NoseCam is something I created to stabilize (de-spin) high power rocket flight videos. It also forms the basis for a Vertical Orientation Control (VOC) system. What I am sharing in this distribution does NOT contain info on the VOC subsystem. That being said, if you manage to successfully build and fly a NoseCam using what I am sharing here then you are 90% of the way to having a VOC system. Almost everything you need is already here. You “just” have to add the needed components to the upper nose (servo(s), winglets/canards, etc.) and write some additional code.

The Rocketry Form (TRF) is where I have been posting everything I’ve been doing with NoseCam. Here are some useful links to posts:

https://www.rocketryforum.com/threads/a-new-spin-on-vertical-orientation-control.174525/

https://www.rocketryforum.com/threads/falcon-nosecam-build-thread.174591/

https://www.rocketryforum.com/threads/two-stage-flight-with-video-stabilization.172553/

I plan on continuing to use TRF as the only location where I will answer questions. So if you have a question, post it to the “Falcon NoseCam Build Thread” on TRF.

What I have provided in this distribution is intended to be a **STARTING POINT**. It is **NOT** a complete, soup-to-nuts set of instructions that would allow anyone to build a NoseCam. That would have been far too much work to create. (Sorry) Simply put, if you are going to build one of these things, you are going to need to maintain and tune it. And before you can do those things, you need to understand it. So being an active participant in the overall process (and not just a user) is critical. I’m sure there are things missing in this distribution (such as overall schematics LOL). I’m also very sure there are things wrong that I haven’t found yet. This has been and always will be a research project for me. It is also a work in progress. I consider the Arduino code to be the closest thing there will be to an instruction manual. (Again, sorry) And unfortunately, I write mediocre code at best !

In truth, I don’t have any expectations that someone will build an exact copy of what I have shared here. What I do hope is that like-minded folks can take what I have shared and use some part of it (code, design concepts, etc.) to build something cool for themselves. That being said, if you do try to build a NoseCam using what I have provided here, then you will already need to be VERY GOOD with the following skills:

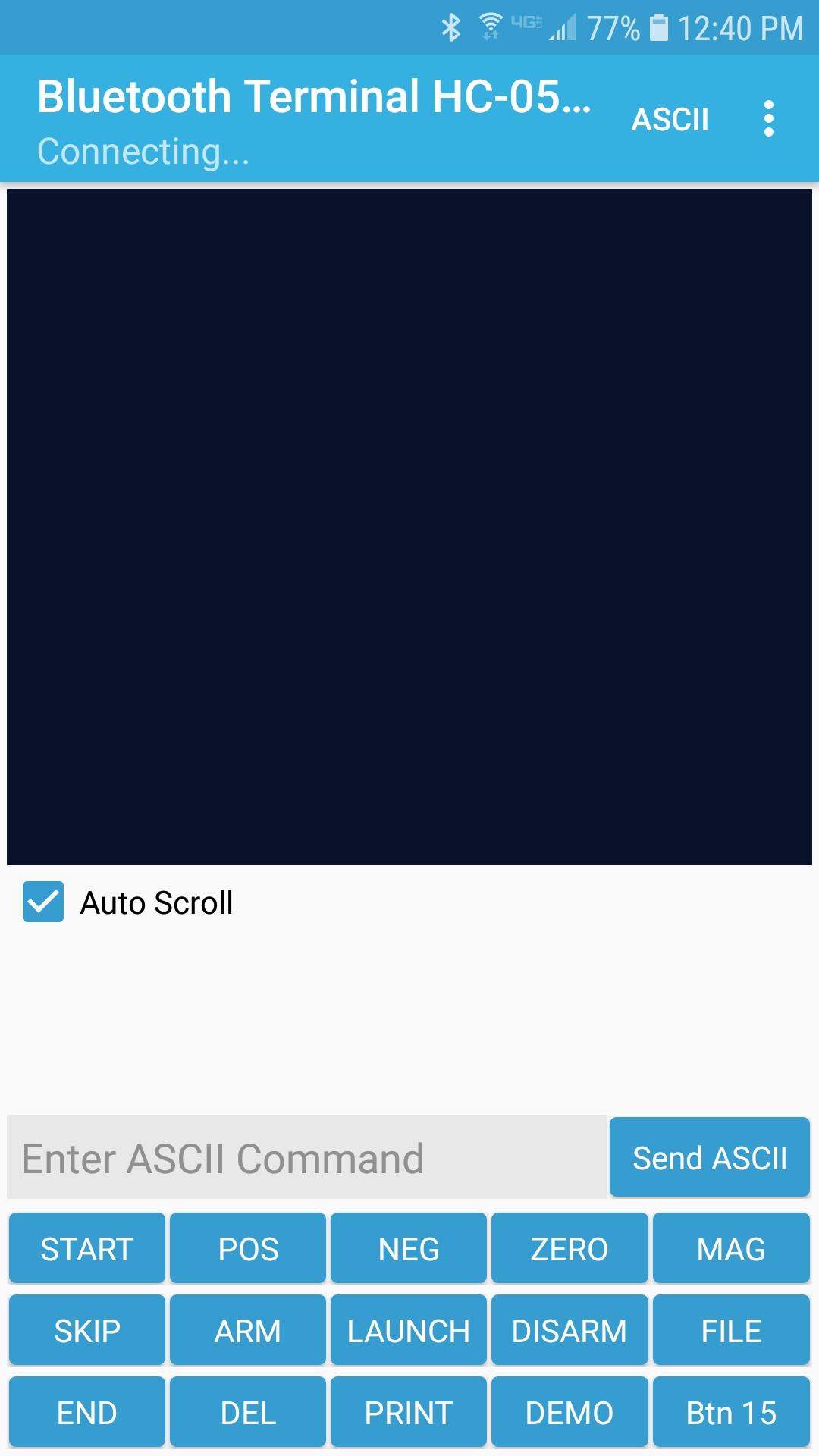
* High Power Rocketry (HPR) – I have been building and flying for many years and have performed hundreds of high power / electronic dual deploy / GPS tracker flights. I’ve also done a lot of high power two-stage flights. The NoseCam by itself can be a handful to use when you first start out, so it helps if the “rocketry” part of using it is very familiar to you.
* 3D CAD (I use Fusion 360)
  + Modifying / tweaking CAD part files to meet your particular needs
* 3D Printing (and modifying printed parts to fit)
  + These are structural parts going into a rocket. They need strength. PLA printers need not apply.
* Arduino coding and debugging (LOTS and LOTS of debugging)
  + Be prepared to spend a **LOT** of time mastering the SimpleFOC Arduino library
* Overall mechanical design and fabrication skills
* Electronic design, assembly and debugging (LOTS and LOTS of debugging)

Things you will need:

* Money – I’ve spent several thousand dollars on this. You will spend less, but you will still spend a lot. Plan on buying two or three of everything needed. You will fry / break / trash / crash a lot of stuff. If you buy one set of everything and get it to work and it lasts more than one flight you are a genius.
* Time – I’ve easily spent over 1000 hours developing this. I’ve built several NoseCams, and even with all my experience it takes about 50 hours to build a new one. For someone starting from scratch you should plan on many months of fabrication, learning and mistakes.
* Access to a machine shop / maker space is really helpful. You will definitely need a 3D printer and an electronics assembly bench. Standard debugging tools (DVM, O-Scope, solder rework tools) are a must.

Other stuff:

* There are a lot of “READ ME” files in the various folders. You really want to read all of them. It will save you a LOT of time.
* My expectation is that reading the NoseCam Arduino code will provide enough of an “instruction manual” to get you going. My commenting isn’t great, but it is there. I know it’s a lot of code. But understanding it will be good for you. Just like eating kale ….
* I use Bluetooth classic and my Android phone to communicate with the NoseCam. You can’t use the NoseCam without some way of communicating over Bluetooth. There’s a wonderful Android phone app called “Bluetooth Terminal HC-05 Pro” that I use.



* The NoseCam Arduino code has data logging for every flight. The data is stored to the 2MB flash on the Adafruit Metro M4. Unfortunately ………….
  + Getting the data off of the NoseCam and into a useful format like Excel is ugly.
    - I connect to the NoseCam with my laptop via Bluetooth
    - I open the Arduino IDE and use the serial monitor to enter the needed commands to get the NoseCam to start streaming the logged data from a particular file to the laptop serial monitor
    - Once the file is done streaming to the serial monitor, I copy-and-paste the data from the serial monitor to a .txt file on my laptop
    - I then open the .txt file as a .csv file in Excel
    - I told you it was ugly
  + I download data after every flight and then delete the data file from the NoseCam since there is only 2MB of flash storage there. I have no idea how many data files can be written to the flash before some weird file management error occurs. You have been warned 😊
* I’ll put this info here and repeat it in other places
  + Use the 1.7.5 version of the Adafruit SAMD board library for the Metro M4. Do NOT use the latest version. It will not work. I have no idea why.
  + Use the SimpleFOC libraries I have provided it this distribution. They are an older version but they work. Do NOT try to download and use the latest version of the libraries. They will not work. I have no idea why.
    - Just FYI, my SimpleFOC libraries have two slight modifications which make synchronizing the BLDC motor to the encoder less aggressive.

That’s it ! Good luck !!!!!!!

Steve Gregorski