Waylon Gao – Henry Morris Award Report 2024

Transforming my Astrophotography by designing and building my own star tracking mount

First, what is a star tracking mount, and why is it necessary?

To take a photo of the night sky, a camera set to a long exposure time would be used. This means that the camera is able to catch the faint details of the night sky such as stars, nebulae and galaxies. However, if the camera was simply mounted on a standard static tripod, the stars would appear streaked and elongated. This is caused by the rotation of the earth; as the earth rotates the stars appear to move across the sky. A star tracker is a piece of specialised equipment designed to rotate the camera/telescope to compensate for the apparent movement of the stars, solving this problem.

This is crucial to good astrophotography. Until starting this project, I was limited to only using a static tripod to mount my camera, and consequently the results were rather limited. I investigated buying an off the shelf star tracking mount, however they all cost well over £200! I love engineering and designing and making things to solve problems, so I decided to design and make my own instead.

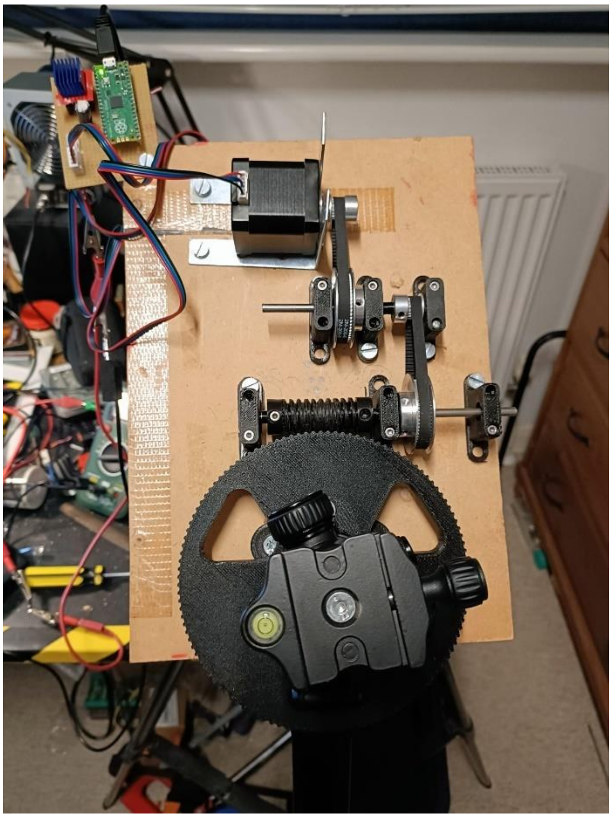
The principle of operation of a star tracker is remarkably simple; the camera needs to counter-rotate at the same speed as the earth (one rotation in 23h, 56m, 04.09s – known as a sidereal day), along the polar axis. This can be achieved by using a motor to slowly spin the camera on an axis aligned (North) to the Pole Star. The earth rotates at a speed of about 0.00418 degrees per second (the sideral rate), which is impossible to achieve accurately by attaching a camera directly to a motor shaft. Instead, I needed to design a gearbox with a reduction ratio of over 1:1000.

Designing and making the star tracking mount

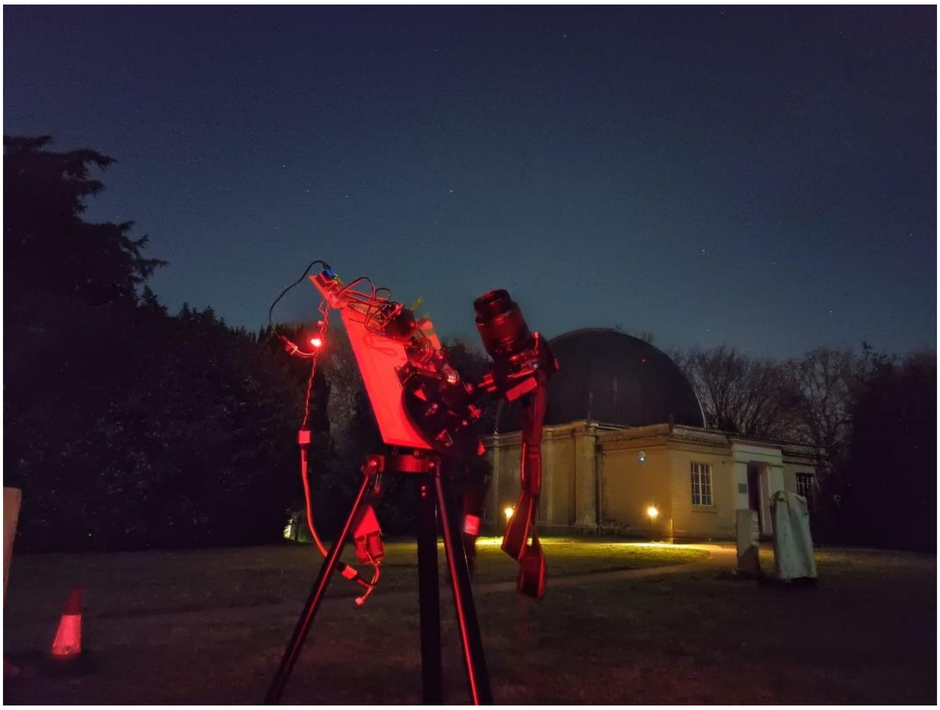
I started off by designing the star tracker mount using, 'Fusion 360', a 3D design software package. Previously, I had used different software, 'OnShape', for such projects however I wanted to progress to a something new and more powerful. I taught myself Fusion 360 with the aid of YouTube tutorials. I started designing my mount straight away with some old metal gears I had lying around from a dismantled machine, and I hoped to use these to reduce the speed of the motor to the sidereal rate. However, the gear reduction of these did not come close to the needed 1:1000. I then took a different approach and investigated using worm gears. It is much easier to achieve a high reduction ratio this way, so I designed my own and printed them on my 3D printer. I combined my custom worm gear with a series of two belts and pulleys to further reduce the ratio. Now, I had to focus on housing this gearbox, motor and camera mount in one assembly. My design went through 4 main iterations, each of which had large improvements over their predecessor.

V1:

For the first iteration, the goal was to make a quick proof of concept to test my gearbox design. I designed and 3D printed some components to mount some ball bearings to hold the 5mm steel rod axles in place. I then mounted these onto a wooden plank using nuts and bolts:

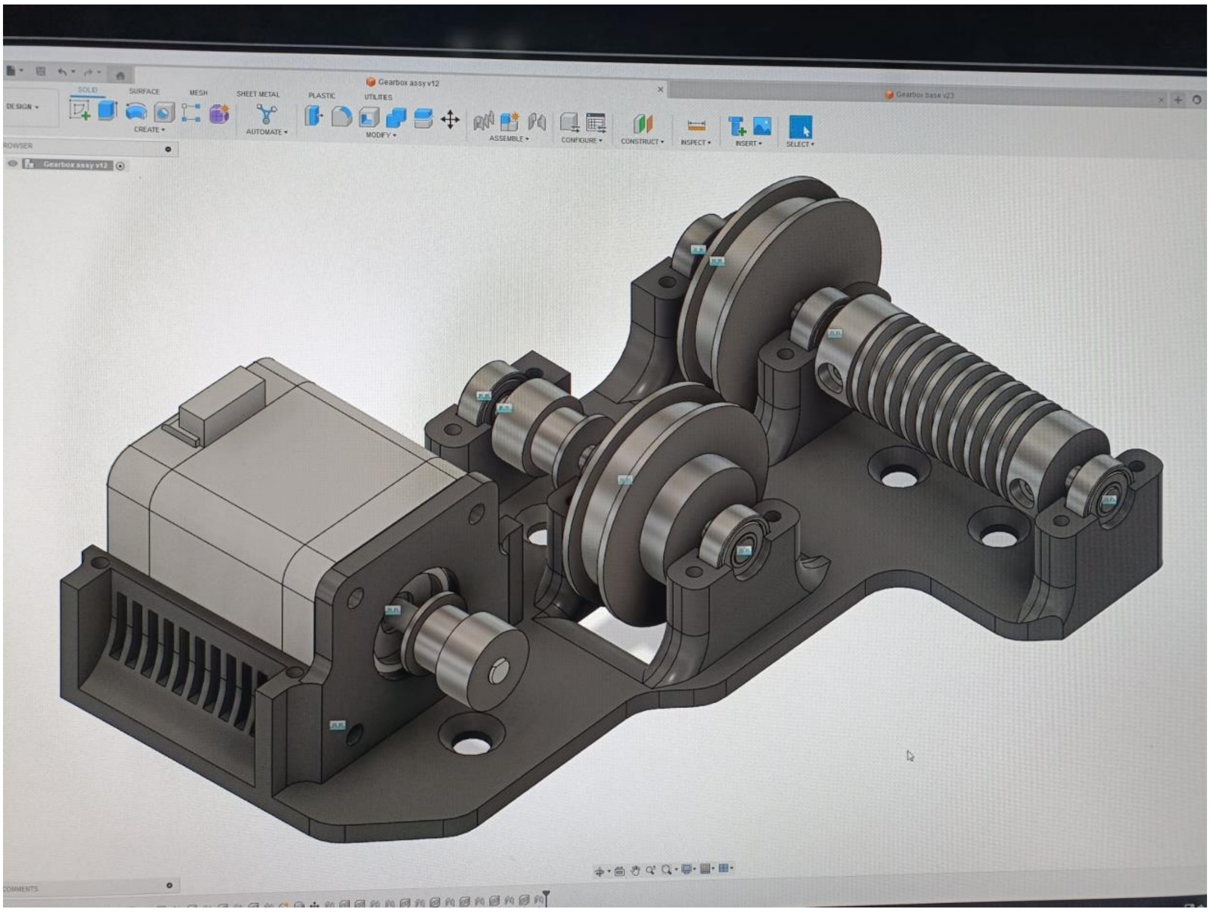


I took this design out to my local observatory to test, and it worked! Whilst the design was finnicky and not close to a finished product, it demonstrated that my gearbox was viable. It also highlighted the issue of belt tension, which I looked to fix in my next version.



V2:

For the second iteration, I wanted to focus on moving the gearbox axle mounts from the wooden plank and into a single 3D printed part. This would make the design much neater, as well as more durable and most importantly, maintain better belt tension. It took a few tries to get the belt tension right, but unfortunately, upon testing, I realized that the belts stretch over time, causing the tracking to drift. The biggest improvement of the V2 design was the development and incorporation of a custom PCB. This made the electronics much more reliable. The PCB will be further discussed later in the report.

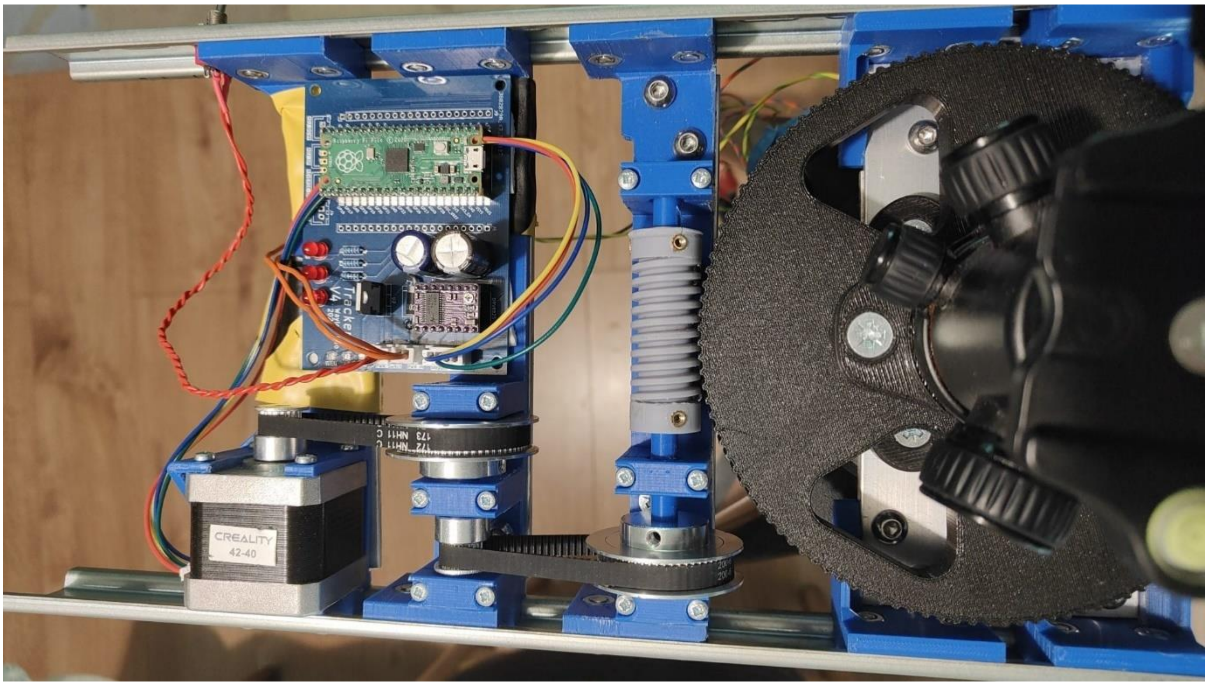


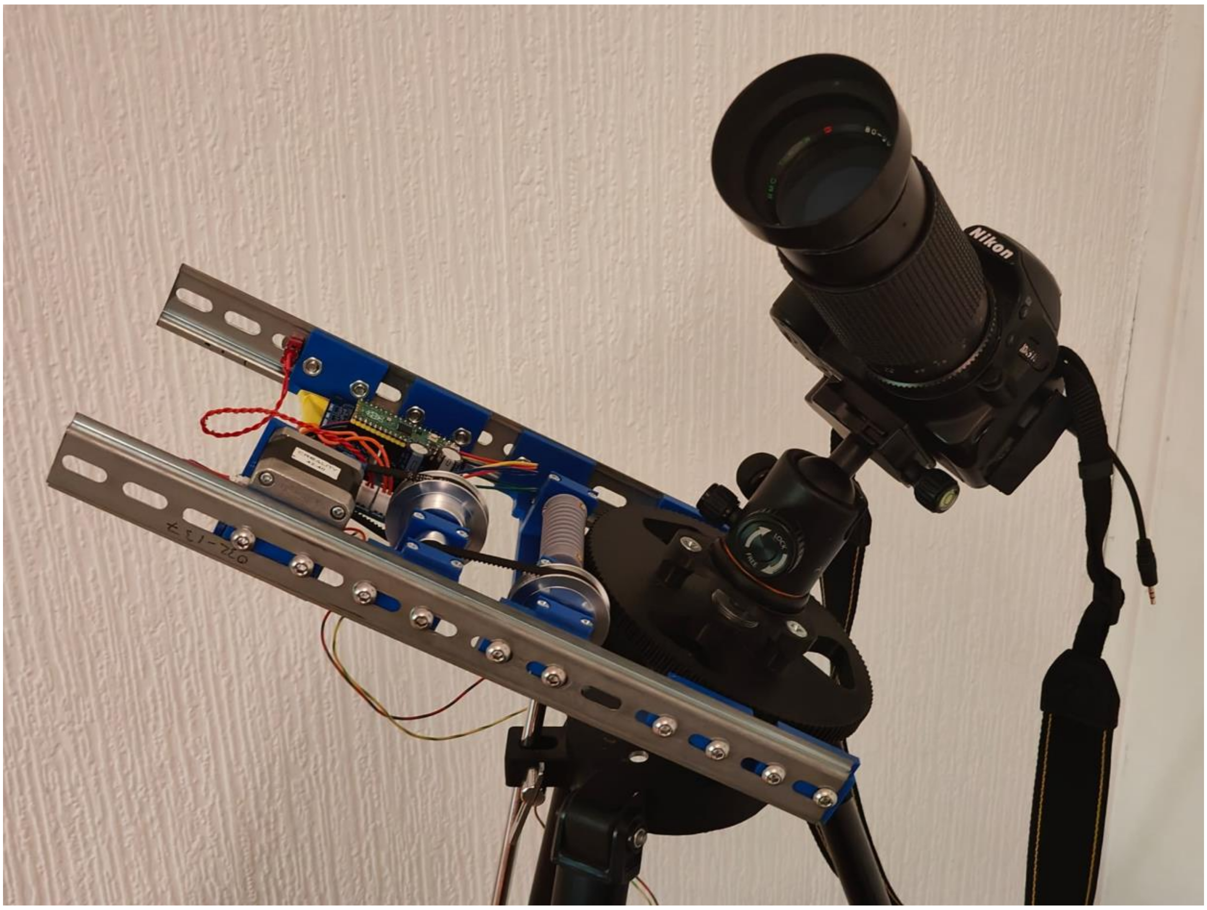
V3:

For the third iteration, I wanted to move everything, including the gearbox, into a single 3D printed enclosure. The previous version was still mounted on a wooden board, so I wanted to make the design neater and more polished. I worked on this design for a while, until I realised that even though I would increase the spacing of- the axles, the belts would still need re-tensioning every now and then due to stretching.

V4:

For the final iteration, I wanted to combine what I had achieved with all previous versions, learning from the design issues encountered in each iteration. I came up with a different approach, to make the design modular. The modularity of this new design meant that I could easily adjust the belt tension, whilst keeping the design as a single compact unit. I took a different approach by mounting each stage of the gearbox on a metal rail, clamped in with bolts. This meant that adjusting the tension in the belts was simply a matter of loosening, moving and then tightening the bolts. I was also able to switch to a smaller stepper motor to extend battery running time.

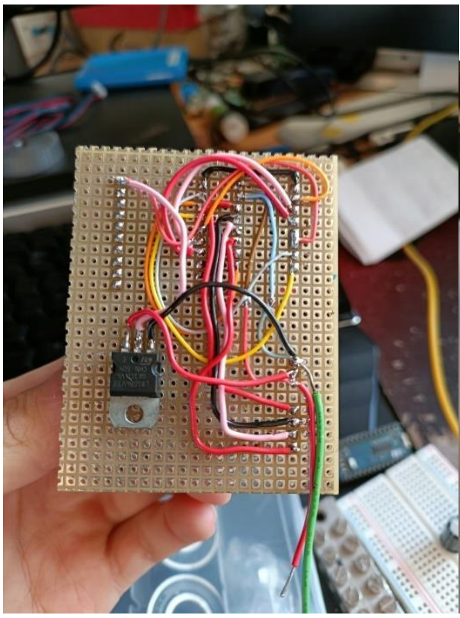
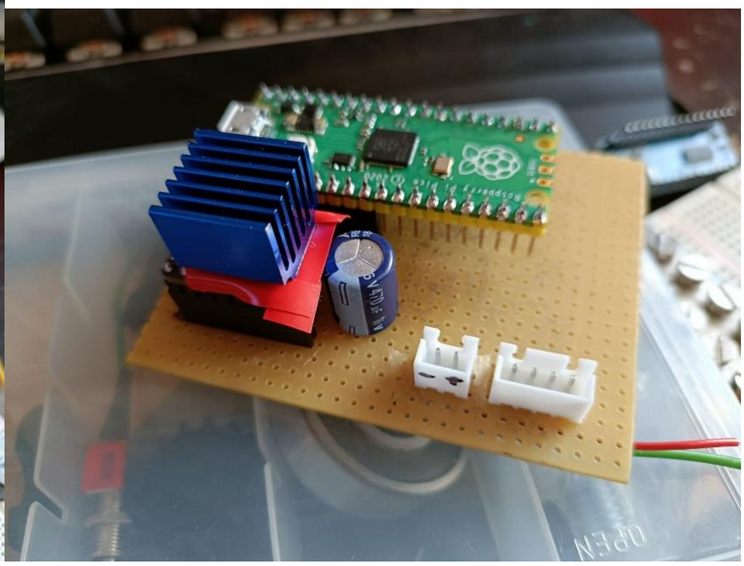




Electronics design: Making the stepper motor work

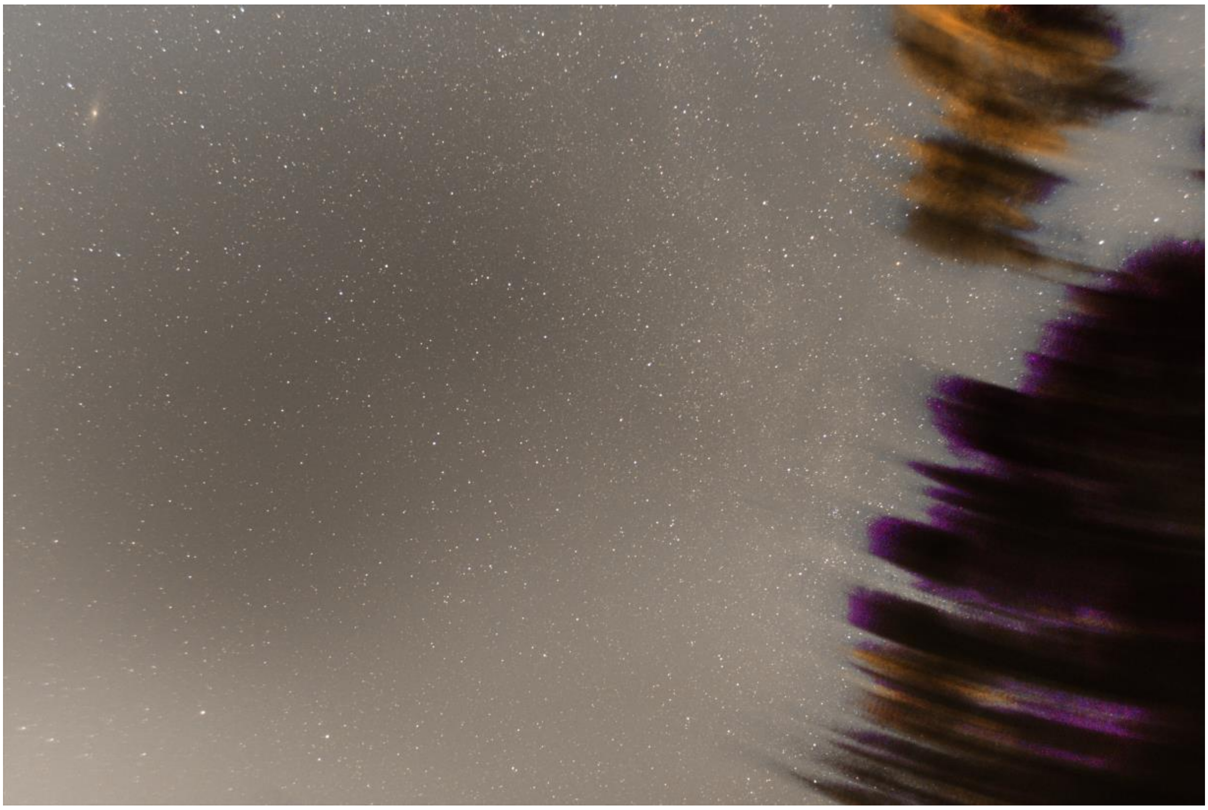
I had a stepper motor dismantled from an old machine, so I decided to try and see if I could use it in this project. I started off by researching how to drive it. Since it had 8 wires, it was a unipolar stepper motor. I found a guide on how to convert it to a standard bipolar stepper motor, which allows for the use of common stepper motor drivers. Once I had rewired it, I connected it to a L98N H bridge driver.

After writing some Arduino code to output a step drive to the driver board, I was able to make the motor spin! However, it was rotating in fairly large increments which meant it lacked the precision needed for my star tracker. Moreover, the motor and driver became extremely hot very quickly, wasting considerable energy and jeopardising other components. I measured the current being drawn, and calculated the power draw to be over 70 Watts! To solve this problem, I settled on using a conventional NEMA17 standard stepper motor and DRV8825 driver board. I made a prototype control circuit board using Perfboard, a Raspberry Pi Pico, and a DRV8825 driver board, and found it to work very well!

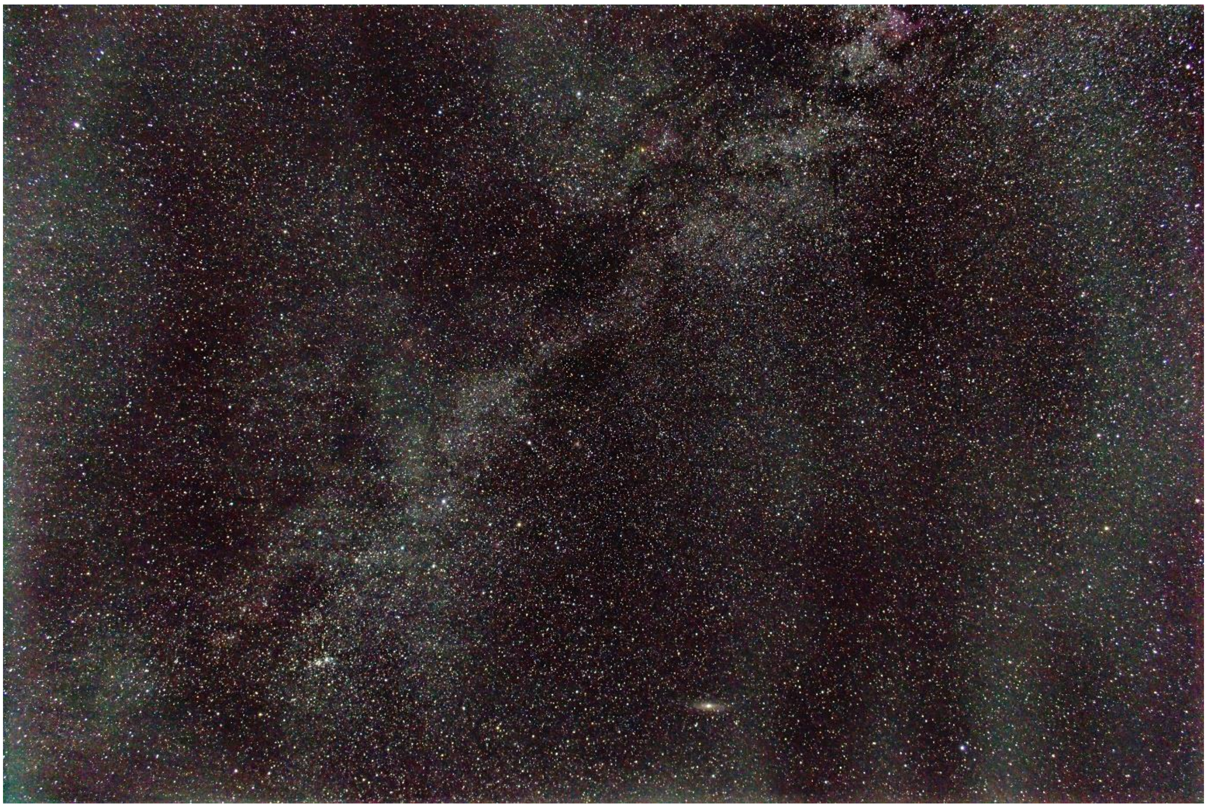



Testing the mount

Now that the star tracker was completed, I decided to take it out for a test. I first tried to use it in my back garden. I used a wide angle lens to take a photo of the Milky Way. I used my phone, set at 30 seconds exposure at an angle of 18mm. I set my camera to start the series of 200 exposures, and left it to run. After I processed the images, here is the result:



Although the sky was mostly obscured in the image, the stars were distinct dots unlike the streaky smudges I had in previous images. I was delighted with my first result, and I couldn’t wait to take my tracker out and take more photos! A few days later, I went to a local park and imaged the Milky way again:



I am very happy with the result! I was able to align my mount to achieve very accurate tracking which allowed for a 35 second exposure time. Even though the exposure time was long, the stars still showed up as small dots, meaning the tracker was doing an excellent job! I was so happy to see this, and especially the final result of the image after stacking. I then decided to take another photo of the Milky Way, this time, focusing on the area near Deneb. As I was taking in a zoomed in shot, I was able to take this from my back garden and avoid the surrounding foliage. Here is the image:

I am very happy with the result! Several deep sky objects are visible, including the North American Nebula and Elephant's Trunk Nebula.

As I was taking these images during the Perseid meteor shower, I also accidentally captured this image of a large meteor:

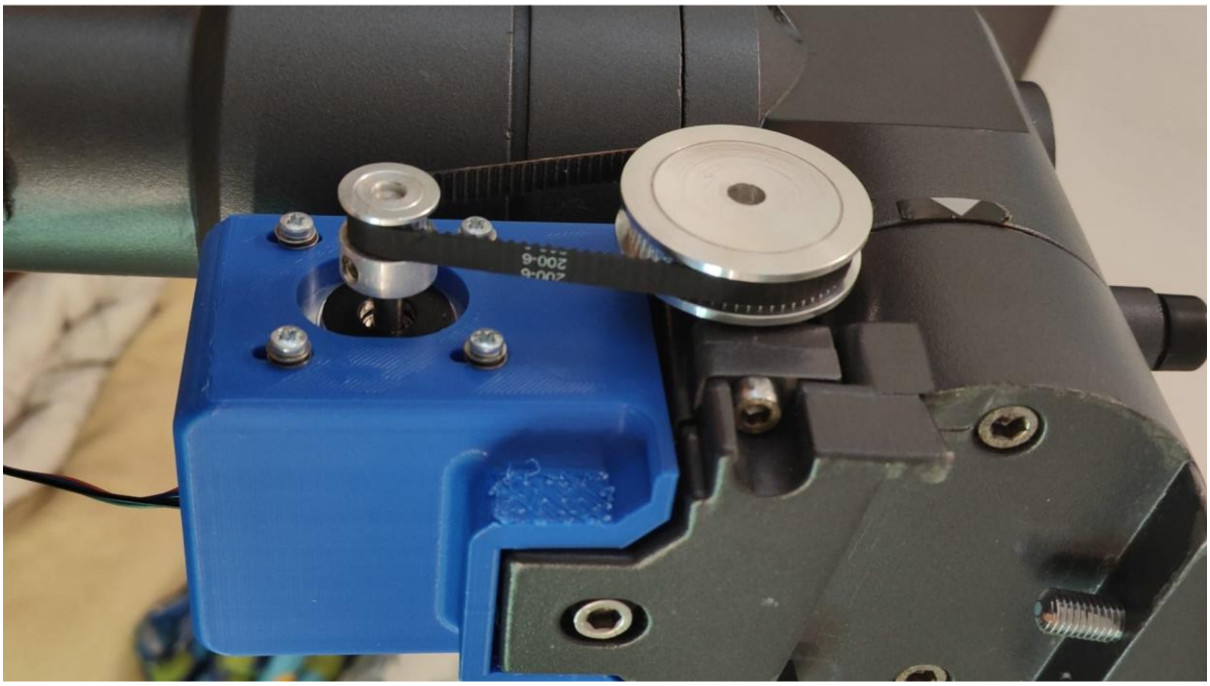
Processing the images

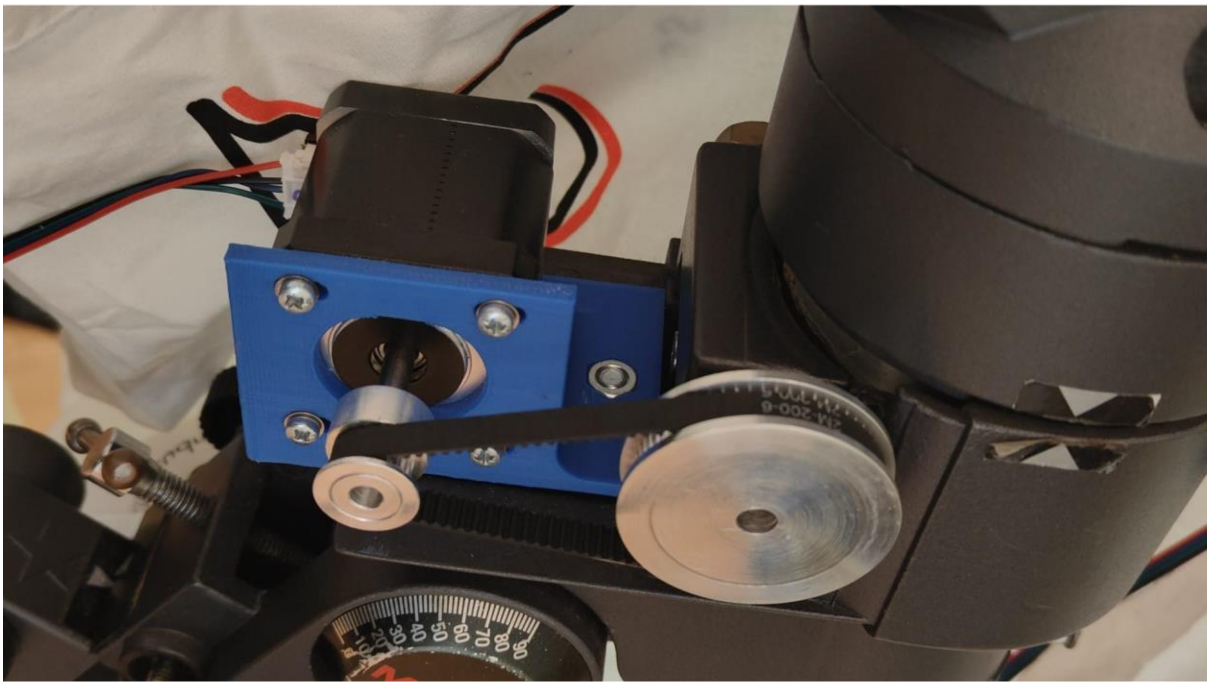
Even with a decent sky tracking system, individual images are still faint and multiple images must be combined by 'stacking' them in software. Further processing with the aid of calibration frames helps remove noise and other artefacts such as satellite and aircraft trails to reveal the full quality and detail of deep sky objects.

Once I had taken enough images, I processed them to create a nice final output image. During the processing of the images, noise and other artefacts in the image such as vignetting are removed. This is done with the help of various calibration frames: Dark frames, light frames and flat frames. Initially I used 'DeepSkyStacker', however the milky way result was a bit disappointing, even after trying different settings. I then tried to use 'Sequator', a piece of astronomical image processing software designed specifically for Milky Way shots. I was amazed by the difference in image quality; Sequator brought out lots of the finer details that DeepSkyStacker was not able to reveal. I also used Sequator to process the image of the Deneb region.

Motorising my telescope mount

Now that I had a working star tracker for my DSLR camera, I wanted to apply what I learnt and designed to enable the same tracking for my telescope using my much larger telescope mount. This very old mount had internal gearing but no working motors or drivers. I started off by disassembling and lubricating the gears inside the mount. This reduced the friction and made the mount more accurate and easier to turn. Next, I took some measurements and designed the motor mount for each axis. I 3D printed these and installed them, attaching the belts and pulley wheels to the worm drives.

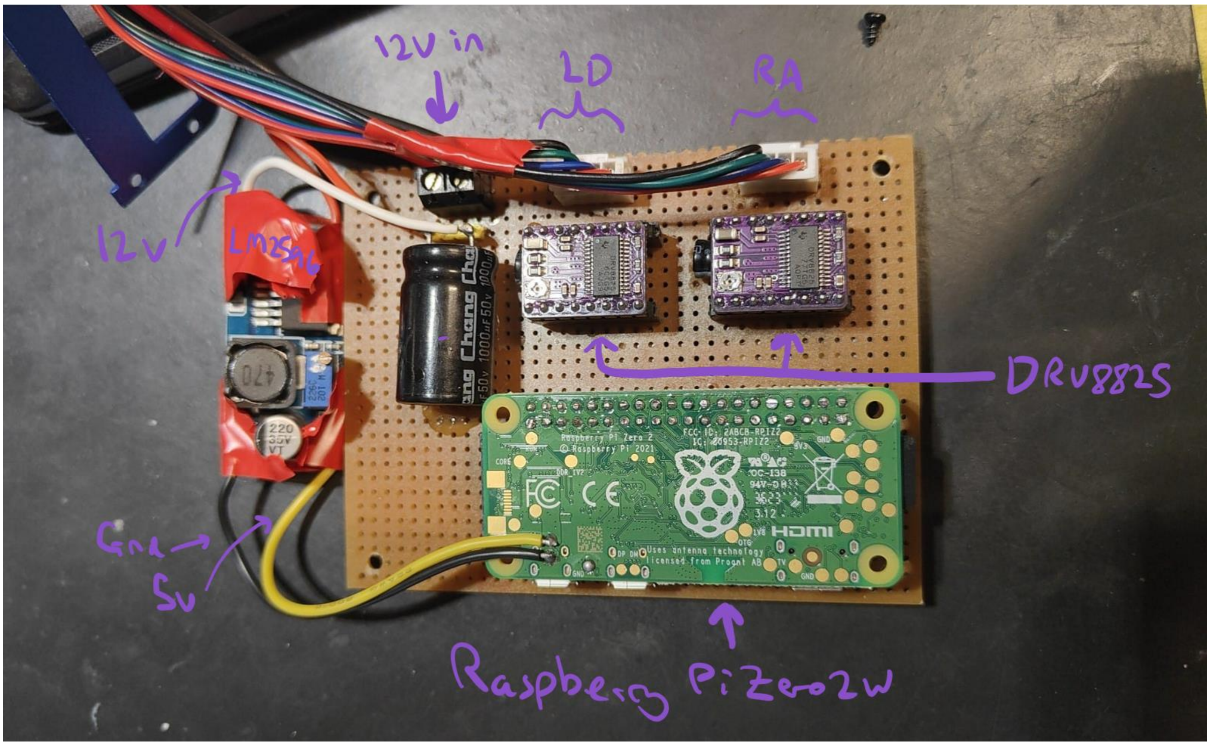


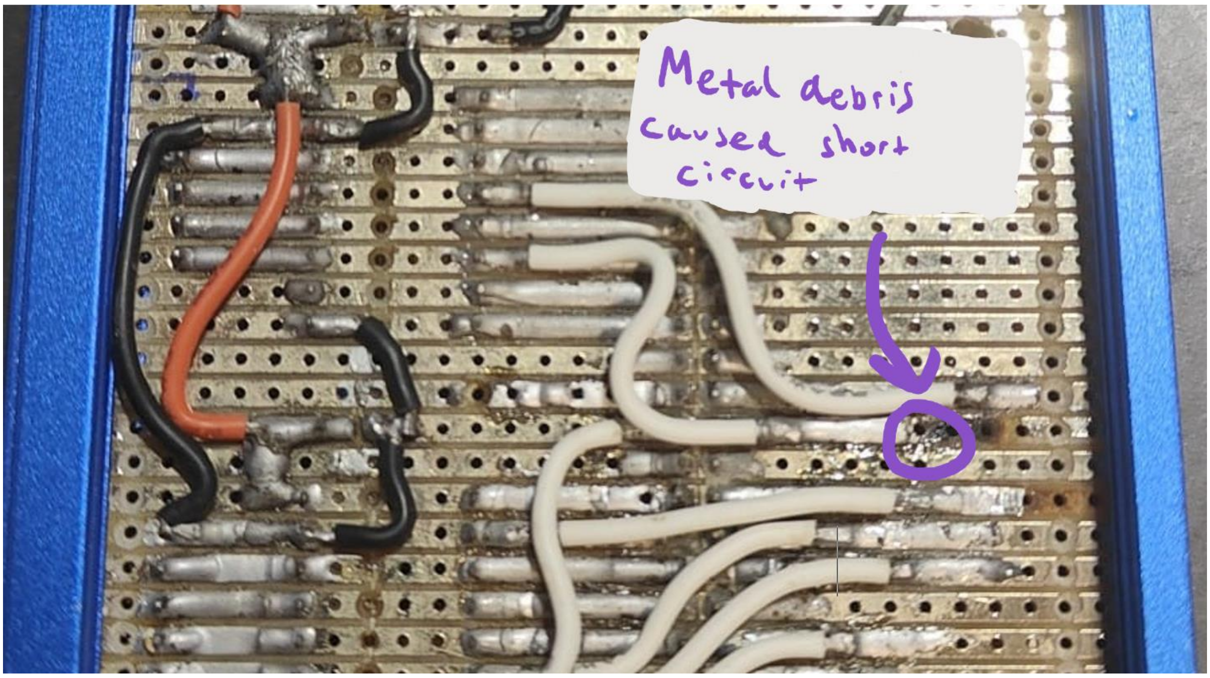


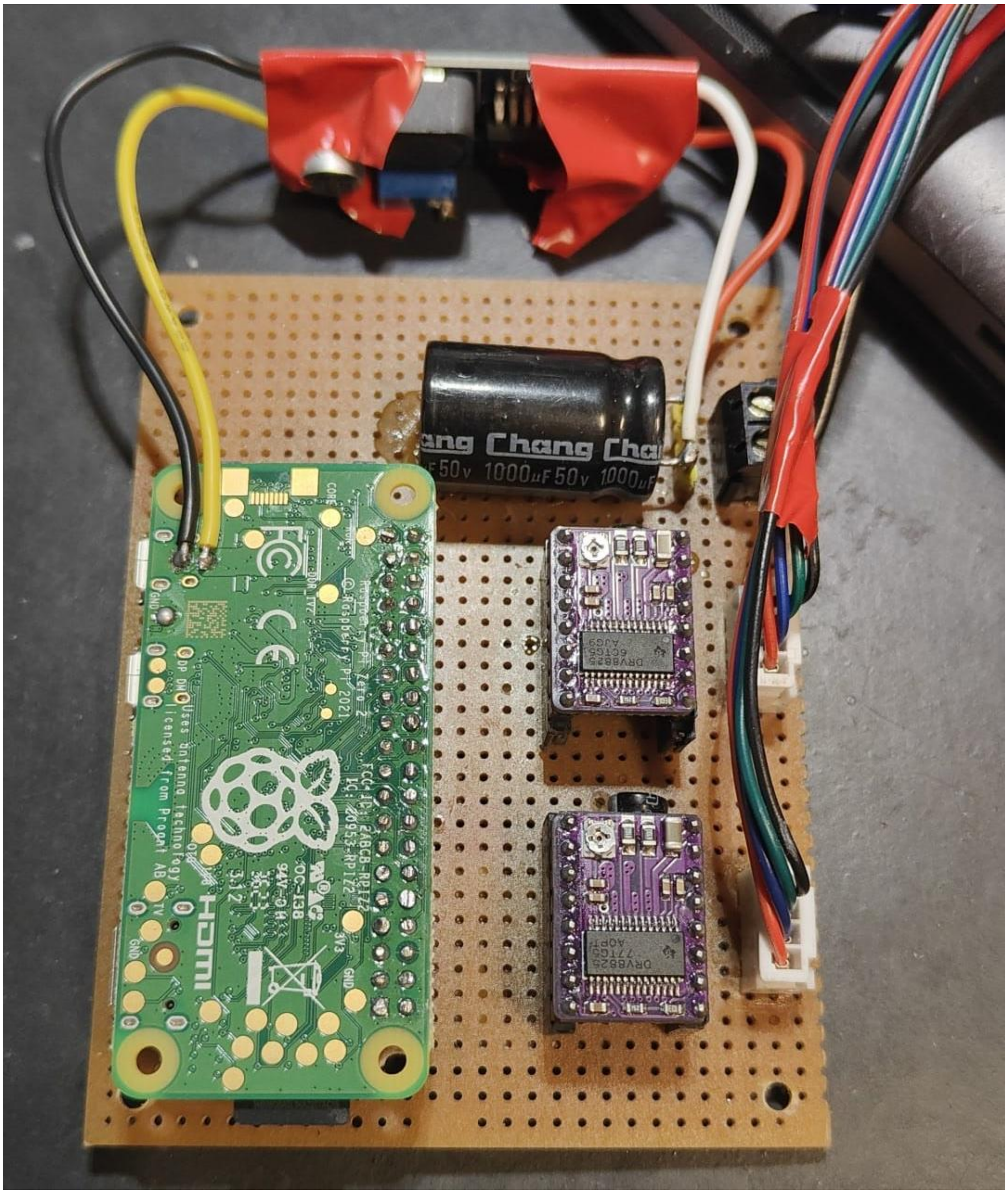
Now that the mechanical side of things was done, I needed to design and build the electronics to control the motors. I considered designing a custom PCB, however I wanted to save time so I just made a circuit on Perf board. I used a Raspberry Pi Zero W, two stepper motor drivers and some capacitors for power smoothing. I used the Raspberry Pi's GPIO pins to interface with the stepper motor driver.

Initially I used a L7805CV linear DC voltage regulator to convert the 12V DC input into 5V for the Raspberry Pi. However, upon connecting power to the system, the Raspberry Pi blew up. Even though the input to the raspberry pi read as 5V, there was some strange fault in the system. I then checked the GPIO pins, until I found that one of the GPIO pins had been short circuited to the 12V supply line by a miniscule stray fleck of solder. I removed this and double checked the pin, and it was ready for a retry, this time with a new Raspberry Pi.

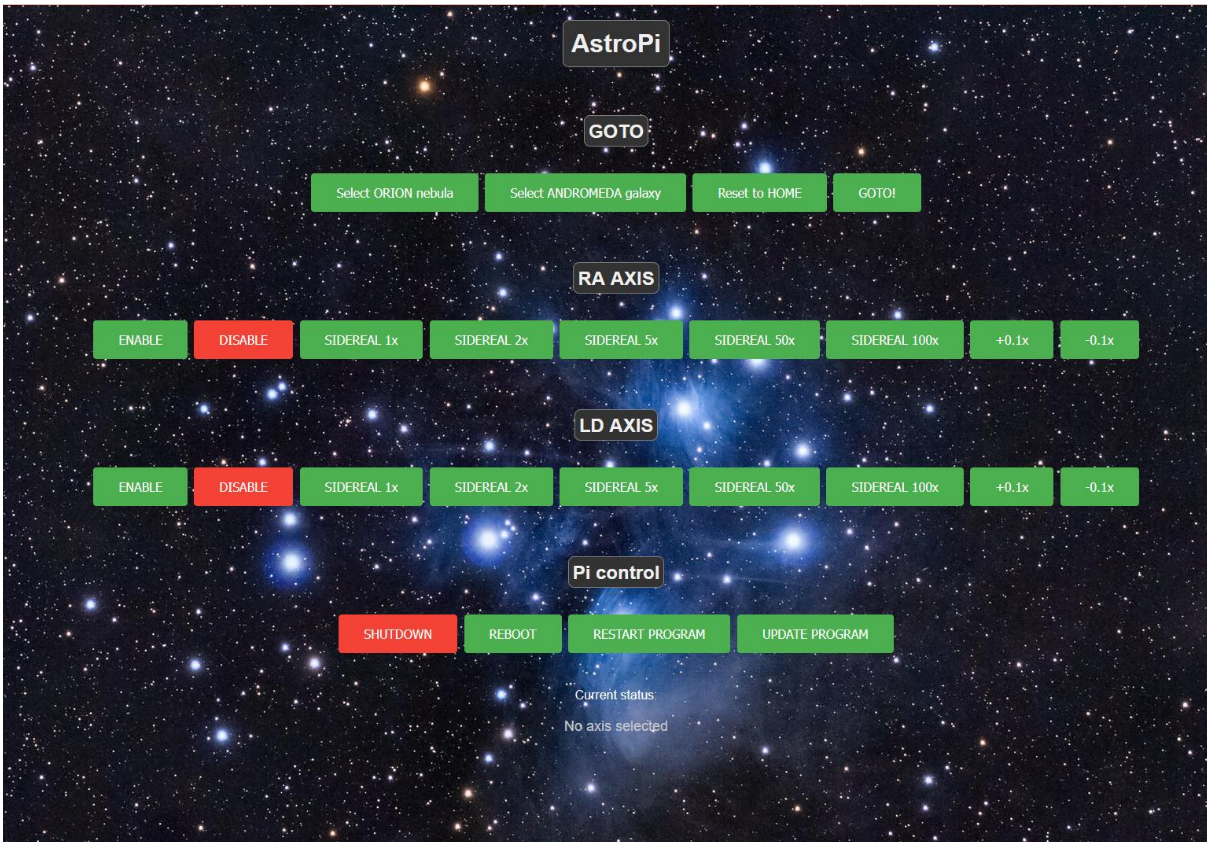
I powered up the system, but the Raspberry Pi fried itself again! I repeated the same procedure of checking the voltage levels of the pins of the Raspberry Pi, and then found another pin where there was the same issue. A tiny metal shaving had become lodged between the signal and power supply line which fried the Raspberry Pi. I removed the metal, and then used Isopropyl Alcohol and a toothbrush to thoroughly clean the board of any further stray metal debris. I bought yet another Raspberry Pi, and this time it finally worked!


12V in
LD
RA
12V
LM2596
DRV8825
Gnd
5v
Raspberry PiZero2W

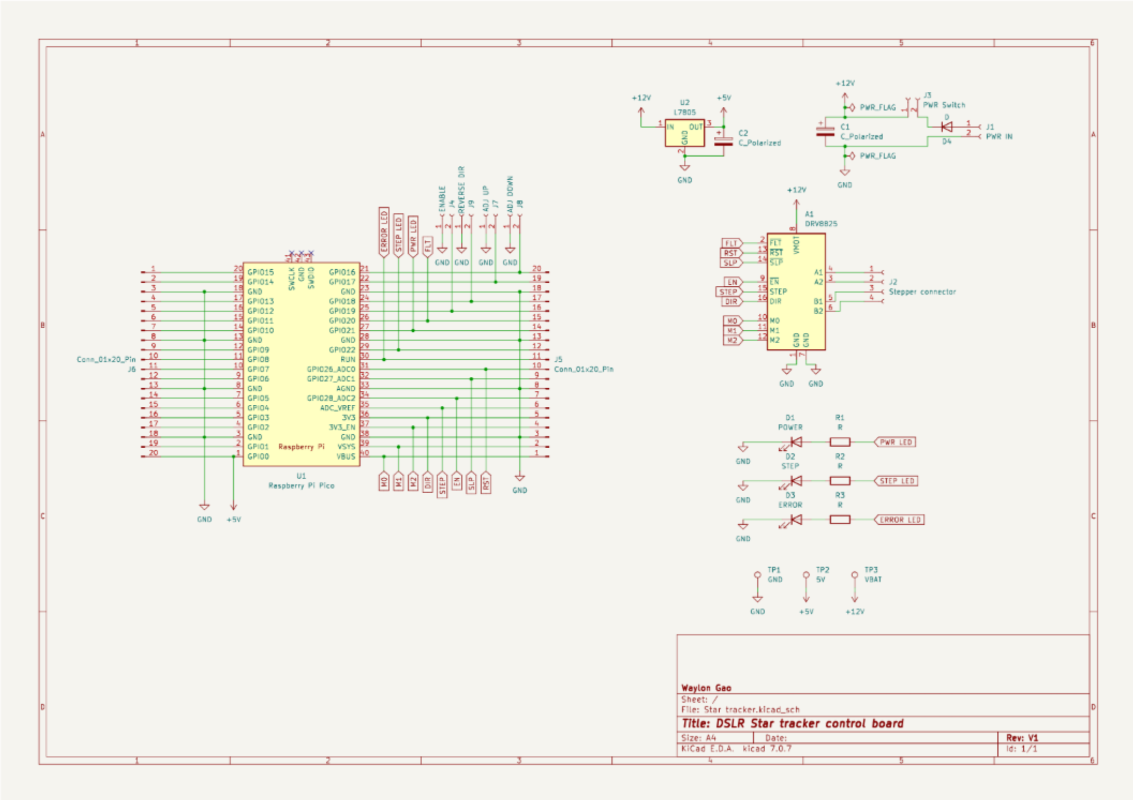

Metal debris caused short circuit

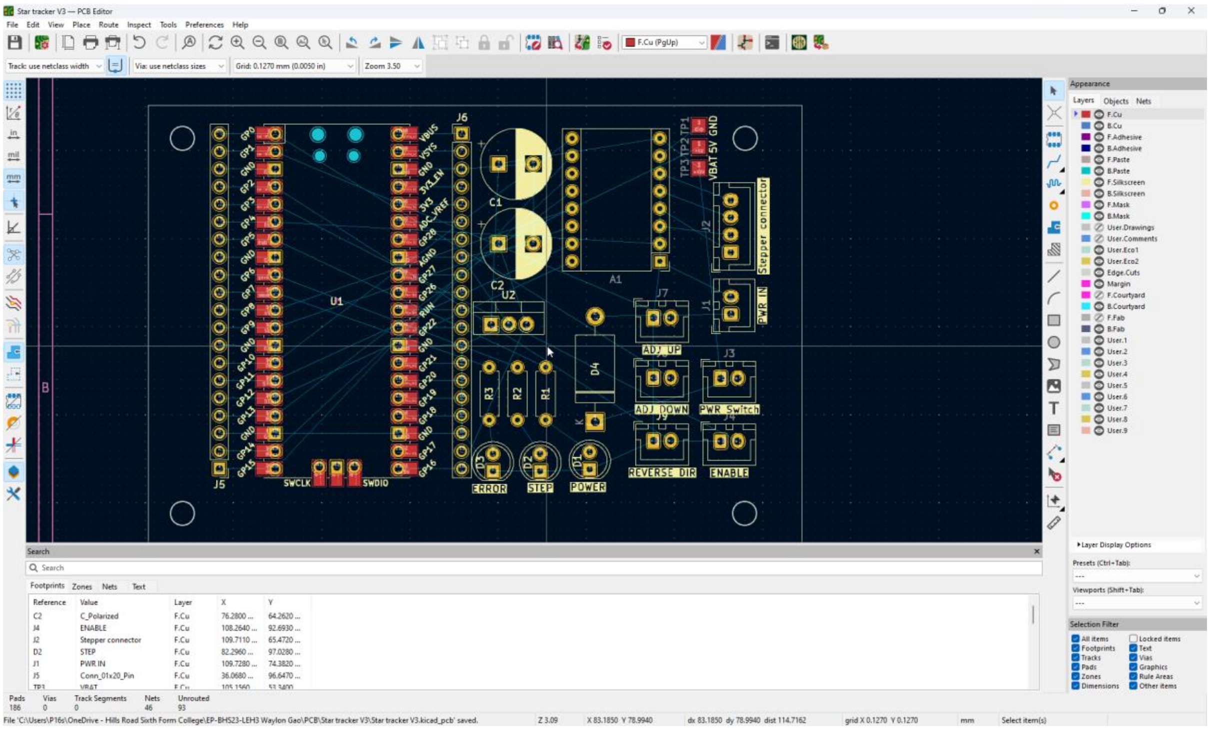


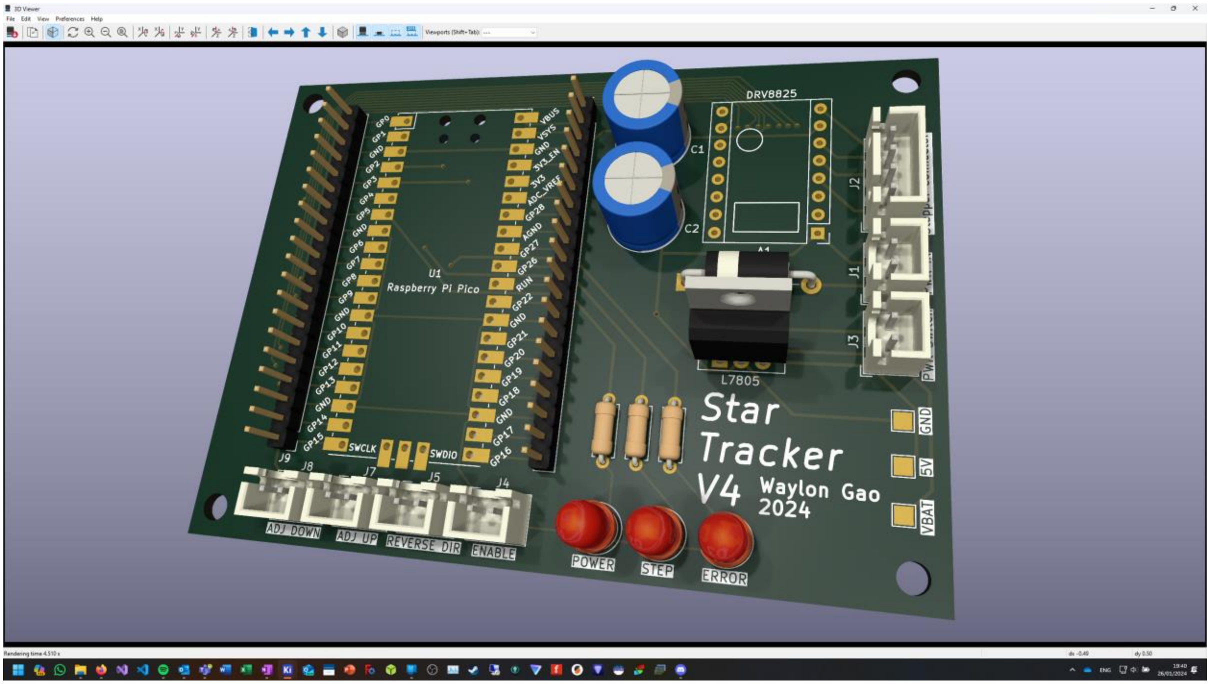
I then moved onto writing the code to run on the Raspberry Pi computer. I used Python to write a simple website which would run on the Raspberry Pi. The website enabled the user to change the tracking settings, as well as point the telescope at a particular target. It took some time to implement this “GoTo” feature, however I am very satisfied with the result.

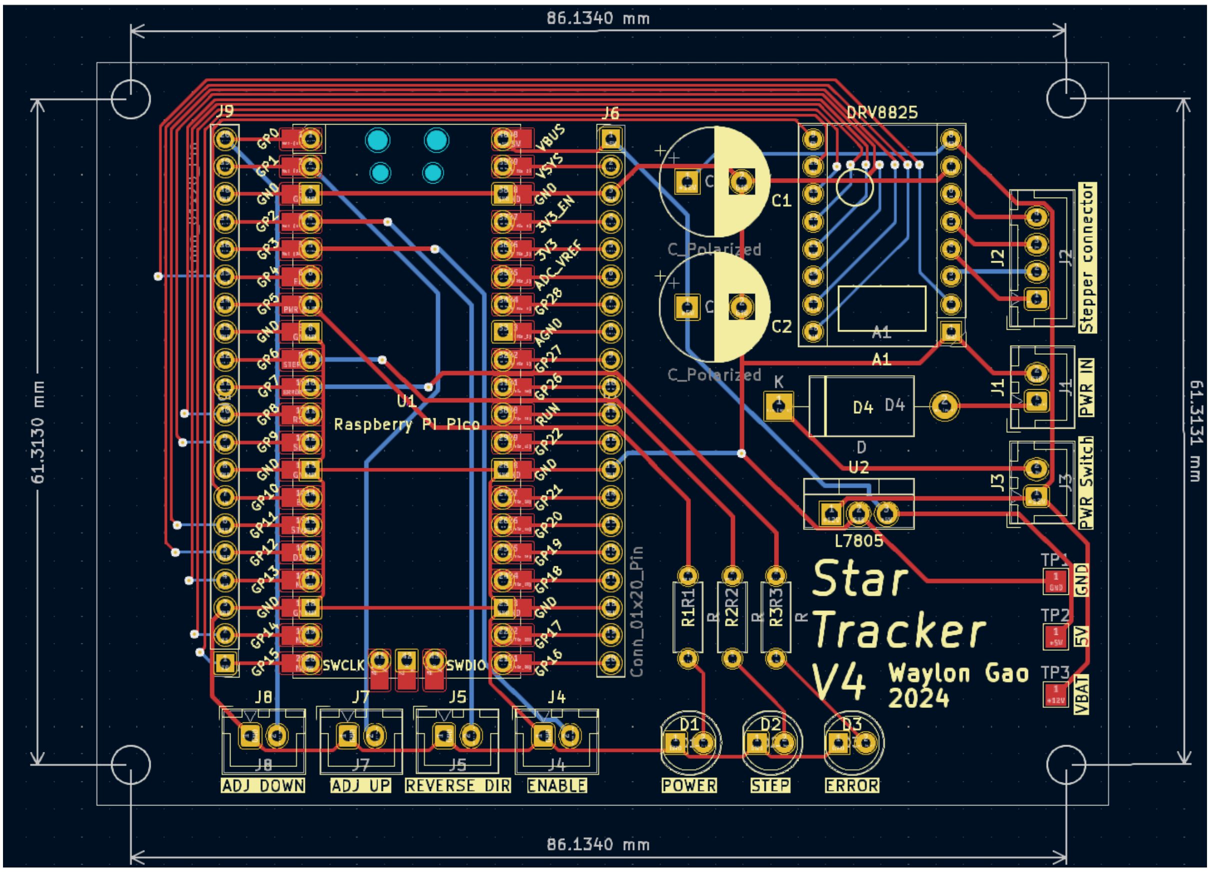


Now that I had a working circuit, I wanted to make a custom printed circuit board (PCB). I taught myself KiCad, a free and popular PCB design software. I then went through 4 design iterations until I was happy with the fourth version. I placed an order on JLCPCB, and just over a week later, 10 of my very own PCBs arrived. I was pleased that they worked as intended.


U1
Raspberry Pi Pico
A1
DRV8825
U2
L7805
C1
C_Polarized
C2
C_Polarized
J1
PWR IN
J2
Stepper connector
J3
PWR Switch
J5
Conn_01x20_Pin
J6
Conn_01x20_Pin
Waylon Gao
Sheet: /
File: Star tracker.kicad_sch
Title: DSLR Star tracker control board
Size: A4
Date:
Rev: V1
KiCad E.D.A. kicad 7.0.7
Id: 1/1

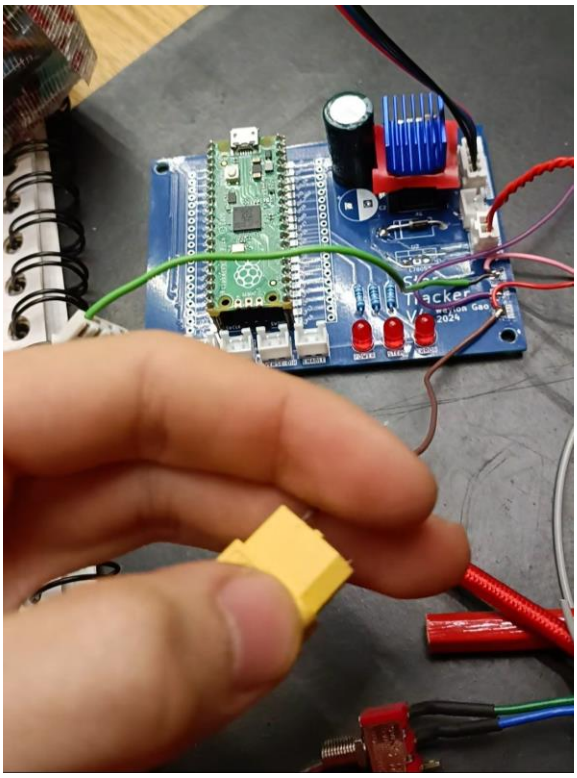

Star tracker V3 — PCB Editor
File Edit View Place Route Inspect Tools Preferences Help
ADJ_UP
ADJ_DOWN
PWR_Switch
REVERSE_DIR
ENABLE
ERROR
STEP
POWER
Stepper connector
PWR_IN
SWCLK
SWDIO
Appearance
Selection Filter





I then placed an order at a PCB factory in China, and received the boards a week later!


41989267
OrderNo:Y3-3882879A 1-1
QTY:10 Panel 1 HASL 1.60mm Blue
Delivery:2024-01-30
InternalNo:2142302WH2 Design:1
test-qc HKTHZXR-RMB
GerberFile:FACTORY



Overall, the project has been a huge success! I am very happy with the results of both star trackers and how they transform my astrophotography. It has unlocked a whole new world of opportunities, enabling me to photograph the myriads of dim objects in the skies. I look to further develop my telescope mount by adding automatic tracking error compensation using a guide camera and a computer, enabling even more accurate tracking.

As this project was part of my EPQ, also made a 7 minute presentation about my project which can be found here:

https://youtu.be/LJaRMj-m2IE