Mount controller for satellite tracking

Optical Makerspace

Revision A – 27 July 2023

OPTICAL MAKERSPACE

sc robotics

Introduction

This sub-activity evaluated the need to develop an electronics motor controller for amateur/prosumer astronomy mounts in order to be used for satellite tracking in optical communications. The end goal was to lower the access barriers to this community in a similar way to what SatNOGS has achieved in the radio communications one. This way, the innovation of this sub-activity will be to enable low-cost mounts transformation into optical communication and optical tracking capable mounts.

Choosing a mount

The first step in our roadmap was to pick the right mount in order to familiarize ourselves with all the different tools. To do that, after a pre-selection (see table 1), we had talks with a few prosumer users of mounts with similar requirements to the one we were targeting. Together with them we decided to use the Skywatcher EQ6-R PRO because the specifications matched our baseline requirements but most importantly, because it seems to be the entry mount for these advanced users. The latter was a key factor to bear in mind because one of the goals of this project is to deliver something the Community can adapt and reuse.

Mount Model	**Maximum Pointing Accuracy (arc sec)**	**Pointing Resolution (arc sec)**	**Maximum Slewing Speed**	**Motors**	**Cost (€)**	**Type**
SkyWatcher EQ6-R PRO (UM)	300	0.14	800x	Stepper	1880	EQ
SkyWatcher Z-EQ5 PRO (UM)	300	0.25	800x	Hybrid Stepper	1660	AltAz/EQ
SkyWathcer EQ8-R PRO (UM)	300	0.03	1000x	Hybrid Stepper	4920	EQ
iOptron AZ Pro	-	0.1	512x	Stepper	2490	AltAz

Table 1. Original pre-selection of target mounts

The Skywatcher EQ6-R PRO is an equatorial motorized mount very common among prosumer astrophotographers. What's very important for this project is the support for INDI (thanks to eqmod driver), an Open Source software that allows hosts control all different pieces of an astronomical observation setup. In case we need to tweak, enhance or even understand how does tracking work (for this or other mounts), we can always refer to the source code.

First of all, to try to understand the kind of tasks we were supposed to perform, we compared the Skywatcher EQ8 and the EQ6-R PRO. Below (table 2) are the specification tables taken from the corresponding User Manuals (we highlighted in red the main differences):

Product Name	EQ6-R Mount
Mount Type	German Equatorial
Payload (Counterweights excluded)	20kg (for Astrophotography)
Latitude Adjustment Range	5° to 65°
Azimuth Adjustment Range	About ±9 °
Weight(Tripod excluded)	17.3 kg
Counterweight	2 x 5kg/ea
Tripod	2-inch stainless steel, 7.5kg
Counterweight Rod	18mm Diameter, Length 240mm + 180mm
Power Requirement	DC11~16V 4A
Motor	1.8 ° Hybrid Stepper Motor
Transmission	180:1 Worm Drive + 48:12 Timing Belt Drive + 64 Micro-step/1.8° Stepper Motor Drive
Gear Ratio	720
Resolution	9216000 Counts/Rev., approx. 0.14 arc-second
Maximum Slewing Speed	4.2 degrees/second
Tracking Rate	Sidereal rate, solar rate, lunar rate
Tracking Mode	Equatorial mode
Auto-guiding Speed	0.125X, 0.25X, 0.5X, 0.75X, 1X
PEC	100 Segments Permanent PEC
Hand Controller	SynScan
Database	42000+ Objects
Celestial Object Catalog	Messier, NGC, IC, SAO, Caldwell, Double Star, Variable Star, Named Star, Planets
Pointing Accuracy	Up to 5 arc-minutes (RMS)

Product Name	EQ8 Mount
Mount Type	German Equatorial Mount
Payload (Rated for astrophotography; counterweights excluded)	50kg
Latitude Adjustment Range	10° to 65°
Azimuth Adjustment Range	±10 °
Weight (Tripod excluded)	25 kg
Counterweight	2 x 10kg/ea
Tripod	29.4kg
Counterweight Rod	2.6kg
Power Requirement	DC11~16V 4A
Motor	0.9 ° Hybrid Stepper Motor
Transmission	435:1 Worm Drive + 64 Micro-step/0.9° Stepper Motor Drive
Gear Ratio	435
Resolution	11136000 Counts/Rev., approx. 0.12 arc-second
Maximum Slewing Speed	3.3 degrees/second
Tracking Rate	Sidereal rate, solar rate, lunar rate
Tracking Mode	Equatorial mode
Auto-guiding Speed	0.125X, 0.25X, 0.5X, 0.75X, 1X
PEC	100 Segments Permanent PEC
Hand Controller	SynScan
Database	42000+ Objects
Celestial Object Catalog	Messier, NGC, IC, SAO, Caldwell, Double Star, Variable Star, Named Star, Planets
Pointing Accuracy	Up to 5 arc-minutes (RMS)
Resolution of Aux. R.A./Dec. Axis Encoders	17624 Counts/Rev., approx. 1.2 arc-minutes

Table 2. EQ6-R (above) and EQ8 (below) specifications

Previous tables show that both mounts are pretty similar. Apart from the price tag difference -roughly 1500€ (EQ6-R PRO) vs ~4500€ (EQ8-R PRO)- there are slight differences in the resolution (due to using more accurate motors). However, **the most important distinguishing factor is the usage of axis encoders in the EQ8**, which could give us some hints of which kind of improvements should be needed.

Evaluation - The state of the art

Issue 1 – The feedback loop

After comparing our mount with more advanced ones, we realized that the main hardware difference had to do with the motors' specifications. Particularly, higher-end mounts tend to use motors with encoders and with more steps.

Since a very stable movement is needed for optical communication purposes, **the lack of encoders** could be very significant problem for our use case because the mount can't know the current position if something goes wrong (e.g., if some steps are missed while slewing due to mechanical problems, wind...). We detected several ways to enhance the guiding algorithm by using encoders which basically consist of closing the feedback control loop in the host controller software (e.g., INDI):

1. Replacing the current motors with an equivalent version with encoders such as the NEMA17-23-01D-AMT112S. This approach will require other equipment such as the motor's programmer, cables and pulleys. Special care has been taken to pick motors with the same number of steps as the original ones (200 per rev), otherwise the gear will not satisfy the firmware expectations.
2. Attaching an external encoder to the mount's gear, such as the AS5145B (a magnetic encoder) or the LPD3806-400BM-G5-24C (a rotary encoder).

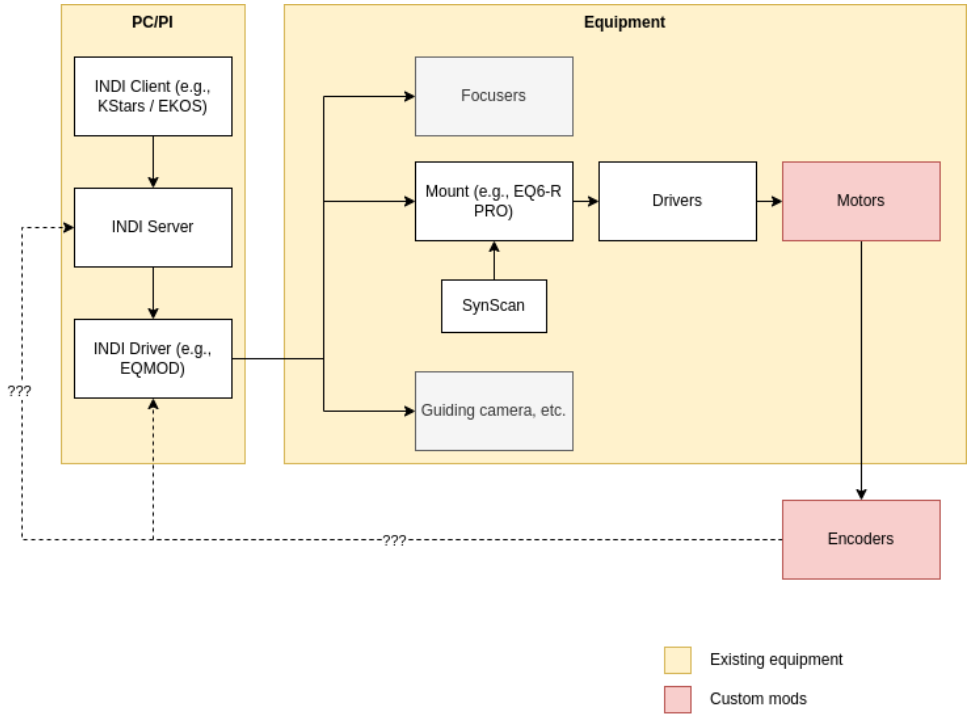

Figure 1. Architecture of a feedback loop through INDI

Another way to solve this problem would consist of the integration of the closed loop in the mount electronics itself. The advantage of this approach is that it doesn't depend on the host tools (like INDI) at all.

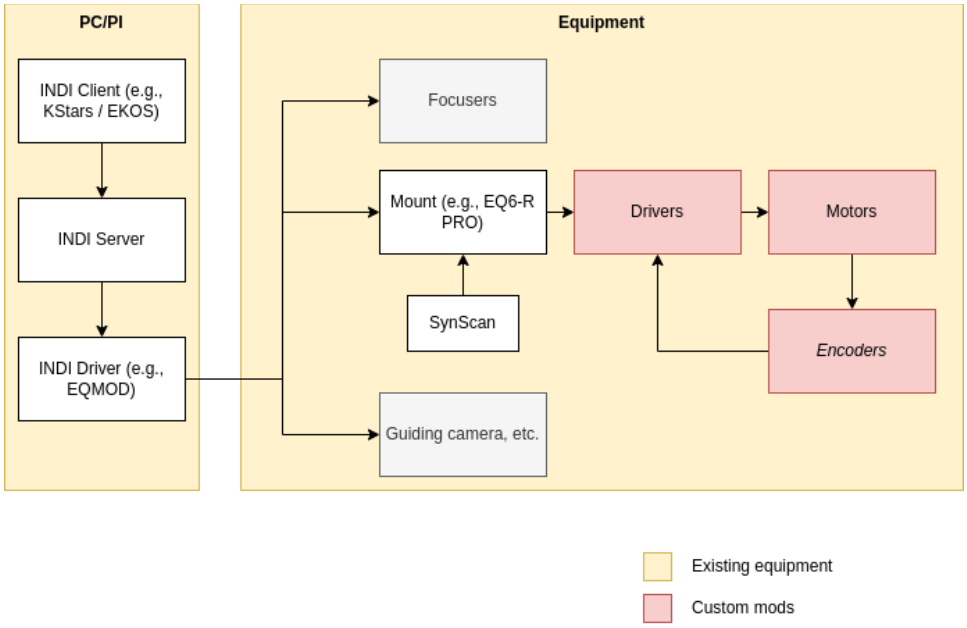

Figure 2. Architecture of a feedback loop at the mount level

Issue 2 – The slew rate

The design of Equatorial mounts –like the one we are using– inherently imposes another limitation to the stability of axes movement. One of the axes of these mounts is aimed to compensate Earth rotation "to allow the instrument attached to it to stay fixed on any celestial object with diurnal motion by driving one axis at a constant speed"[1].

As we are trying to track satellites (artificial objects), we need the two axes (right ascension and declination) to move independently. More importantly, **unlike natural celestial objects, the slew rate of the axes has to be variable**. This interesting discussion emphasizes the need for this need:

> *[…] The satellite tracking is a "leap frog" methodology in which the mount slews ahead of the satellite, stops and then allows the satellite to drift though the field of view, the procedure then repeats itself. Continuous satellite tracking is not possible because of two limitations of the current CEM60 firmware. There are no variable slew rate commands, and also when querying the scope of its current position, the data returned is not necessarily "fresh" - it may be up to one second old which is not sufficient for satellite tracking […]*

At the time of writing there were only a handful of tools which offered some kind of support for satellite tracking, although they didn't seem to have a lot of users. One example would be EQMODLX (see FAQ, which mentions the two issues we previously detected):

> *The EQMODLX is a LX200 Emulator program that accepts satellite tracking issued by a LX200 Compatible Satellite Tracking Software (http://www.heavenscape.com) and converts them to EQ*

[1] https://en.wikipedia.org/wiki/Equatorial_mount

> *custom tracking rate commands. This configuration allows you to track the satellite continuously instead of the "Leap Frog" fashion as made available to the old nextstar protocol.*
>
> *Satellite Tracker must be configured in LX200 GPS Polar mode. This mode allows the speed of the RA and DEC stepper motors to be configured to move at a specified orbital speed allowing continuous tracking of the satellite.*

SkyWatcher has also launched a Satellite Tracker Application which is apparently yielding some interesting results. This tool uses PreviSat Open-Source software to generate the satellite positions (RA/DEC) according to a given TLE. iOptron is another manufacturer working on this feature and they've recently launched a firmware for some mounts that support it.

Results – Testing SkyWatcher's satellite tracking firmware

Solving issue number 1 (adding a feedback loop) would require either adding extra hardware and/or software. However, if mounts were accurate enough (in terms of having the least number of missed steps possible), this step would turn out to be unnecessary. We decided to focus on solving issue 2 (having a variable slew rate).

With this set of tests, **we intended to validate SkyWatcher's tool and check if it relies on leap frogging movement**, which, as stated before, would not be acceptable for our use case. This tracking tool was not available at the time of starting the project, so it was very important to validate it once it was released.

SkyWatcher offers a proprietary application for Windows which was an important roadblock for us, not only due to the Open-Source nature of this project, but also because we needed to understand the communications between the mount and the computer to analyze potential integrations with other tools such as INDI. We contacted SkyWatcher requesting access to the developer's manual and they provided us with a few documents.

This feature relied on an extended command set, which was not available on INDI, however, the support team gave us enough information to develop an Open Source Python tool that replaced their Windows executable. The tool needs to receive a set of Right Ascension and Declination angles of a given satellite every 20-50 ms.

Test 1 – Evaluating motor's movement

The main difficulty we encountered during the first test was the weather conditions. In order to run the test, a clear sky was needed, which was not always possible. In this test we took apart the mount's controller (see figure 3) to decode the signal of the motors' drivers and try to answer the same question: is the algorithm leap frogging?

In this experiment we used:

- Linux laptop
- SkyWatcher EQ6-R PRO / Mount
- Oscilloscope

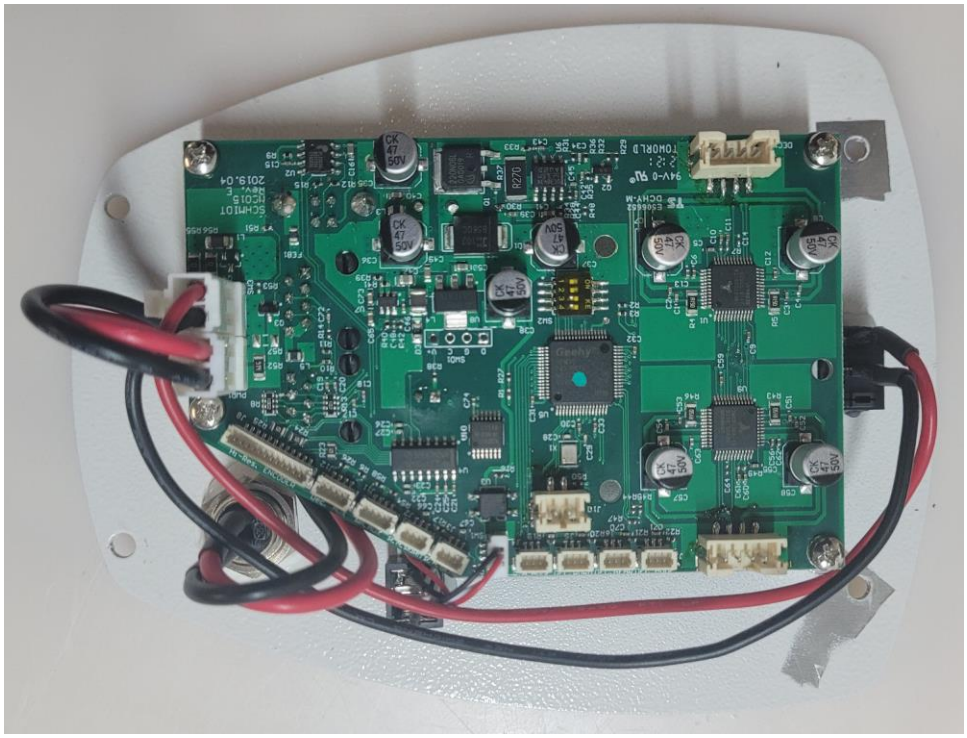

Figure 3. SkyWatcher EQ6-R's mount controller

We expected the drivers to have a quadrature signature, however, we soon realized that this was not the case because the stepper motors of the EQ6 use microsteps, a sinusoidal waveform aimed to increase the smoothness of the movement. This means that computing the axes speed by measuring the signals sent to the stepper motors was not possible with a regular logic analyzer, because these signals are sinusoidal (analog) instead of rectangular (digital) as seen in the diagram below (figure 4). Even if we managed to capture the analog signal, decoding the signals would be very difficult because they are usually proprietary.

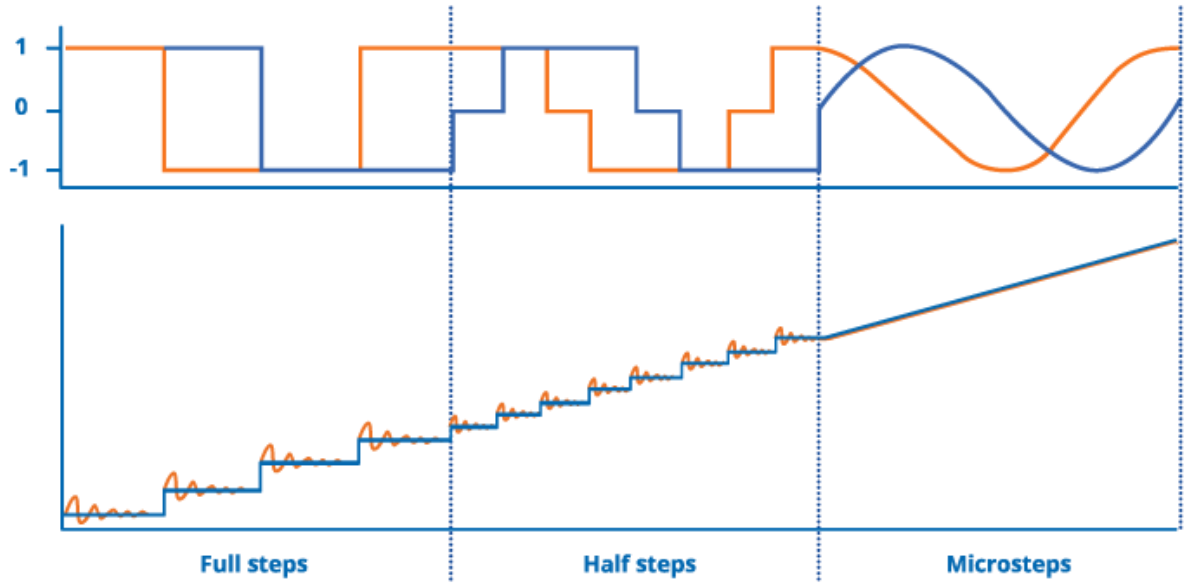

Figure 4. Difference between Full-step drive, Half-stepping and Microstepping

A closer inspection to the mount controller led us to the TMC5130A IC, a motor driver from Integrated Circuits capable of generating 256 microsteps per full step. The mount uses two of these devices to drive each axes' motors. We then used an oscilloscope to visualize the signals sent to the motor (see scope's capture in figures 5-7), which confirmed to us that **this approach would not yield any interesting result**.

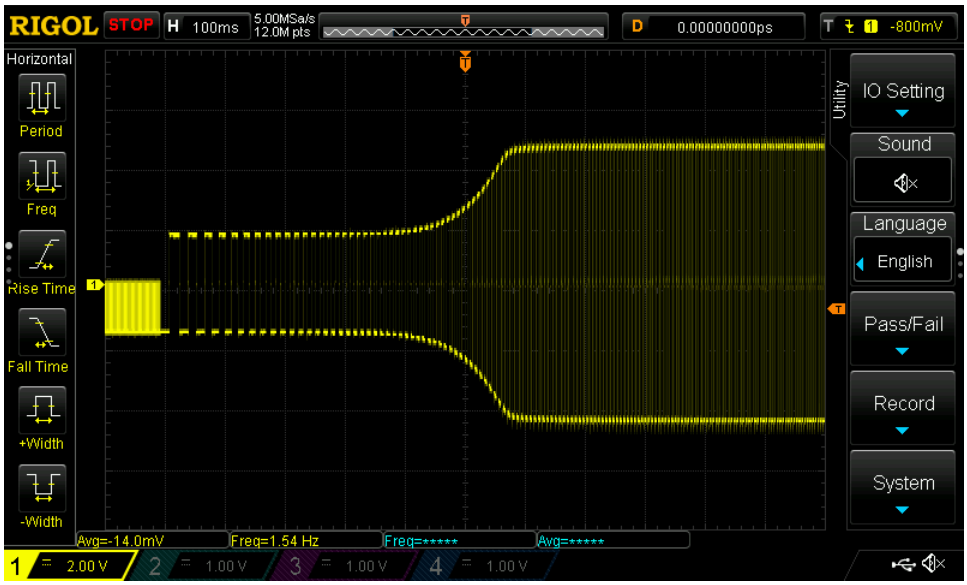

Figure 5. Differential signal corresponding to a single motor pole during ramp-up and nominal speeds

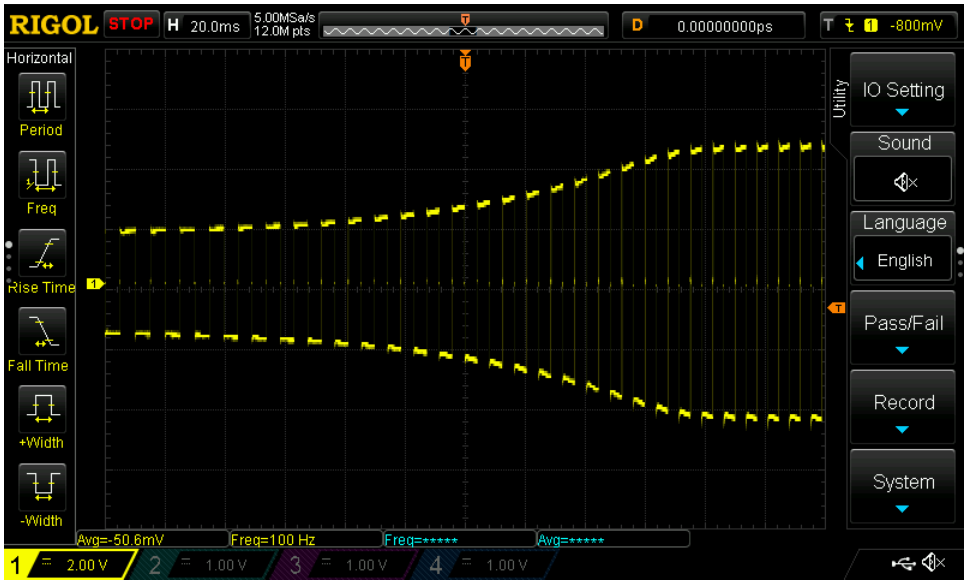

Figure 6. Close up view of Figure X during ramp-up

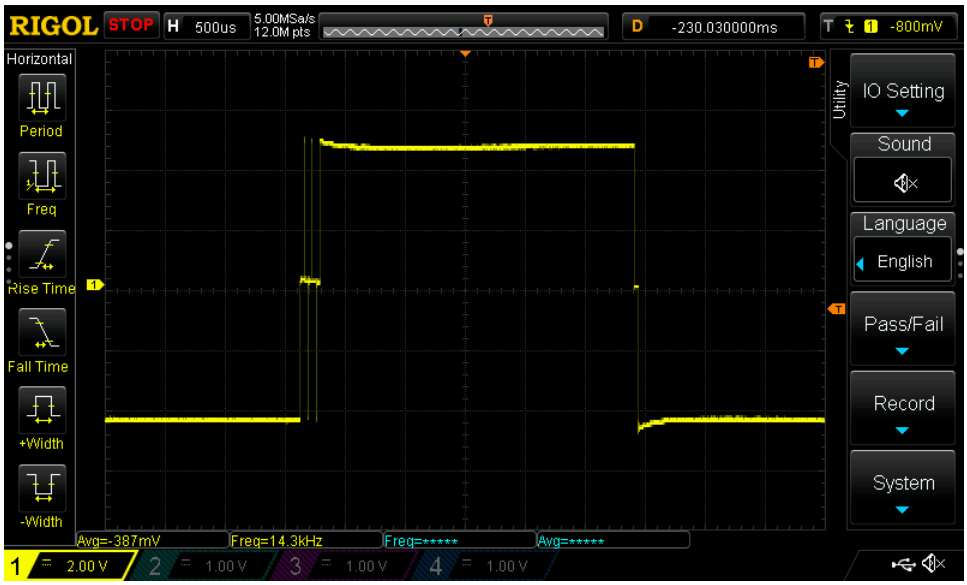

Figure 7. Close up view of Figure X during nominal speed

Test 2 – Evaluating motor's movement (approach B)

This test was similar to the previous one. Given the difficulties we previously found while decoding the signal of the motor's driver, we modified the mount to add an external rotary encoder to one of the axes (see figure 8 below). Again, we expected to see sudden periodical changes in the speed if the algorithm used fixed slew rate.

In this experiment we used:

- Linux laptop
- SkyWatcher EQ6-R PRO with external encoder (LPD3806-400BM-G5-24C)
- Logic Analyzer

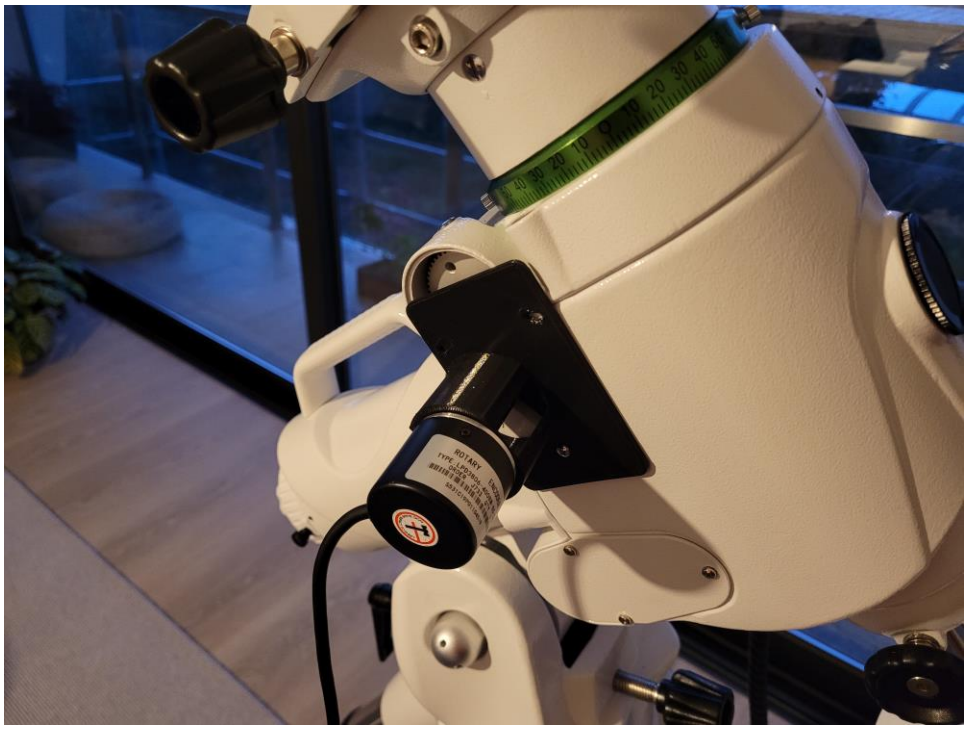

Figure 8. Mount with external encoder

From the results shown in the graphs of figure 9, we can say that, except during ramp up and ramp down, **the axis speed seems to be constant, no signs of leap-frogging.** Even though we can observe some jitter in period/frequency, this is most probably due to sampling inaccuracies or quantization errors.

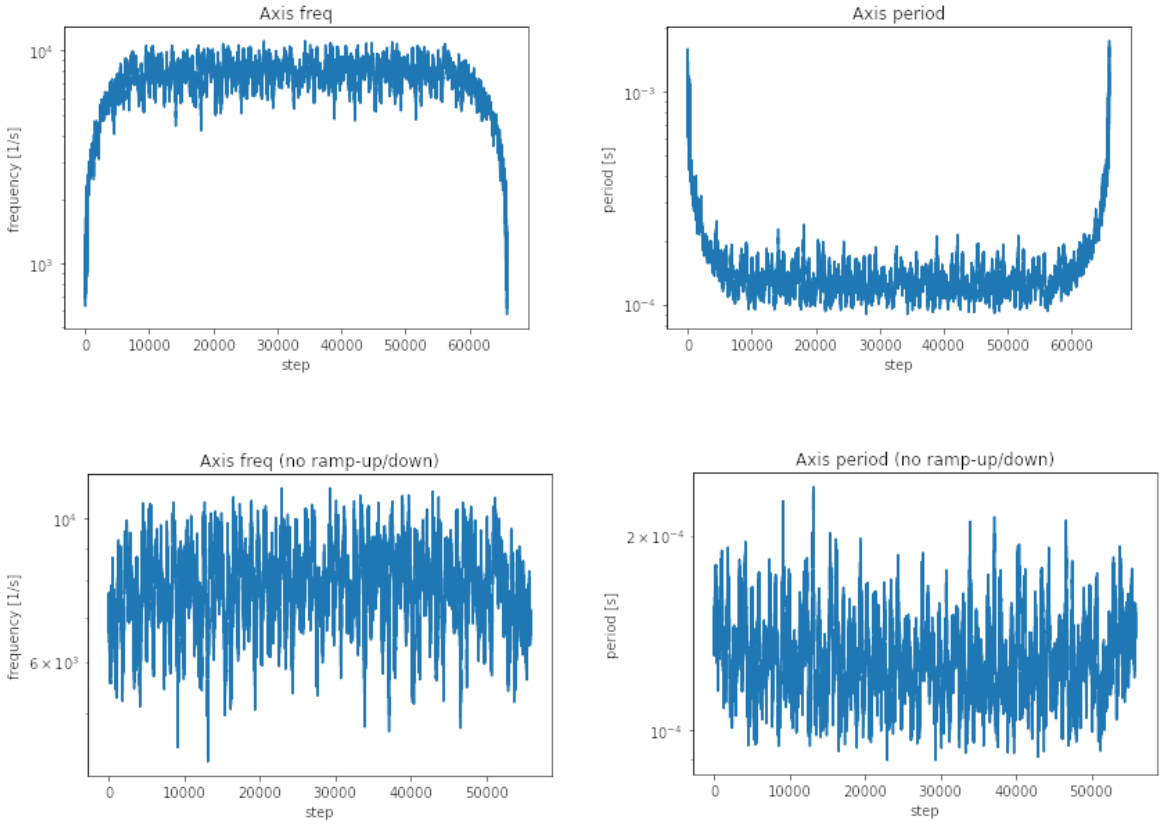

Figure 9. Steps frequency (left) and period (right) - Zoomed in graphs during constant mode (below)

Test 3 – Visual tracking of the ISS

In the previous test we confirmed the mount's motors could move at a constant speed. For this test we run the tracking algorithm using the tool we developed and recorded a video of the telescope. The goal was to track the International Space Station and observe it static on the image frame during the entire pass.

Our testing equipment was based on:

- Linux laptop (with Open Source tools)
- SkyWatcher EQ6-R PRO / Mount
- ZWO ASI 2600 MM PRO MONO / Camera
- CELESTRON TELESCOPE ASTROGRAPH S 203/400 RASA 800 OTA

The results can be obtained on this annotated video, in which we can appreciate that we successfully manage to track the ISS (which is a bright object located in a fixed location of the -moving- image). It must be noted that we were not able to record the entire pass because a meridian flip was needed towards the end. At that point we manually stopped the tool.

The steps to reproduce the test were the following:

1. Use PreviSat to generate the tracepoints data set of a selected satellite (more details in SkyWatcher's official guide). We tracked the ISS and generated tracepoints every 30ms. Using smaller values yielded worse results, even though it seems counterintuitive. We do not have an answer for this behavior.
2. Set up the mount (polar alignment) and slew to Polaris. For this step we used KStars, an open source, cross-platform Astronomy and Ekos, a complete astrophotography solution that can control all INDI devices.
3. Next we are ready to run our tool. Clone the repository and install the dependencies (more on the README). At the moment of writing this, a couple of changes have to be made to the script manually:
    - Get Polaris coordinates (RA/DEC) from KStars in degrees and update tracker.py

```
# TODO use astropy to get current Polaris coordinates or read the current position through drivers
logger.info('Sync to Polaris')
pos_tuple = CelestialPosition(37.9083, 89.26975)
```

    - Update the initial slew direction tracker.py to avoid needing a meridian flip, which has not been tested with this tool

```
# TODO pick slew direction
mount.goto_and_slew(skywatcher.RA_AXIS, coord_prev.ra)
# mount.goto_and_slew(skywatcher.DEC_AXIS, coord_prev.dec)
mount.goto_and_slew(skywatcher.RA_AXIS,
                    coord_prev.ra - 360)  # CCW initial rotation
```

4. Stop INDI. This is an important step because the firmware can't handle both *extra* (the ones added by SkyWatcher for this particular functionality) and regular commands.
5. Launch tracker.py as indicated above. The mount will slew to the first tracepoint and wait there until the ISS arrives. After that, it will periodically send commands through serial to track it.