To mention:

* Fswebcam
* Nohup
* File cleaner
* PiCam (Python)
* Launcher
* Python script to make sure things are running
  + <http://pastebin.com/Y1P0xKfi>
    - Anonymous ☹
* Watchdog
  + <http://blog.ricardoarturocabral.com/2013/01/auto-reboot-hung-raspberry-pi-using-on.html>
    - Referred from Adafruit
  + <http://www.raspberry-pi-geek.com/Archive/2015/09/Making-your-projects-more-reliable>
    - Maybe what I actually used
  + <https://www.raspberrypi.org/forums/viewtopic.php?f=29&t=147501>
    - Now I think this is what I really used in the end…

**DISCLAIMER:** As with the code before, it’s worth mentioning that I’m entirely self-taught, and I’m certain that there are many more elegant approaches to what I’ve done here. Everything works, and I can call that a victory, but I’d love to hear any suggestions or thoughts.

**Raspberry Pi 3 (Python Media Control Code)**

The main Python camera control script only performs a few functions, but is responsible for coordinating most of the actions that the Raspberry Pi performs on Icarus’ journey. There are 4 specific capture modes that are active over the course of the flight, each responsible for controlling photo and video capture of various durations from each of the 3 onboard cameras. The parameters for each phase are optimized (by educated guessing) to capture the highest quality media at each flight phase. For example, the down-facing webcam captures video down at the launch site as it ascends, shutting off at 5,000ft – hypothetically, it should look pretty cool watching the launch team get smaller and smaller as Icarus rises. When video capture is active, booleans and delays are in place to ensure that no 2 webcams capture video simultaneously nor attempt to capture a photo while currently capturing video, which can cause an over current-based reboot event or an error to be returned, respectively.

When the main script is first run, and before entering the main/permanent loop, a text file in the root folder is referenced (“mode\_file.txt”) to determine the mode in which the program will begin (i.e. takeoff capture, main phase, peak capture, etc.). The functions for photo and video capture are separated, with an arguments being called for each to indicate the camera to use for capture and duration, if necessary.

The libraries used were fairly standard, with a few exceptions. The Python “picamera” library allows easy media capture from the RPI camera, and was a straight-forward solution. Media capture using this library comes directly from the script, with no external handlers necessary. The webcams, however, required slightly more work to integrate. Luckily, the RPi recognizes most mainstream webcams natively, assuming you’re running Raspbian. Thanks to Dave Akerman (as with many other things), I use “fswebcam” as the still image capture solution for the webcams and was able to do so with minimal setup (<http://www.daveakerman.com/?p=592>).

Shell script breakout

Flie\_cleaner

PiCam delay/fswebcam delay

**Raspberry Pi 3 (Shell Scripts)**

STUFF & THINGS ABOUT SHELL SCRIPTS

All photos, videos, and logs are stored to dedicated folders that can be cleaned and reset for a new launch by running a Python script (“file\_cleaner.py”).

***FILECLEANERCODE***

**Raspberry Pi 3 (Crontab)**

STUFF & THINGS

**Raspberry Pi 3 (Accessory Scripts & Loose Ends)**

STUFF & THINGS