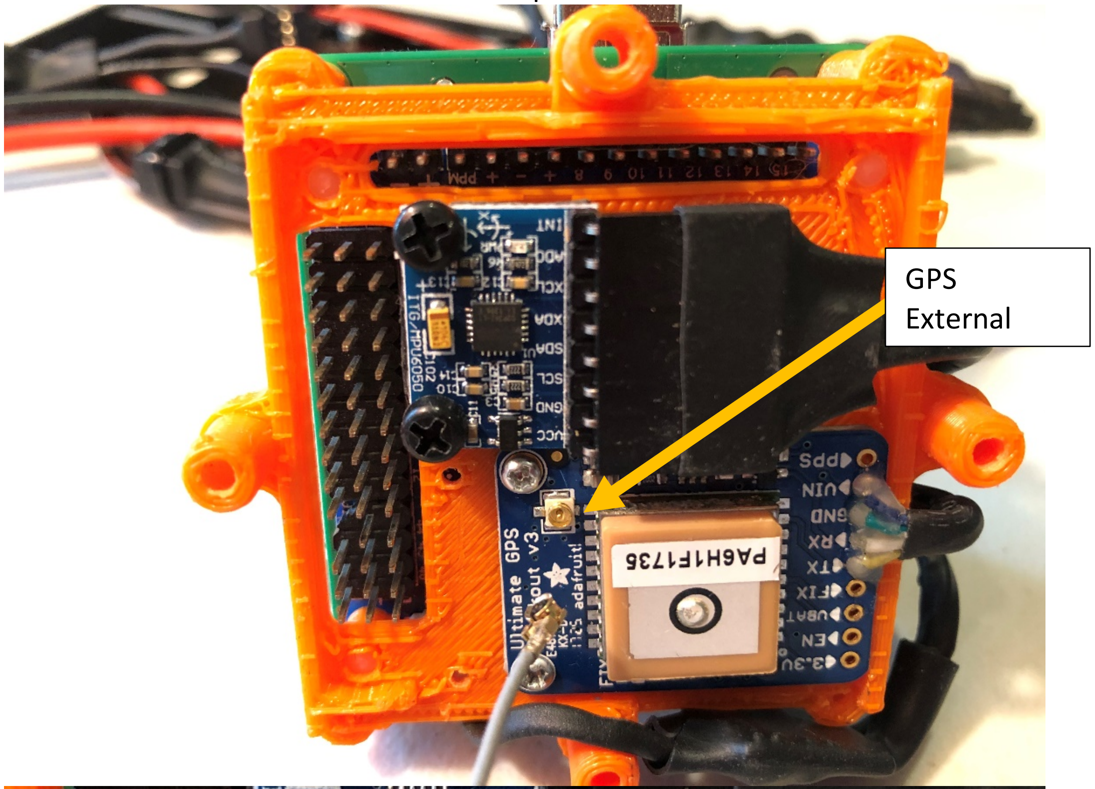
You have successfully established wireless communication between the PC and the Raspberry Pi.

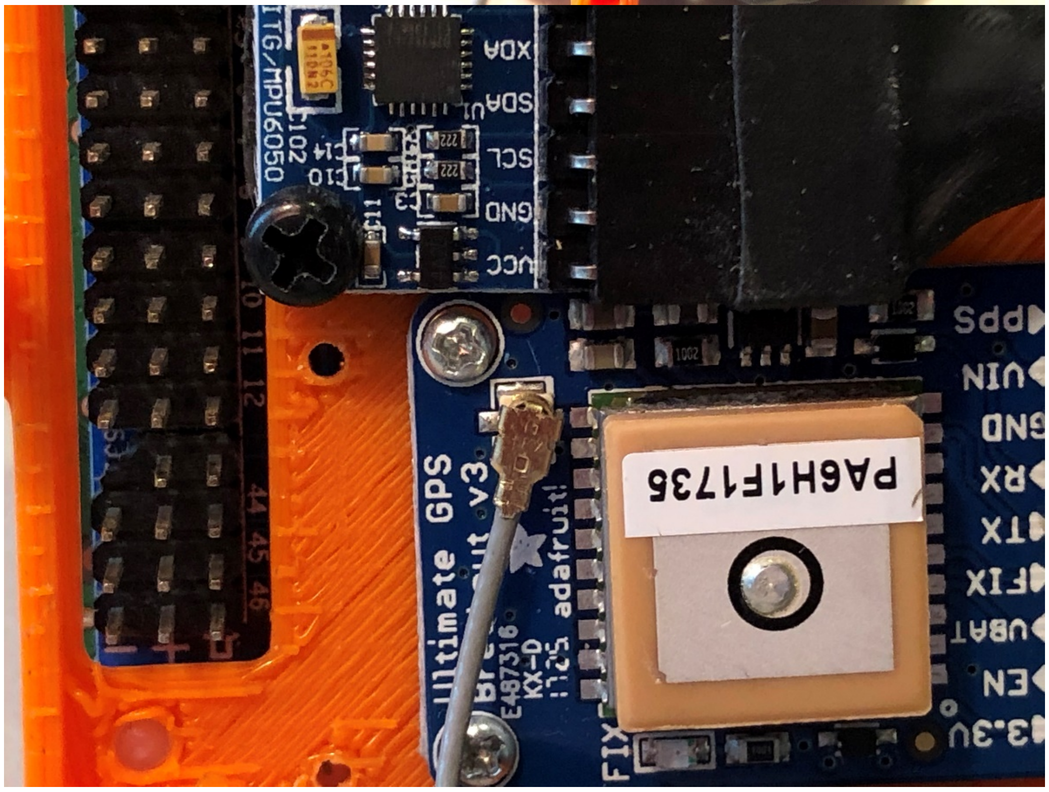
Power down.

4. HARNESSES AND PERIFERALS

GPS ANTENNA ATTACHMENT

Attach the external GPS antenna to the antenna port as shown below.



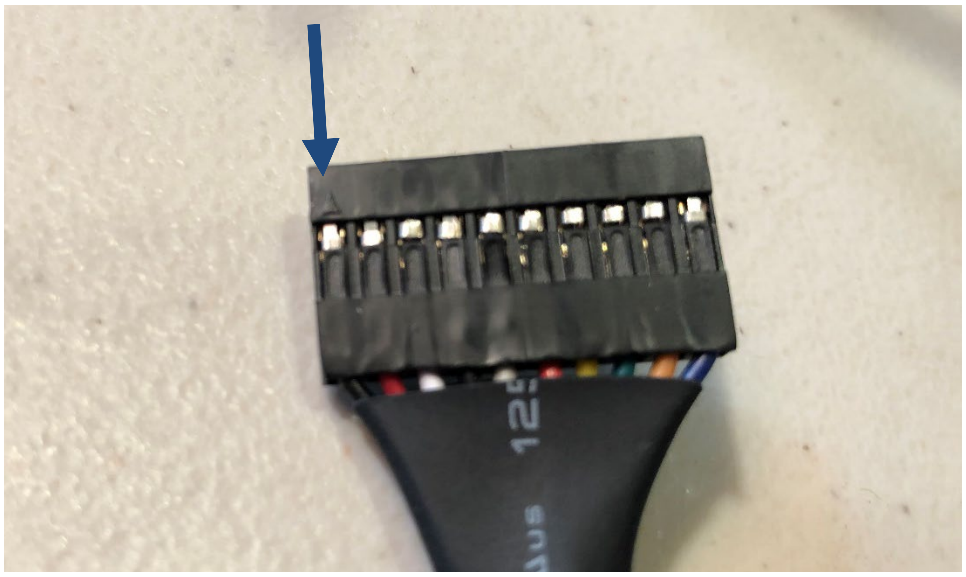


You may plug in the LiPo at this point. The Pi will power up, but the CRIUS and GPS will not because the CRIUS draws power from the Turnigy receiver, not yet connected.

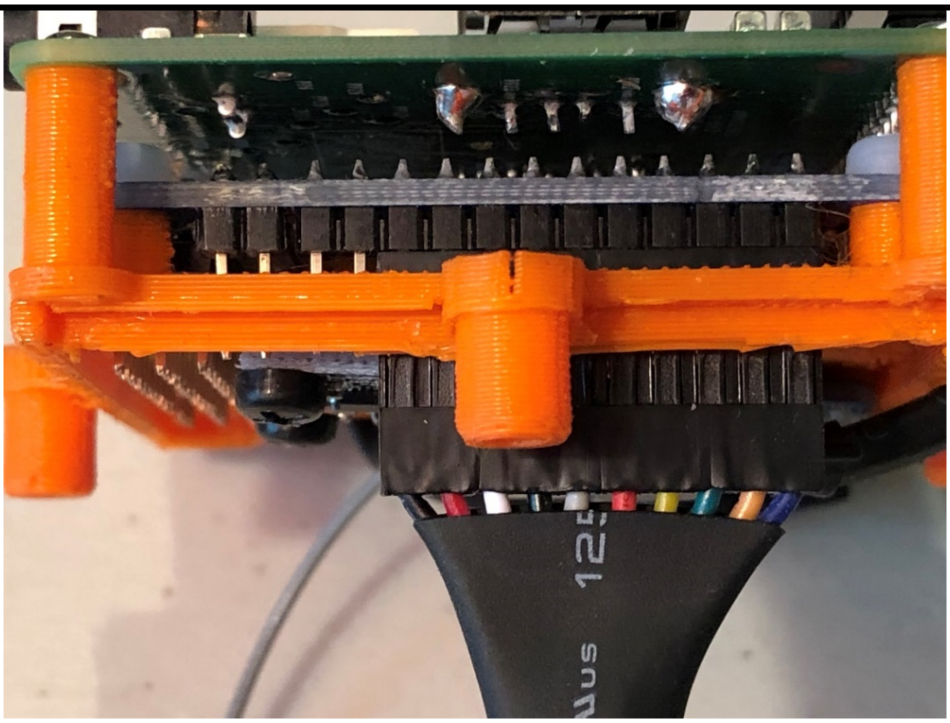
CRIUS, RC RECEIVER, AND GPS ATTACHMENT

Take the Turnigy Receiver with harness attached (if the harness is not yet attached to the receiver, go to the PiQuad Build Instructions, Harness 2: RC Receiver Cable (8 ch) – hook up the Receiver Harness as instructed).

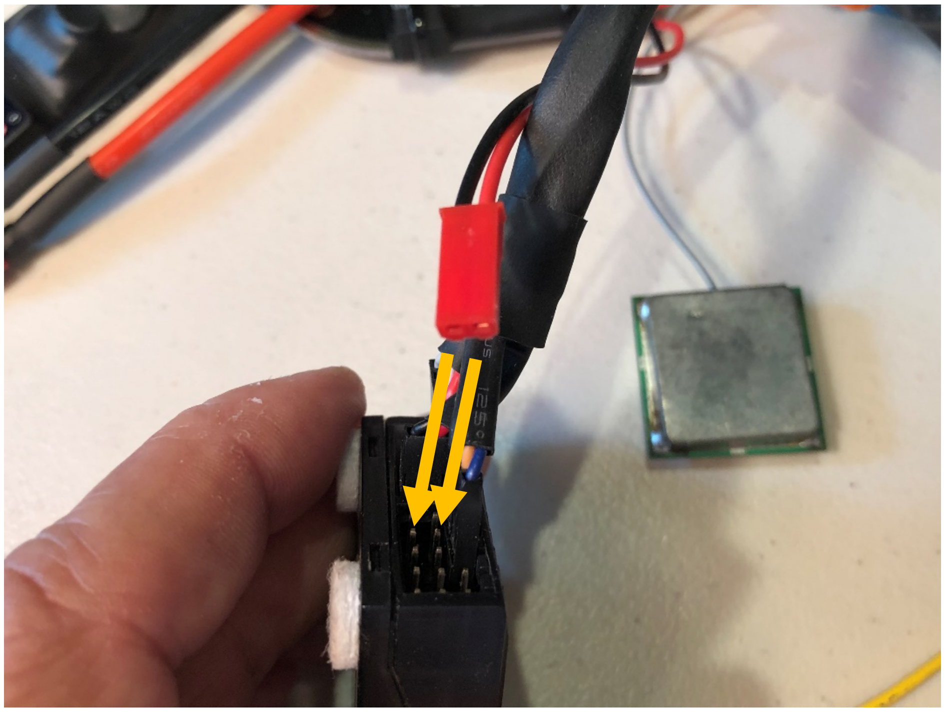
Next, attached the 8x1 connector to the CRIUS. Note the pin 1 designator on the connector:



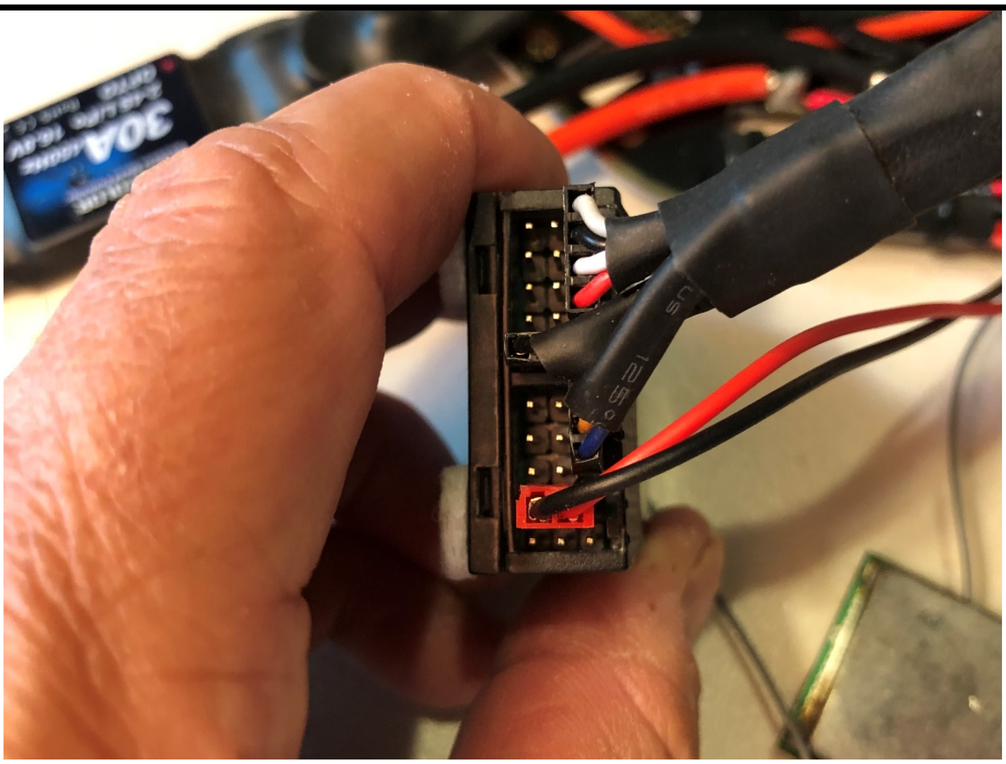
This female pin goes onto pin #5 from the left on the CRIUS single row connector as shown here. The connector attaches to pins 5 through 14, counting from the left.



Next connect the remaining female red JST 5 Vdc power connector coming from the Power Distribution Plate to the Turnigy receiver. The center row of pins is the 5-volt power pin, and the lower row the ground pin. The top row carries the received signals.

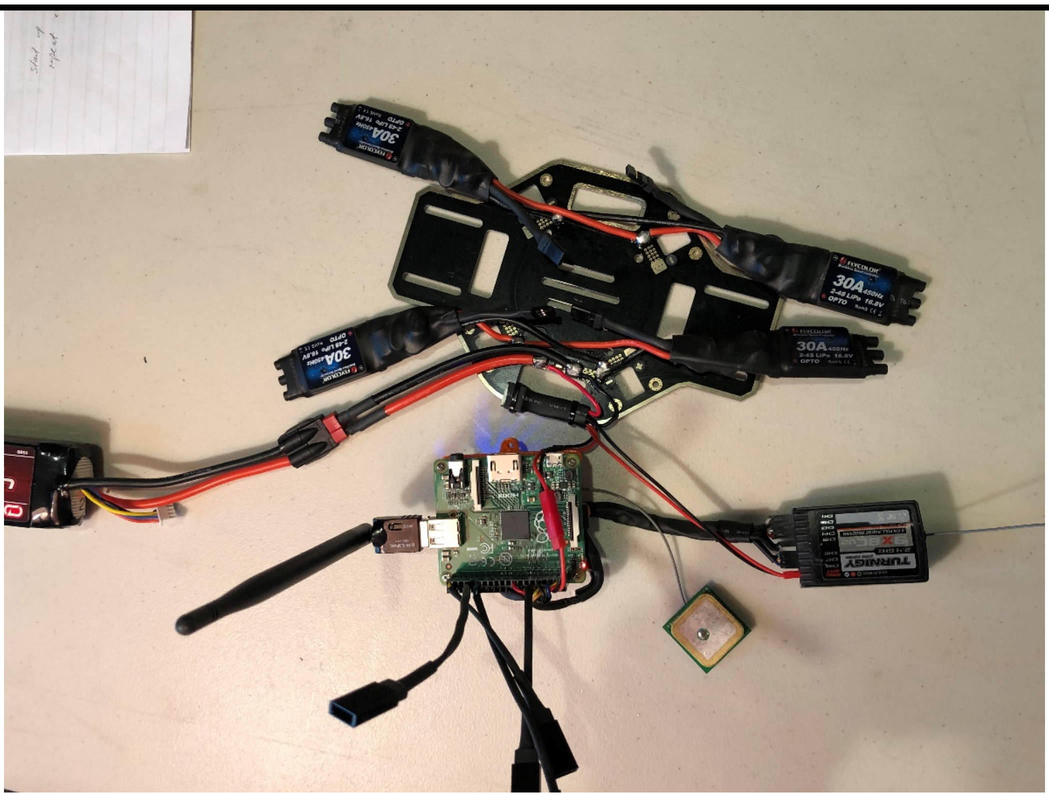


Insert the connector onto the middle and lower pins -- red to center, black to bottom – and onto the set of 3 pins labelled 'BAT' for battery:



Plug in the LiPo. You will see the rPi red LED lite up and the green LED will flash indicating SD card read/write activity during boot. You will see the red GPS LED flash and the blue LED on the CRIUS indicating that it is powered on.

Note that the GPS flashes at a rate of 1 Hz until a GPS satellite lock is established, at which time the flashing rate decreases. It may take several minutes for GPS to acquire depending on signal strength. Indoors takes longer.

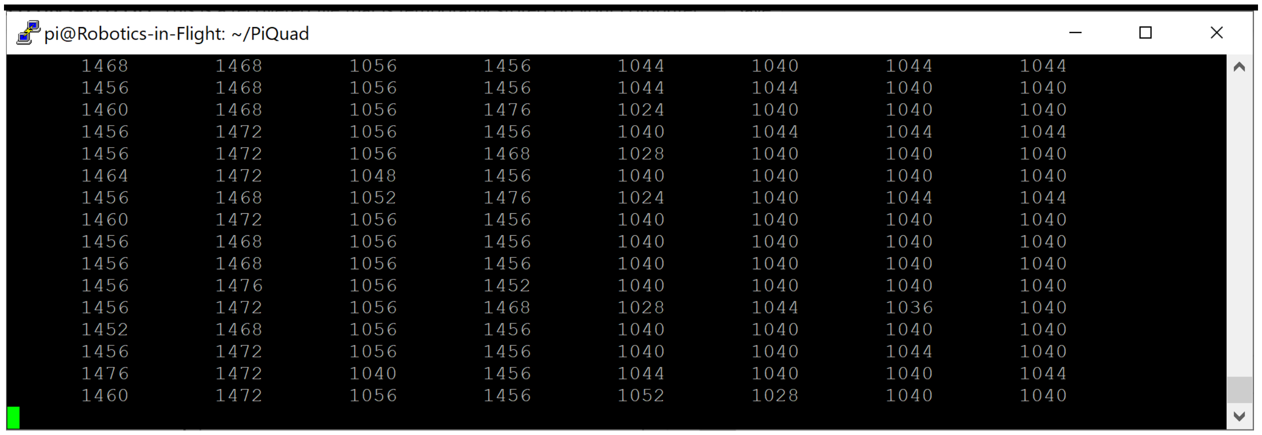


The Turnigy Receiver connection reliability is critical to the safe flight of the PiQuad. After installation on the PiQuad assembly, we recommend that the wax string and tie-wraps provided be used secure these connectors strongly to the Turnigy Receiver.

5. TESTING RC INPUTS

TURNIGY RC INPUTS

- Power up as described earlier.
- Connect WiFi to 'Robotics_In_Flight_WAP'.
- Turn on the Turnigy RC Transmitter. If bound to the Receiver, the red LED inside the receiver will turn on and remain steady; if not bound it will flash and in that case go through the binding procedure.
- Start Putty using the 'PiQuad_WiFi' setting saved earlier. Enter 'pi' and 'raspberry'.
- On the laptop keyboard hit the number '5'. This will start streaming of the RC input values to the console thru WiFi. It will look something like this:

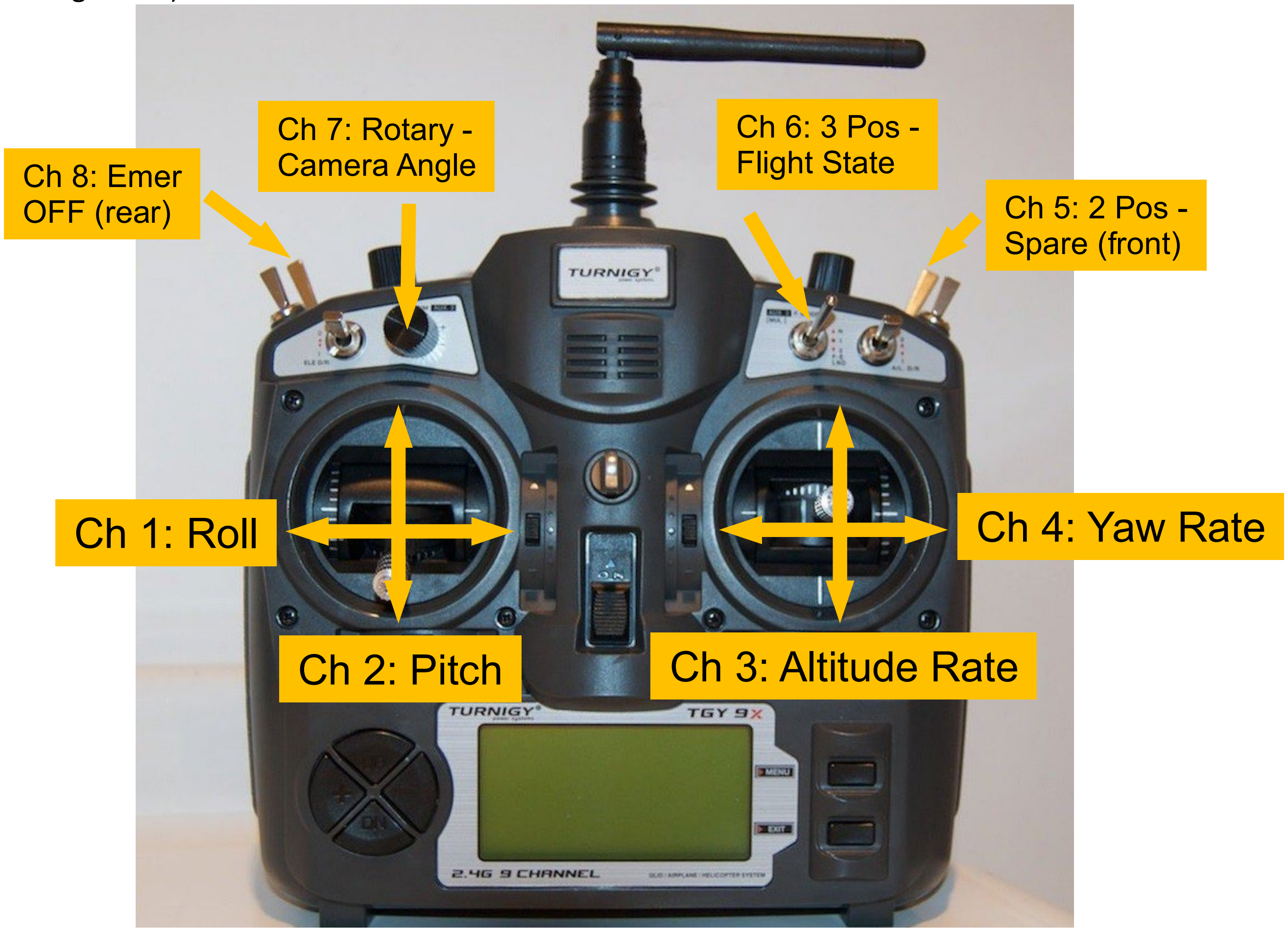


```
    1468    1468    1056    1456    1044    1040    1044    1044
    1456    1468    1056    1456    1044    1044    1040    1040
    1460    1468    1056    1476    1024    1040    1040    1040
    1456    1472    1056    1456    1040    1044    1044    1044
    1456    1472    1056    1468    1028    1040    1040    1040
    1464    1472    1048    1456    1040    1040    1040    1040
    1456    1468    1052    1476    1024    1040    1044    1044
    1460    1472    1056    1456    1040    1040    1040    1040
    1456    1468    1056    1456    1040    1040    1040    1040
    1456    1468    1056    1456    1040    1040    1040    1040
    1456    1476    1056    1452    1040    1040    1040    1040
    1456    1472    1056    1468    1028    1044    1036    1040
    1452    1468    1056    1456    1040    1040    1040    1040
    1456    1472    1056    1456    1040    1040    1044    1040
    1476    1472    1040    1456    1044    1040    1040    1044
    1460    1472    1056    1456    1052    1028    1040    1040
```

These columns are streaming the signal data received from the Turnigy 9x transmitter. Eight channels are being received:

<stk_roll> <stk_pitch> <stk_vert> <stk_yaw> <ch 5> <ch 6> <ch 7> <ch 8>

In the Turnigy units provided, the Roll and Pitch controls are on the left, and the Vertical and Yaw controls are on the right as shown here (some Turnigy's have the opposite control arrangement):



Move the Turnigy stick settings and note how the input values change. They are expected to be in the range of 1000 to 2000 counts. Move the RC Channel Controls 5 thru 8 and note how they are changing.

6. SENSOR MEASUREMENT TESTS

KEYBOARD CONTROL INPUT

Input commands from the keyboard of the PC connected to the PiQuad thru Putty and WiFi are setup as follows:

Input	Name	Description
0	Idle Mode	System in Idle Mode; no command signals to the motors. Default startup up mode.
1	Flight Mode	System in Flight Mode and will respond to commands from the Turnigy RC; commands to motors are active
2	Hold Control	The default flight control mode, in which the altitude and heading setpoints are derived from the Vert and Yaw RC inputs, and the vehicle altitude and heading are automatically controlled with on-board active control algorithms. Roll and pitch angle command setpoints are derived directly from RC input Roll and Pitch stick location inputs, setpoints sent to control loops that measure and adjust vehicle roll and pitch attitude. Default mode.
3	Return Home Control	A future control capability
4	Telemetry OFF	No telemetry data returned to PuTTy. Default mode.
5	RC Tel Data ON	RC Command telemetry data returned to PuTTy
6	Altitude Data	Altitude and other data returned to PuTTy
7	Heading Data	Heading and other data returned to PuTTy
8	Gyro Data	Gyro Rate and other data returned to PuTTy
9	Accelerometer	Accelerometer and other data returned to PuTTy

Telemetry data for Keyboard Inputs of 4 thru 9 are arranged as follows

Tel1	Tel2	Tel3	Alt_ Rate	mag_x	$\hat{V}_Z$	$\hat{b}_{AZ}$	lat	lon	gps_ alt	cv	flt_state	on-time	con_mag

The values Tel1, Tel2, and Tel3 vary as the keyboard input varies from 5 to 9. The other variables are defined as follows:

Symbol	Units	Description
Alt_Rate	-	Altitude rate input command directly from RC Ch. 4
mag_x	-	Magnetometer flux reading in x (forward) axis direction
$\hat{V}_Z$	m/sec	Vertical velocity estimate from Nav filter
$\hat{b}_{AZ}$	tbd	z-axis accelerometer bias estimate

lat	degrees	GPS latitude north
lon	degrees	GPS longitude (negative is west)
gps_alt	meters	GPS altitude above sea level

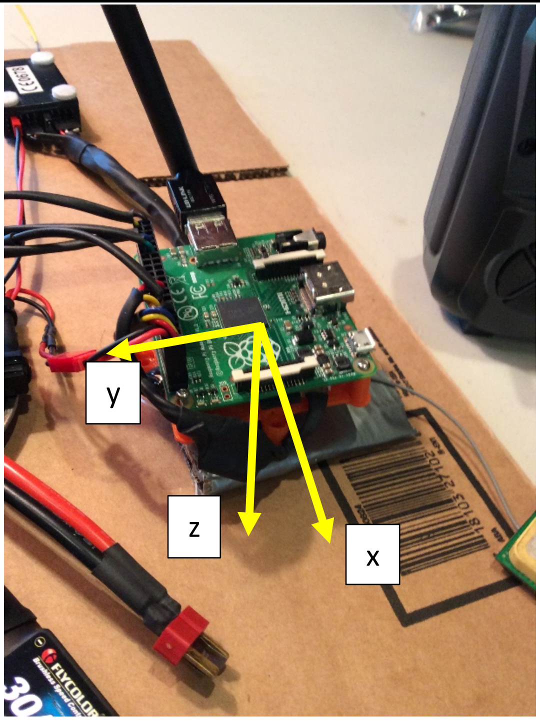
When 'Input' selections are 5 thru 9 the changeable telemetry settings are:

Input	Tel1	Tel2	Tel3	units
6	Baro-alt	Filtered-alt	Alt-setpoint	meters
7	Measured heading	Filtered heading	Heading setpoint	radians
8	Gyro rate x	Gyro rate y	Gyro rate z	deg/sec
9	Accel x	Accel y	Accel z	G's

GYRO ANGULAR RATE

- Turn on the Turnigy.
- Plug in the LiPo.
- Connect WiFi to Robotics_In_Flight.
- Bring up the system and PuTTY. We first must kill the instantiation of PythonPilot that starts automatically (PiQuad_main_auto.py). This one does not output telemetry data to Putty. To do this we use the Linux kill command:
  - Type in the command 'top' which displays all active threads
    - Read the thread number for the instantiation of 'python' and remember it
    - Cnt C out to stop 'top'
  - Type in:
    - kill -9 <the thread number -- i.e. 2477>
    - This ends that copy of python; now a new can be started, the one that outputs telemetry.
- ~~If not already, change directory -- cd PiQuad~~
- The unit comes up in directory /home/pi. We will start PythonPilot from this directory.
- ~~Start the PiQuad controller – sudo python PiQuad_main_200625.py~~
- Enter into Putty at the prompt: sudo /home/pi/PiQuad/PiQuad_main_201005.py
- Enter keyboard input '8'. This will output the IMU gyro rates.

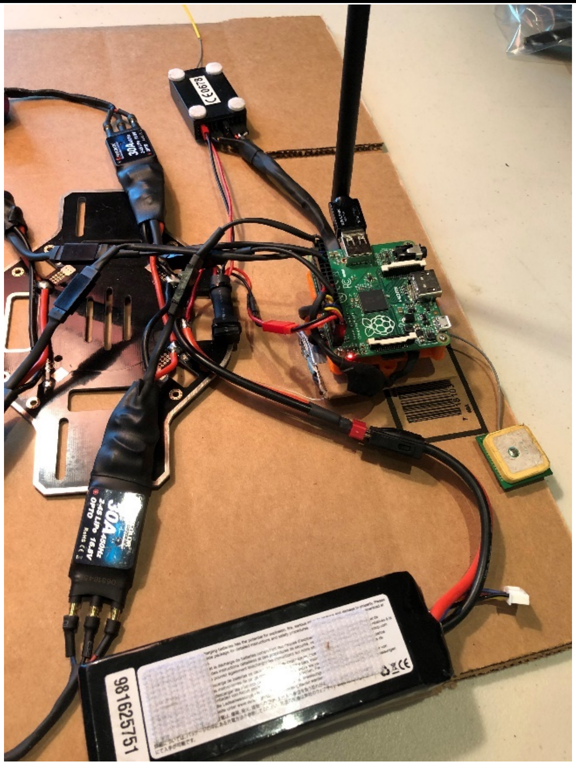
Angular rates obey the right-hand rule. The coordinate frame is shown below, with x forward, y right, and z down:



Rotate the electronics assembly by hand and note that the gyro rates vary with signs having the correct polarity and magnitudes (deg/sec). Also, note that gyro biases are present in these raw outputs. (These are measured during IMU Calibration, described in the *System Setup* documentation, to be provided shortly.) Leave the setup running for the next test – the Baro.

BARO ALTIMETER

Place the Power Distribution Assembly, Battery, RC Receiver and Electronics Assembly on a piece of cardboard or something else rigid enough to support the load. Possibly add a support piece of cardboard so that the Electronics Assembly is more level.



With the flight controller running (PiQuad_main_200625.py) and the Turnigy on, hit key ‘6’ to output telemetry containing the altitude data. The first column is the baro altimeter reading relative to the initial altitude measurement measured at power on; i.e. this the altitude above ground.

Raise and lower the platform and note that the baro changes. The units are meters so you can probably cause a change of 2 to 3 meters overall.

HEADING

Keeping the system on the board, hit key ‘7’. This will output the heading data uncalibrated in radians. The first column will vary as you rotate the assembly. Because it is not calibrated, the results may not make sense. We’ll recheck after calibration.

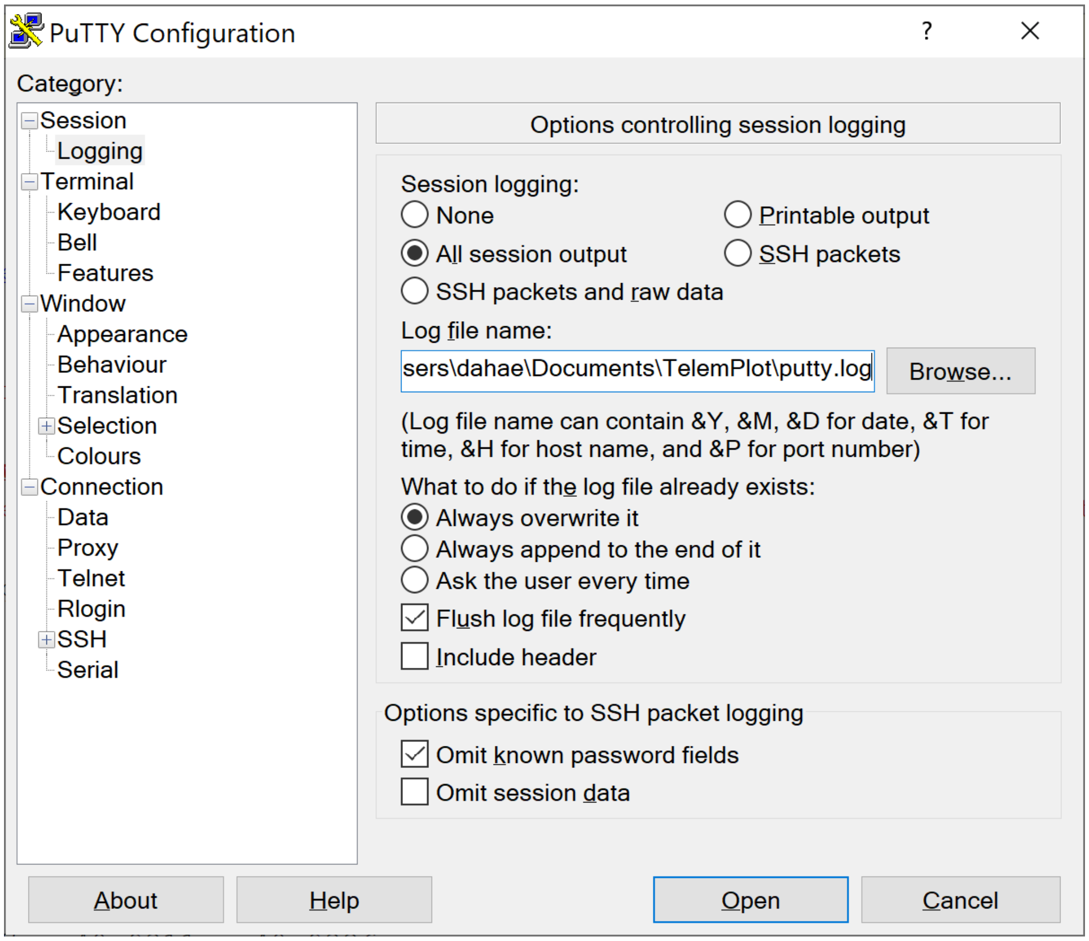
7. GPS MAPPING AND TELEMETRY PLOTTING

Setup Putty for logging of the data that is being written to the console.

Start PuTTY. Go to Session->Logging:

- Session logging: All session output
- Log file name: Enter the location and name of the file to be written
- Save the PuTTY Session file

Create a folder that will contain the matlab script (telemetry_wGPS.m), and the map file (main.png). Use this location for writing the file 'putty.log' in the Log file name location below.



Session logging:
None
[✓ All session output]
SSH packets and raw data
Printable output
SSH packets

Log file name:
sers\dahae\Documents\TelemPlot\putty.log

(Log file name can contain &Y, &M, &D for date, &T for time, &H for host name, and &P for port number)

What to do if the log file already exists:
[✓ Always overwrite it]
Always append to the end of it
Ask the user every time

[✓ Flush log file frequently]
Include header

Options specific to SSH packet logging
[✓ Omit known password fields]
Omit session data

Setup the matlab program (telemetry_wGPS.m) for GPS trajectory plotting:

- Create a .png file containing a Google aerial picture of the location where you will be flying
- Point to the file on line 49
- Update the GPS lat / lon axis range on line 50

Telemetry data is plotted in the time domain figure and on the GPS map.

Note to Ron: We have created a version of the telemetry plotter -- telemetry_wGPS_Ogan.m. It is setup for your home location. Also, the *System Setup* documentation contains a section describing the creation of maps for any location to be used for flying.

Now with the realtime telemetry plotting underway, go back and repeat the Sensor Tests of Section 6, this time observing the sensor data on the stripchart plot.

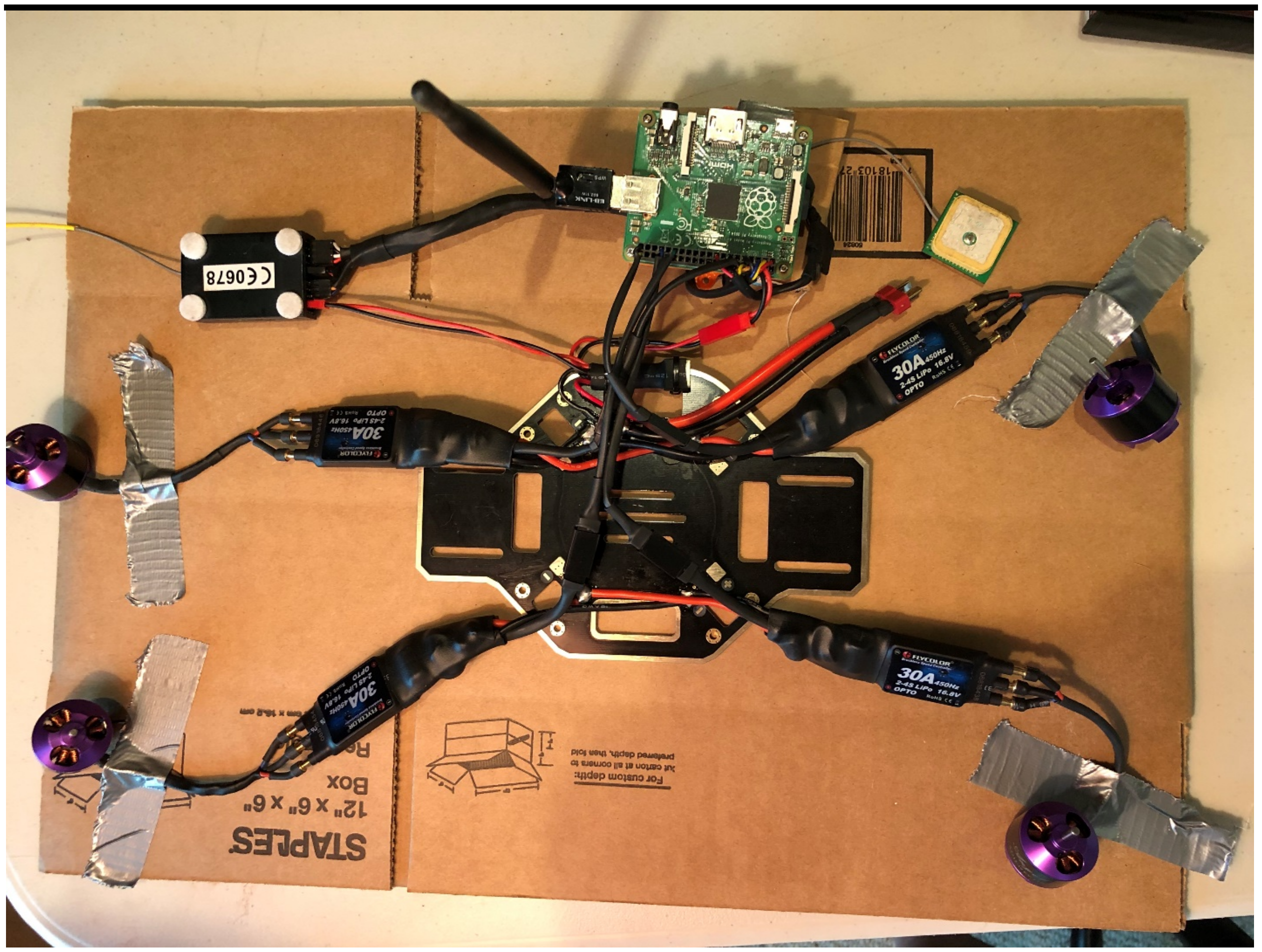
8. ESC & MOTOR TESTING

Attach 4 motors – place the black wire in the middle connector on the ESC, and the red and green on the outside connectors. No need to push in completely since we will be removing.

Attach the ESC PWM connectors as shown below. Note they are keyed and will go together without much force. The connection pattern – yellow BR, blue BL, green FL, orange FR.

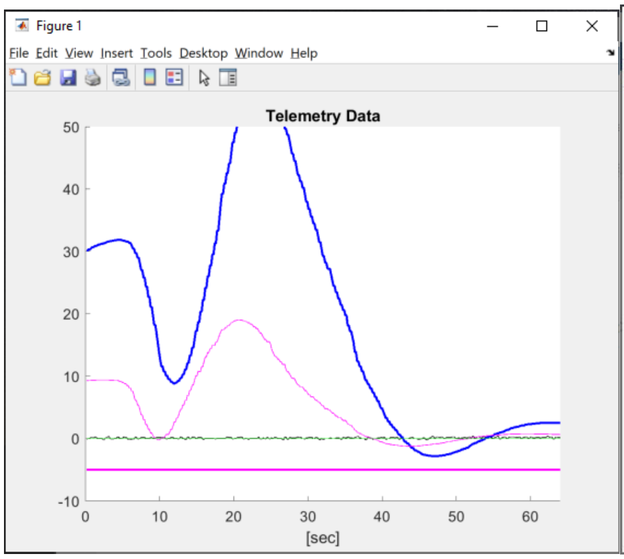
BR – back right
BL – back left
FL – front left
FR – front right

Tape the motors to the support plate so they will not rub when spinning.



Plug in the battery.   Hook up to WiFi Robotics_In_Flight.   Start PuTTy.

Start sending telemetry – key '6'.   Start Matlab running 'telemetry_wGPS_Ogan.m'.   You will get the following.



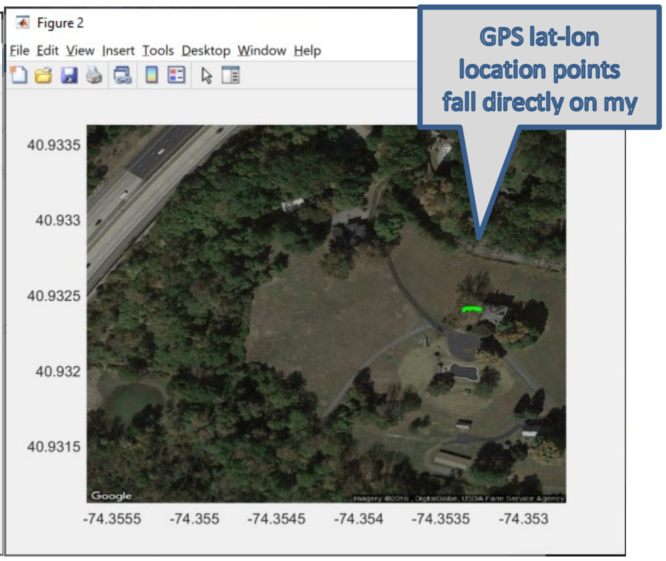
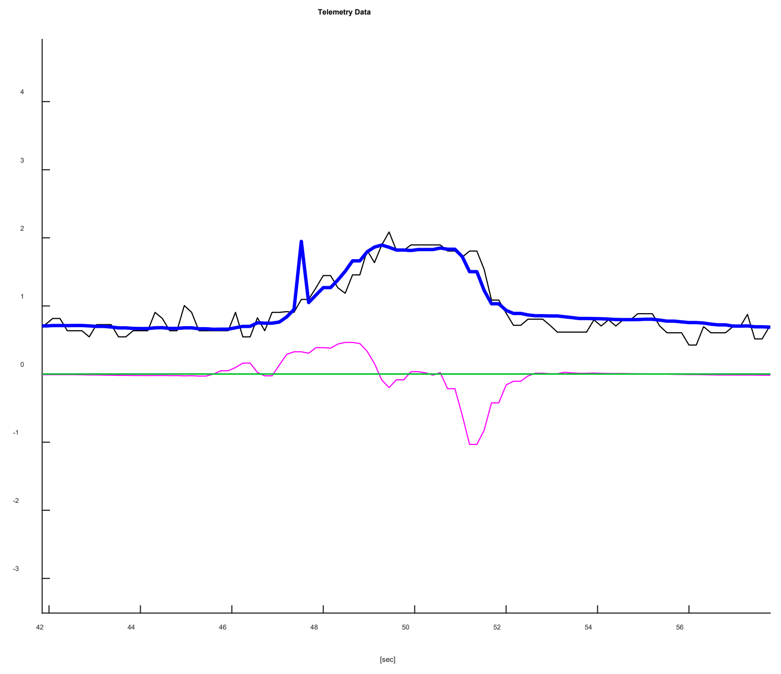


Figure 2
File Edit View Insert Tools Desktop Window Help
GPS lat-lon location points fall directly on my
40.9335
40.933
40.9325
40.932
40.9315
-74.3555
-74.355
-74.3545
-74.354
-74.3535
-74.353
Google

In the telemetry:

- Filtered Altitude Estimate – Dark blue
- Vertical Velocity Estimate – Thin magenta
- Altitude setpoint command – Thick magenta
- Raw Baro Altitude -- black

The green is just a line at zero altitude. Here is a close-up:



Next perform the motor test. When you start 'PiQuad_main_200625.py' you will not a short kick to the motors (20 msec full on). This is a priming operation.

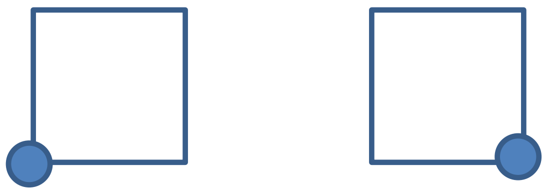
While streaming altitude telemetry ('6'), the right most column sends out the magnitude of the control levels, reaching a maximum in the 2000 count range. You cannot command a state change to Flight Mode unless this value is less than 100. It is blocked when above 100 as a safety precaution – don't engage the motors is a high control action is being requested.

Switch to Flight Mode by hitting the '1' key. The motors may begin to rotate in part because the unit is not calibrated. You can attempt to bring all to a halt by adjusting the heading command, right stick left and right.

Move the vertical stick to a position above the midpoint, and note that the Altitude Setpoint goes up. When above the current altitude the motors begin to increase in speed. Adjust the Altitude Setpoint to a moderate to slow motor speed.

Grab the Electronics Assembly and inject some roll and pitch. Note that the motor pairs begin to spin in reaction to oppose the motion. Recall that the assembly is rotated 180 degrees from its orientation when assembled on the airframe.

Try the Emergency OFF stick locations:



This will kill the control signals to the motors.

This concludes the pre-assembly test process.

You can proceed to the Airframe Assembly steps described in – PiQuad™ Build Instructions – '200609_PiQuad_Build_Instruc_rev-.docx'

9. CALIBRATION

After the unit is fully assembled and all assembly integration and tests procedures defined herein passed, it is time for initial and full calibration.

APPENDIX A: Acronyms

The following table provides definitions for terms relevant to this document.

Acronym	Definition
IO	**Input-Output**
IMU	**Inertial Measurement Unit**
RPAS	**Remotely Piloted Air System**
UAS	**Unmanned Air System**
UAV	**Unmanned Air Vehicle**
FAA	**Federal Aviation Administration**
RIF	**Robotics In Flight™**

APPENDIX B: Disclaimer

If you're in doubt of your skills as a UAV, Drone, multi-copter or RPAS pilot, or are unsure of your equipment or piloting skills, do not fly.

Multicopters are inherently dangerous and should not be used by anyone without professional training and experience. It is the responsibility of the user to decide if the equipment is suitable for your intended use.

You (purchaser/user) are responsible for following all federal, state, and local laws which may regulate the use of Remotely Piloted Air Systems operated in your area.

Purchaser/user assumes all liability for damages to property and persons from the equipment.

Purchaser/user confirms that they will operate UAV's, Drones or RPAS in accordance with FAA rules.

Purchaser/user assumes all liability for improper use of the equipment.

Purchaser/user is responsible for proper configuration and maintenance of the equipment.

It is highly recommended that users of this document and/or pilots of the PiQuad have proper liability insurance coverage. RC aircraft are not covered under standard liability insurance. Please check with your insurer regarding the correct insurance coverage.

Robotics In Flight LLC shall not be responsible for any warranty claim due to misuse, crashes, damage, incorrect setup/configuration, or other negligent behavior.

In no event shall Robotics In Flight, LLC be liable for special, indirect, incidental or consequential damages related to the use of this document. Indemnity: users of this document shall indemnify and hold harmless Robotics In Flight, LLC, its officers, directors, agents, representatives and employees from any and all claims, liabilities, damages, and expenses (including attorneys' fees actually incurred) on account of death or injury to any person or damage to any property arising from or in connection with any goods supplied. This indemnity shall apply without regard to whether the claim, damage, liability, or expense is based on breach of contract, breach of warranty, negligence, strict liability, or other.